Image Inpainting for Engineering Image Datasets

Gauri Jagatap, Viraj Shah, Vignesh Suresh, Prosper Punitharaj

1. Introduction:

Inpainting is the process of reconstructing lost or deteriorated parts of images and videos. In the museum world, in the case of a valuable painting, this task would be carried out by a skilled art conservator or art restorer. In the digital world, inpainting (also known as image interpolation or video interpolation) refers to the application of sophisticated algorithms to replace lost or corrupted parts of the image data (mainly small regions or to remove small defects). We have proposed to use deep learning techniques to reconstruct the lost pixels in images. Deep learning is a class of machine learning that uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. The output of the previous layer is used as input for the next layer. Unlike the other machine learning techniques, deep learning tries to learn sub features from features with which the efficiency of the learning algorithm (classification or regression) will be more. Two types of datasets are used:

1. Celebrity dataset
2. Metal plate dataset

2. Celebrity dataset:

CelebFaces Attributes Dataset (CelebA) is a large-scale face attributes dataset with more than 200K celebrity images, each with 40 attribute annotations. The images in this dataset cover large pose variations and background clutter [3].

3. Metal plate dataset:

The data set consists of images that are generated as a result of 3D shape measurement using Structured Light system (SLS) technique. SLS technique uses a camera and a projector for creating 3D images of the objects being measured. The object to be captured is exposed to sinusoidal patterns and the camera captures the distorted patterns. 3D image of the same is reconstructed using the camera images.

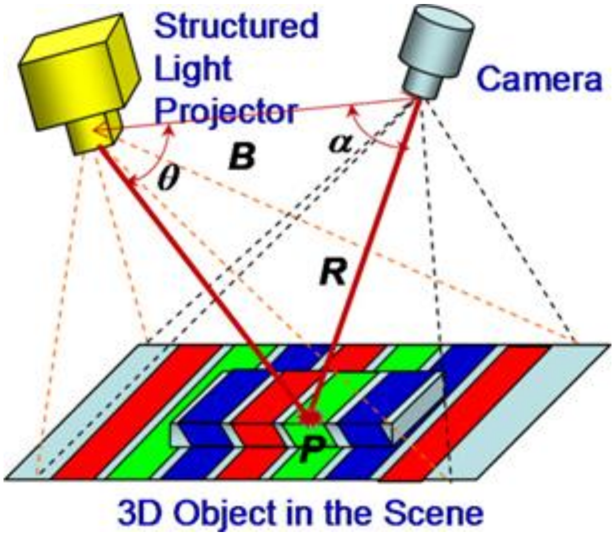


Figure 1: Typical SLS system

The object used in the dataset is a metal plate (Steel-300X300).

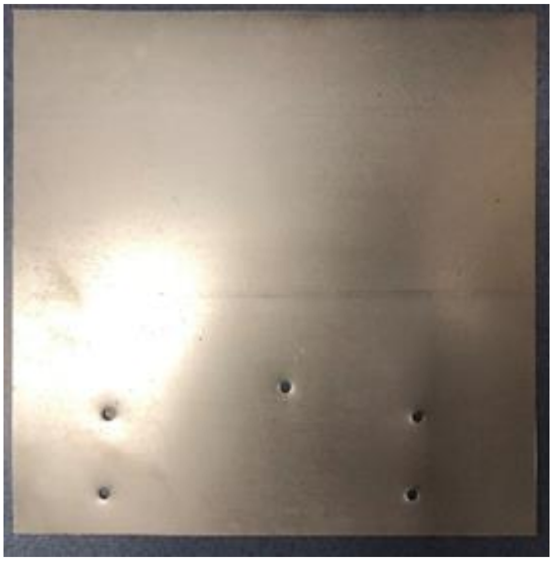


Figure 2: Sample capture of the steel plate

Saturation is an unavoidable problem in objects with high contrast. Many pixels in the object (saturated one) has values more than 255 (8-bit image) so the photograph of the same has a lot of white zones (saturated pixels). This problem was solved using the HDR mode in cameras where the object is exposed to many exposures of light and the one with minimal saturation is used. 3D photography also has the problem of saturation. Though many HDR methods were developed for the same, they tend to have a compromise on the speed of the capture. This prevents it from being real time. Suresh et al [2] proposed a technique to overcome the problem of saturation, it doesn't work well when the objects have a glossy finish.

So we proposed to use image inpainting technique using deep learning to solve the above mentioned problem. The object used for capture is shown in Fig.2. 3-step phase shifting algorithm along with dual frequency is used for 3D reconstruction using SLS.

3.1. Three step phase shifting algorithm:

$$I_1(x,y) = I'\,(x,y) + I''\,(x,y)cos[\emptyset(x,y) - 2\pi/3] \tag{1}$$

$$I_2(x,y) = I'\,(x,y) + I''\,(x,y)cos[\emptyset(x,y)], \tag{2}$$

$$I_3(x,y) = I'\,(x,y) + I''\,(x,y)cos[\emptyset(x,y) + 2\pi/3] \tag{3}$$

$$\emptyset\,(x,y) = tan^{-1}[\,(\sqrt{3}(I_1 - I_3\,))/(2I_2 - I_1 - I_3)] \tag{4}$$

In a single frequency method, 3 images are captured for the three phases (eq.1-3). The output image which is a combination of the three phases will be an unwrapped phase i.e., there will be a phase jump of 2Π (tan inverse function ranges from -Π to +Π). So an unwrapped phase is obtained by using the equation 4. Since two frequencies are used to enhance the 3D reconstruction method, 6 camera captures are required for generating a 3D image. The dimension of the image used obtained from SLS technique was 546X424. So the number of samples should be more than (546X424) to have a good efficiency (for the algorithm).

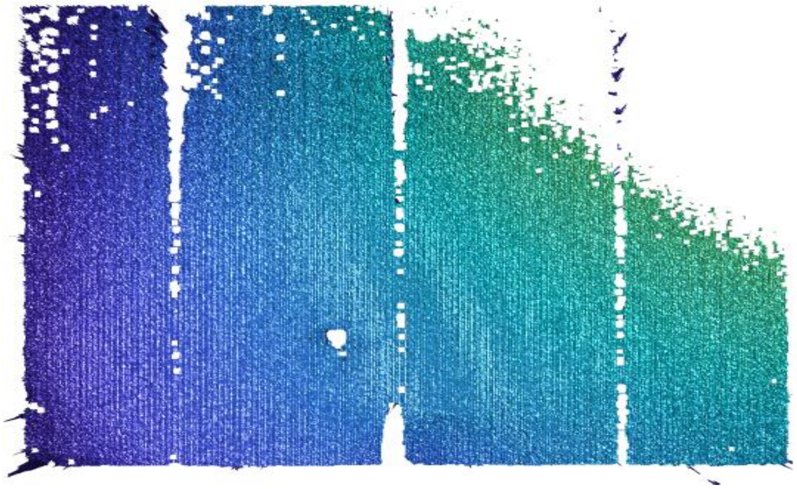


Figure 3: Sample train image

From the image (Fig.3) 16 patches of size 136X106 were extracted.

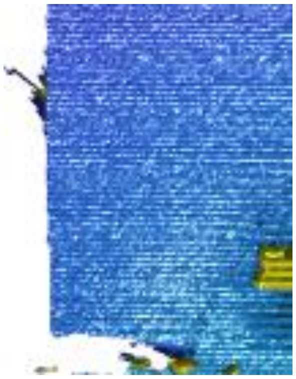


Figure 4: Sample patch image

The patch images (train and the corresponding ground truth) were used for training the algorithm.

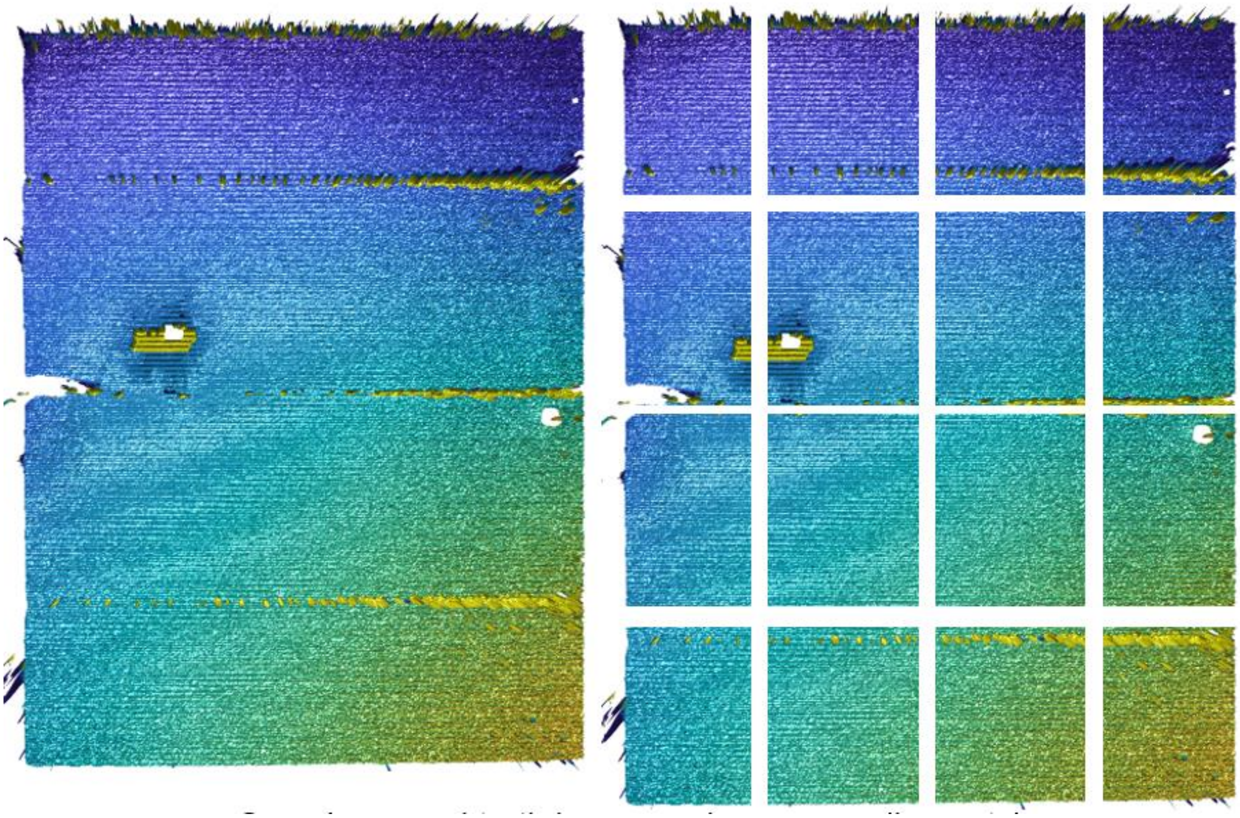


Figure 5: Sample ground truth image and it’s corresponding patches

4. Literature review:

The following are some of the currently used methods for image impainting:

1. Hand based signal priors:

   a. Linear inverse problems of the form: $min_x \frac{1}{2}||y - Ax||_2^2 + \lambda\phi(x)$

   b. Prior ϕ is usually handcrafted for ex. $\phi(x) = ||W_x||_1$

   c. +requires less data, wide variety of inverse problems can be solved via same optimization routine.

   d. - cannot perfectly model datasets → lower accuracy

2. Learning-based methods:

   a. Train large datasets to learn the actual mapping between input/output pairs

   b. If $y = Ax$, then learn the linear inverse mapping $f = A^{-1}$

   c. + can solve individual problems such as inpainting or deblurring or super-resolution with high accuracy

   d. - requires re-training to solve different problems

3. Deep generative models:

   a. Dataset is assumed to have prior distribution → estimate P(x) or learn joint distribution P(x, y).

5. Image Inpainting:

The following are the three approaches that can potentially solve the above mentioned problem:

5.1. Approach 1: Using Deep Generative Models:

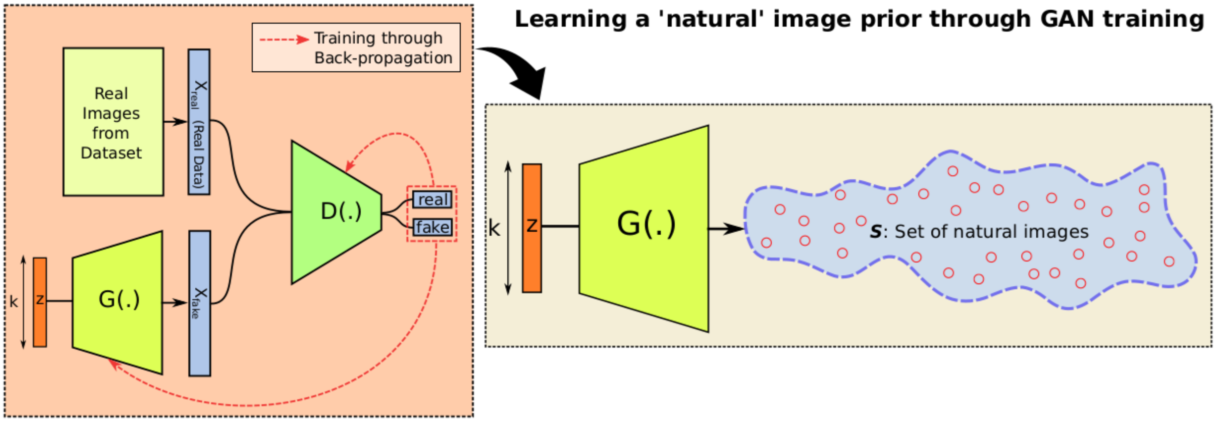


Figure 6: Deep Generative models

This method uses a pre-trained generative model such as Generative Adverserial networks (GAN) or Variational Auto Encoders (VAE) to approximate the set S, a set of all natural images without any requirement of the hand-crafted prior. It can be said that the prior is learnt by the Network itself. The next step is to restrict the solution to lie within the set S. This can be ensured by minimizing the following l-2 loss (Eqn. 5) over the latent variable z. This approach is depicted in the Fig. 7 [4].

$$L(z) = ||y - AG(z)||_2^2 \tag{5}$$

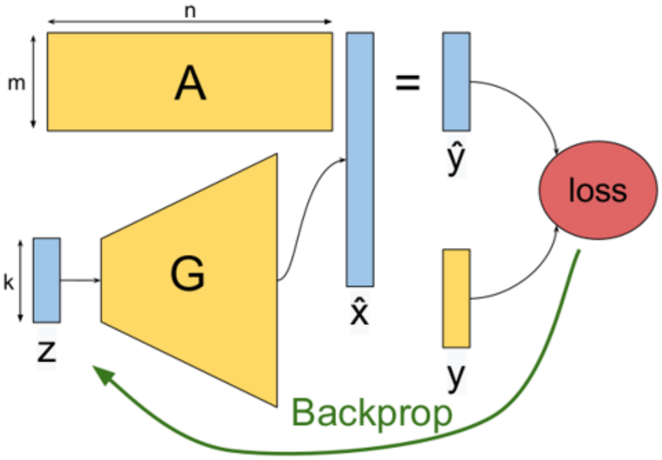
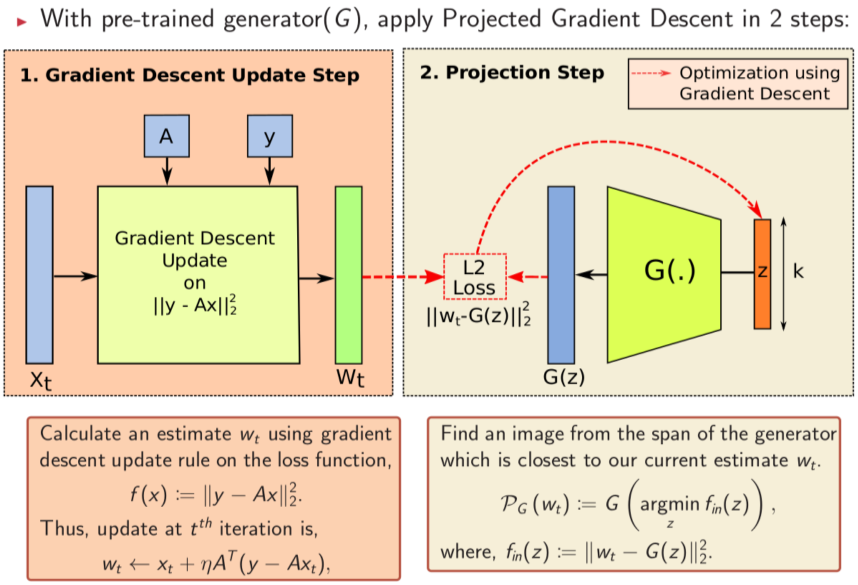


Figure 7: Illustration of generative model based approach

5.2. Generative Models with Projections:

In this case, instead of directly minimizing the Eqn. 5, we advocate a different approach of Projected Gradient Descent. We calculate the initial estimate using a gradient descent update, and project it back on the set S by finding the image closest to our estimate from the set S. This approach is depicted in Fig. 8 [5].



- With pre-trained generator(G), apply Projected Gradient Descent in 2 steps:

1. Gradient Descent Update Step
A
y
Gradient Descent Update on $||y - Ax||_2^2$
x_t
w_t
2. Projection Step
Optimization using Gradient Descent
L2 Loss
$||w_t - G(z)||_2^2$
G(z)
G(.)
z
k

Calculate an estimate w_t using gradient descent update rule on the loss function,

$$f(x) := \|y - Ax\|_2^2.$$

Thus, update at t^{th} iteration is,

$$w_t \leftarrow x_t + \eta A^T(y - Ax_t),$$

Find an image from the span of the generator which is closest to our current estimate w_t.

$$\mathcal{P}_G(w_t) := G\left(\underset{z}{\text{argmin}}\, f_{in}(z)\right),$$

where, $f_{in}(z) := \|w_t - G(z)\|_2^2$.

Figure 8: Generative model based approach with projections

5.3. Deep Generative Models with "Learned" Projections:

Instead of using hand-crafted signal prior such as sparsity in wavelet basis or gradients, learn the signal prior by using large dataset. Solve the same linear inverse problem using standard optimization procedures, except with an improved regularization learnt via deep projection models. The following are the steps involved in the algorithm:

1. Estimate "best" signal prior I_x instead of hand-crafted signal prior:
    1. Train classifier D whose classification cost function approximates I_x .
    2. Learn projection function P that maps signal v to set defined by I_x .

2. Use ADMM to solve the inverse linear mapping problem of the form:

    $$y = Ax + e$$

    1. Objective function + updates: (paste)

    2. Proximal operation -- plug in what is learnt using Step 1.

The key difference we have in this approach is, instead of obtaining the projection through Euclidean distance minimization, we directly learn the projections from outer space to set S. Learning is achieved with **Denoising Encoder (E).** We learn the boundary of set S by employing a Discriminator (D) training adversarially with the Encoder (E). To improve performance, we also use another Discriminator for latent space (D_l).

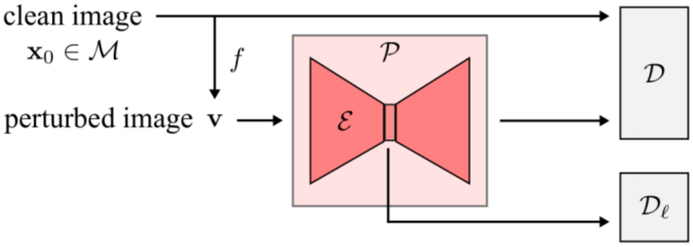


Figure 9: Training of the projection network P

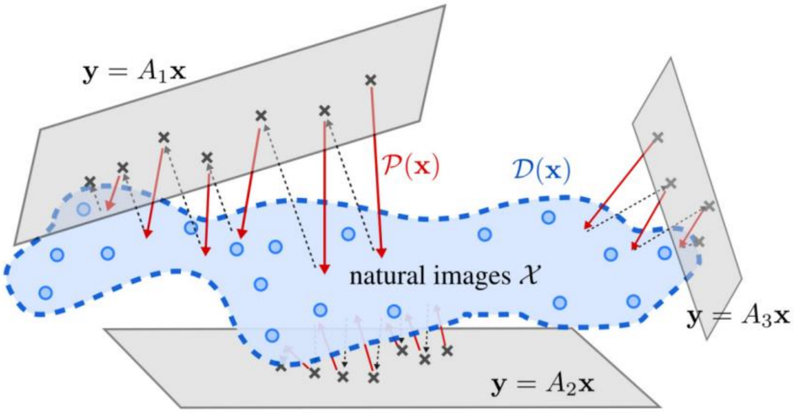

$\mathbf{y} = A_1\mathbf{x}$
$\mathcal{P}(\mathbf{x})$
$\mathcal{D}(\mathbf{x})$
natural images $\mathcal{X}$
$\mathbf{y} = A_3\mathbf{x}$
$\mathbf{y} = A_2\mathbf{x}$

Figure 10: Proposed framework

Here, we can see that once the projections are learnt, it can be used for solving any linear inverse problem irrespective of the fact that the linear operator A maybe different in each case.

5.3.1. Network architecture:

$$\begin{array}{lcccc}
Input & \rightarrow & conv(4,64,1)_{ve} & \rightarrow & conv(4,128,1)_{ve} \\
 & \rightarrow & conv(4,256,2)_{ve} & \rightarrow & conv(4,512,2)_{ve} \\
 & \rightarrow & conv(4,1024,2)_{ve} & \rightarrow & cfc \\
 & \rightarrow & conv(2,1024,1)_{ve} & & (latent) \\
latent & \rightarrow & dconv(4,512,2)_{ve} & \rightarrow & dconv(4,256,2)_{ve} \\
 & \rightarrow & dconv(4,128,1)_{ve} & \rightarrow & dconv(4,64,1)_{ve} \\
 & \rightarrow & dconv(4,3,1) & & (Output)
\end{array}$$

Figure 11: Encoder and decoder layers in architecture

Here,

conv(w,f,c) → convolutional layer with w × w window size, ‘f’ filters and ‘c’ channels

v,e → refers to batch normalization and elu activation

bottleneck(half/same/quarter) → bottleneck unit, keeping the dimensions same or reduce it by half, or quarter

$$\begin{array}{lcccc}
Input & \rightarrow & conv(4,64,1) & & \\
 & \rightarrow & bottleneck(half) & \rightarrow & \{bottleneck(same)\}_{\times 3} \\
 & \rightarrow & bottleneck(half) & \rightarrow & \{bottleneck(same)\}_{\times 4} \\
 & \rightarrow & bottleneck(half) & \rightarrow & \{bottleneck(same)\}_{\times 6} \\
 & \rightarrow & bottleneck(half) & \rightarrow & \{bottleneck(same)\}_{\times 3} \\
 & \rightarrow & vbn\ \&\ elu & \rightarrow & fc(1)\ (output), \\
Input & \rightarrow & bottleneck(same)_{\times 3} & & \\
 & \rightarrow & bottleneck(quarter) & & \\
 & \rightarrow & \{bottleneck(same)\}_{\times 2} & & \\
 & \rightarrow & vbn\ \&\ elu & & \\
 & \rightarrow & fc(1)\ (output) & &
\end{array}$$

Figure 12: Layers in discriminator: Image space & Latent space

6. Results:

Deep generative models were used to inpaint the images in celebrity dataset. Following are the results:

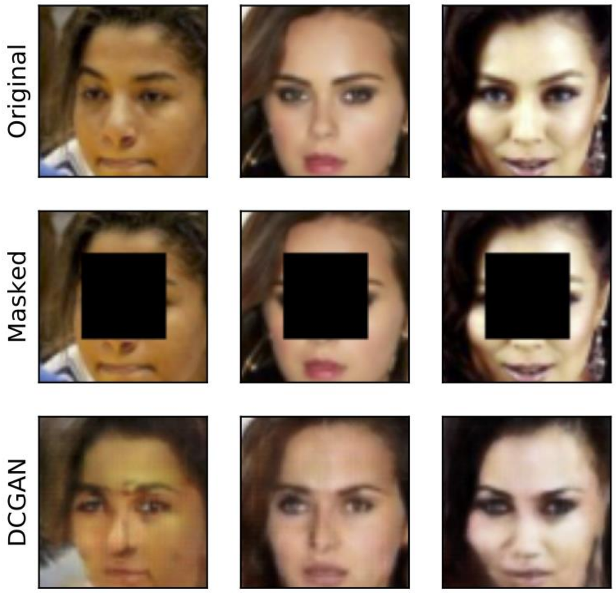


Figure 13: Celebrity images inpainted using deep generative models

Following are the results of images inpainted using deep generative models with learned projections:

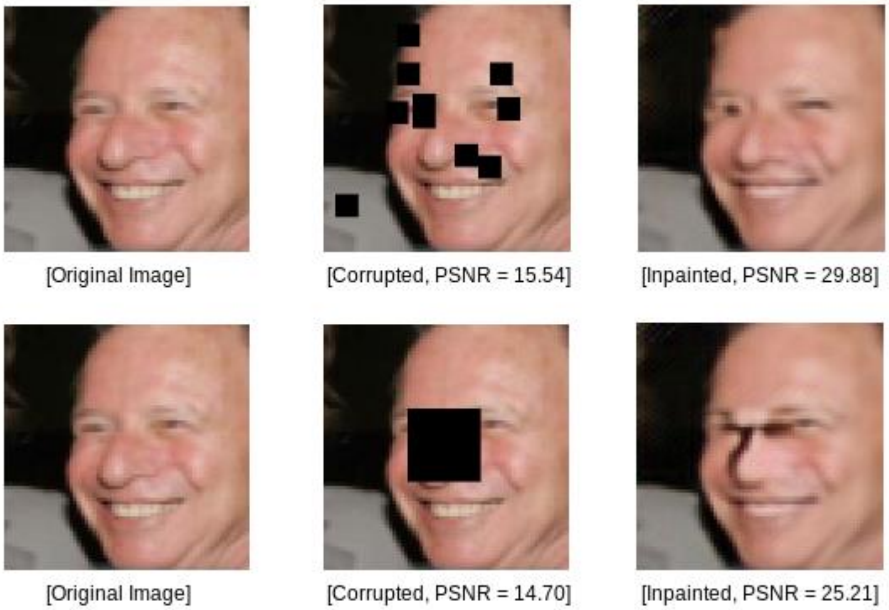

[Original Image]
[Corrupted, PSNR = 15.54]
[Inpainted, PSNR = 29.88]
[Original Image]
[Corrupted, PSNR = 14.70]
[Inpainted, PSNR = 25.21]

Figure 14: Results of inpainted celebrity images

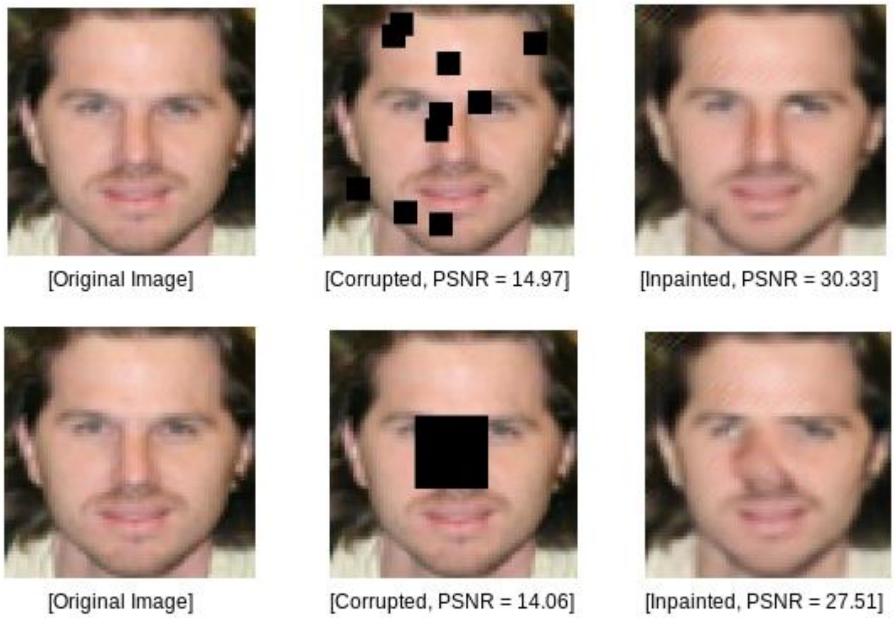

[Original Image]
[Corrupted, PSNR = 14.97]
[Inpainted, PSNR = 30.33]
[Original Image]
[Corrupted, PSNR = 14.06]
[Inpainted, PSNR = 27.51]

Figure 15: Results of inpainted celebrity images

The PSNR value for the inpainted image is more than that of the corrupted image which indicates the success of the algorithm.

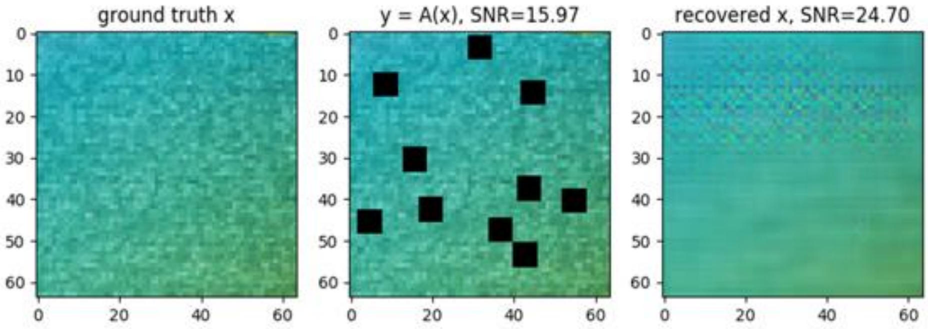


Figure 16: Results of inpainted plate images (artificially corrupted)

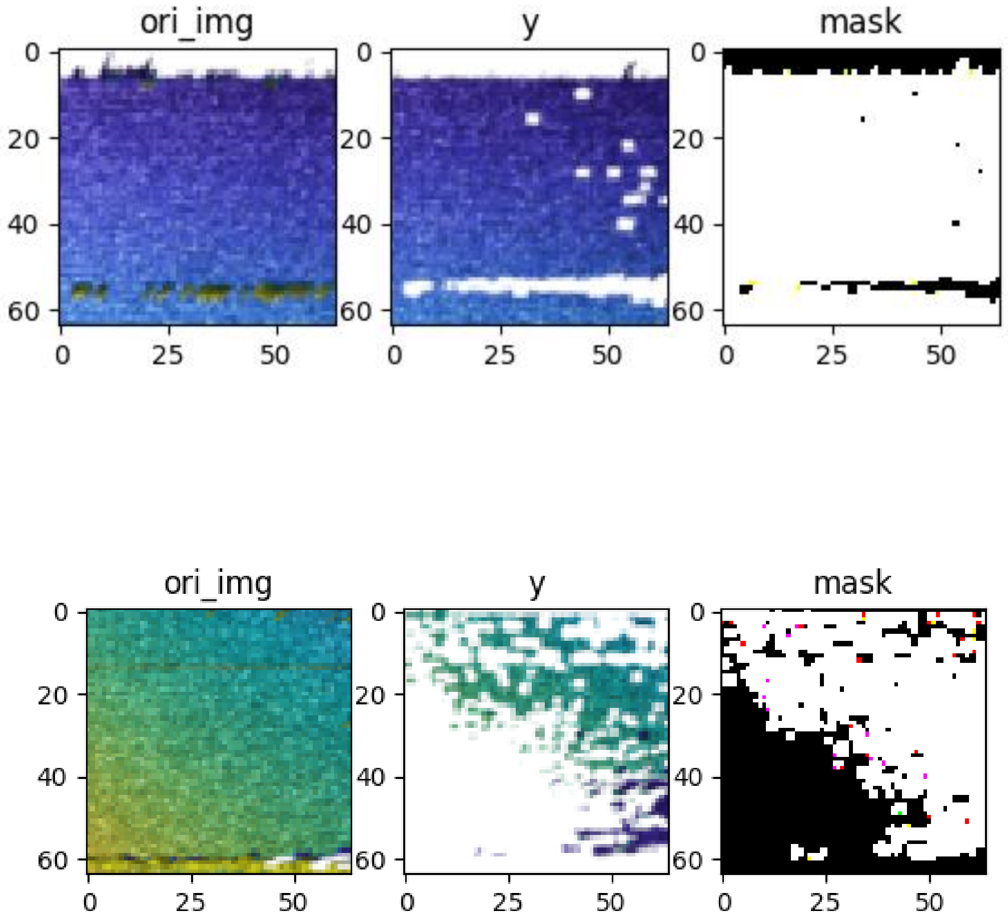


Figure 18: Results of inpainted plate images (by creating a mask for the saturated pixels)

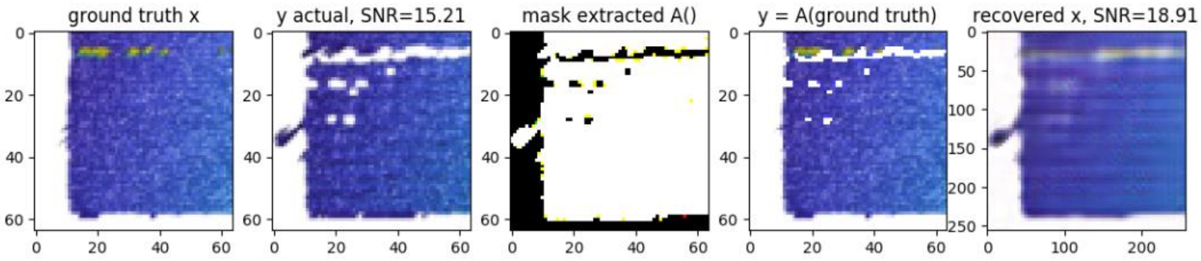


Figure 19: Results of image inpainting

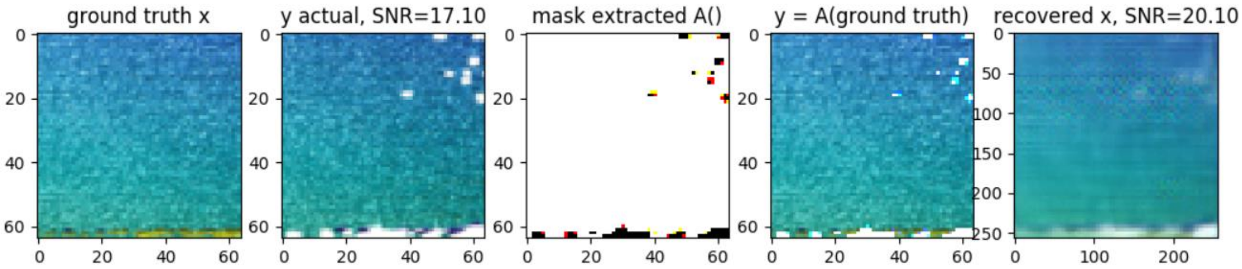


Figure 20: Results of image inpainting

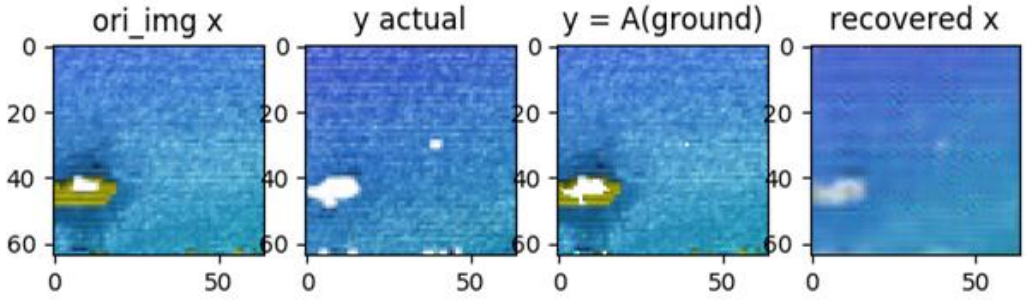


Figure 21: Results of inpainted plate images

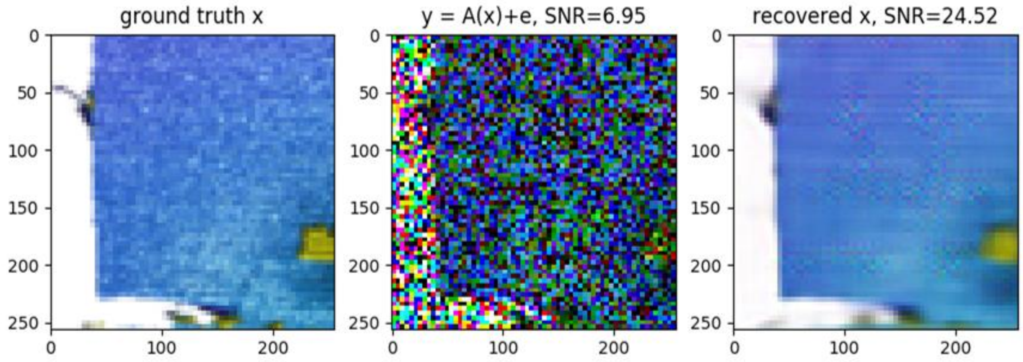
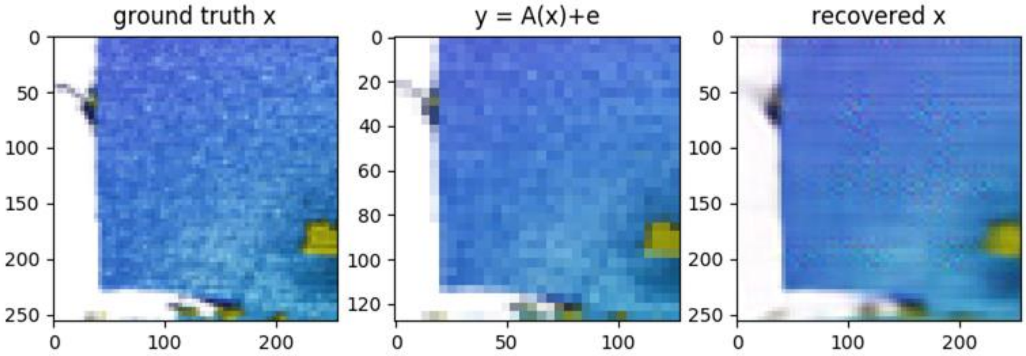


Figure 22: Results of inpainted plate images (by adding noise artificially)



256 x 256 128 x 128 256 x 256

Figure 23: Results of inpainted plate images (super resolution)

7. Conclusion and clarification:

In this project, we implemented and applied 2 different Deep learnt prior based methods to solve image inpainting problem for CelebA dataset. We are able to reproduce the state-of-the-art results for CelebA dataset.

Apart from that, we also employed learnt projections based method for inpainting on the Metal plate dataset obtained through Structured Light system (SLS) technique. In the case of Metal plate dataset, we solved variety of inverse problems such as denoising and super-resolution. Though the results are not perfect because of limitations in available training data, computation and parameters tuning, we depicted significant evidence on the correctness of our model.

With respect to the question asked during the project presentation, I would like to clarify that the Encoder we are using for learning the projections, is a **Denoising Encoder**. It is not a Variational Auto Encoder as we stated in the presentation mistakenly.

References:

1. J. H. Rick Chang, Chun-Liang Li, Barnab́as Poczos, B. V. K. Vijaya Kumar, and Aswin C. Sankaranarayanan, “One Network to Solve Them All — Solving Linear Inverse Problems using Deep Projection Models”. Link: https://arxiv.org/pdf/1703.09912.pdf
2. Vignesh Suresh, Yajun Wang, Beiwen Li, “High-dynamic-range 3D shape measurement utilizing the transitioning state of digital micromirror device”. Link: https://www.sciencedirect.com/science/article/pii/S0143816618301210
3. Celeb A. Link: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html / MS Celeb 1M. Link: https://www.msceleb.org/
4. Ashish Bora, Ajil Jalal, Eric Price, Alexandros G. Dimakis, “Compressed Sensing using Generative Models”. Link: https://arxiv.org/abs/1703.03208
5. Viraj Shah, Chinmay Hegde, “Solving Linear Inverse Problems using GAN priors: an Algorithm with provable guarantees”.