ENEE631- DIGITAL IMAGE AND VIDEO PROCESSING

IMAGE INPAINTING

Anirudh Nakra
UID: 118134444

Investigating Image Inpainting

Anirudh Nakra
Department of Electrical and Computer Engineering,
University of Maryland
College Park, USA
anakra@umd.edu

***Abstract*—In this project I implement a background extraction and object removal mechanism in a given scene through the technique called Image Inpainting. Image Inpainting has been a task for decades in the field of Image processing and Computer Vision. However with the advent of Deep Learning, the task has gone through a radical change in approach. Some of the earliest applications of Image Inpainting include watermark removal, removing a person from a scenery, Logo removal and so on and so forth. In this project, I implement a data preprocessing module, an object detection/masking system using Mask-RCNN and GUIs as well as an image inpainting module using various different methods to provide a comparative study between some major Image Processing techniques as well as SOTA Deep learning techniques.**

I. Introduction

Image inpainting refers to the task of filling in the unknown/masked out parts of an image. However the constraints are not so simple. The pixels must be filled in a way that the resultant image is photorealistic and accurately captures the scene. Image inpainting aims to solve the standard restoration and reconstruction problems we have for images. One of the most obvious applications is in the restoration of old manuscripts, most of which have some corruption over time and might even have parts torn off. This means that there needs to be incorporation of both semantics as well as aesthetics and traditional solutions such as style transfer or interpolation is not the right approach.

Traditionally, image processing methods have sought to draw inspirations from concrete mathematical models. One of the earliest successful inpainting model, the Naive-Stokes algorithm drew parallels between fluid dynamics and Stokes' equation and the image inpainting problem. The algorithm added edge continuity and color space contraints to help make the inpainted image look real. Later on Telea proposed a Fast Marching method which is similar to the Exemplar matching technique in that it does weighted average operations in patches along edges with continuous edge updation after inpainting.

Recently with the advent of deep learning, the image inpainting problem has been one of the problems that the different architectures and techniques are benchmarking themselves on. This is especially true for techniques that rely on GANs and deal with synthetic image synthesis. Modern DL based approaches have focused on encoder based representations. For instance, contextual encoders and CNN based topologies have found much success due to shift invariance and other important properties. There has also been work using Convolutional Autoencoders and extensions with partial convolutions. However over the past 5 years, some techniques have stood out due to the uniqueness in their approach. These are deep image priors, the OpenAI technique GLIDE and Fourier Convolution based structures. These techniques are the focus of the inpainting problems studied in this paper and will be discussed in depth in the subsequent sections. The `colab` notebook along with the plots can be accessed at the Drive link `here`.

II. Data Preprocessing

The datasets I use are quite wide varying. I play around with both some state of the art and widely adopted datasets such as ImageNet and COCO while also looking at a self curated set of images. Lets talk about the self curated dataset. Conventionally, image inpainting has been used for solving a multitude of important image processing problems. To compare the different algorithms effectively, I sought to make a small set of 5 images that I would check the inpainting on just to analyse how they perform in different categories. The categories I chose were:

- *Numerous Small Objects:* I select a scenery of several flowers with a sunset based background and test the inpainting after the removal of a single flower
- *One Main Object:* I select a scenery of a bird flying with it's wings spread in the blue sky.
- *Watermark Removal:* I select the scene of a protest with several posters to use for text removal and inapinting based problems similar to a traditional watermarking case.
- *Texture inpainting:* I select a color gradient image with the intention of checking how well the systems can recreate high level patterns.
- *Classical Primary Secondary task:* I select an airplane image with two main objects in the scene to recreate a primary-secondary object scene

Other than this set of curated images aligned with specific tasks, I use the standardised ImageNet and COCO dataset due to the high number of samples in those datasets for training some of the more involved NN based techniques such as

Mask-RCNN which is extremely data hungry.

For the bare bones object extraction approach, I use a couple of methods. The first method involves using the scores and the bounding boxes intrinsic to the dataset itself. The COCO dataset comes with utilities through the library `pycocotools` that allow me to manipulate the bounding boxes and scores for the annotated mask directly. From thereon forth, the task is relatively straight forward. I change the mask from boolean to binary using standard looping and convert it into an inverse mask which is really what we need since we want to the image other than the object to inpaint and not the scenery itself. In the bare bones extraction approach, I also create a GUI in a MATLAB script and extract the masks to my Colab Notebook to allow testing more versatility in the masks created.

The more involved object extraction approach is formulated using Mask-RCNN, a natural progression to the now relatively extinct Fast RCNN and YOLO systems due to its ability to provide pixel wise classification masks and with the advent of semantic segmentation tasks. This technique has been trained on ImageNet but it is straight forward to use it for the COCO dataset as well.

III. Object Detection/Extraction

The first involved part of the project is to create an object extraction technique that can allow us to identify objects of interest in a given object, perform *pixel-wise* semantic segmentation and then generate a mask using it. This pixel wise property is of importance due to the fact that we want masks that adapt closely to the shape of an object and are not the traditional rectangular anchors we encounter in the YOLO and RCNN techniques. Inpainting into square boxes has a downside of it leaving a lot of information as well as the fact that sometimes with bigger objects, the scene extracted after object detection has a lot of blank space which needs to be filled leading to issues related to computational cost.

To talk some more about the technique used here, Mask RCNN has become the SOTA technique for semantic segmentation task, most of which involve detecting/assigning scores and probabilities of an object existing at each pixel of an image. This is in contrast to earlier object detection techniques such as YOLO (You Only Look Once) which provide anchor boxes around the detected object but are developed for real time problems. Mask R-CNN extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. The Faster RCNN backbone of the Mask-RCNN involves using CNNs to extract feature vectors from the images and then use an RPN (Region Proposal Network) which is the object prediction aspect of the system. It then applies an ROI pooling based on the possible anchors to scale them to a similar degree and pass them through a fully connected layer acting as a linear classifier to output the bounding boxes. Some of the practical drawbacks I noticed was the fact that this semantic segmentation while being very impressive, is not fully accurate. It does track the shape of the object extremely well and does give a polygonal fit to the object but the masks often leave the edge portions of the object detected intact, which causes artifacts in the inpainted images. This will become more clear when we talk about image inpainting using Deep Image Priors after selecting large objects.

IV. Benchmarking Techniques

First, I implement **Exemplar Matching**. This technique is probably the least involved of all the inpainting techniques I apply. This however is the reason that this technique was chosen. It is not chosen to display the power of an inpainting algorithm but rather to use as a benchmarking tool (along with Naive Stokes and TELEA inpainting methods).

Unlike the more learning based techniques I will talk about later, this Exemplar Matching technique I implement has much more fundamental origins. The authors model the image inpainting problem using what they call an "isophote driven sampling approach" with previously proved results on the adaptability of exemplar based methods for texture synthesis atleast for linear structures inside the image itself. At a higher level the algorithm for exemplar matching is composed of a recursion of 8 major steps. The firs couple of steps deal with the extraction of the mask and therefore the ROI that needs to be filled. The algorithm then segments the target (to be inpainted) and the source region and makes patches along the boundary of the target region. Using either a gradient or a tensor based approach, the priorities of the patch are calculated according to the given formulae. The maximum priority patch is then selected and a similarity metric is then computed with the background to recreate what the best matching patch in source area would be. This is then repeated for all patches on the boundary until the ROI is inpainted completely.

$$P(p) = C(p)D(p)$$

where $C(p) =$ is the confidence term and $D(p)$ is the data centric term.
The dataset which Exemplar Matching is tested on is the curated dataset I talked about earlier. The ROI and masks are created using the GUI approach with user flexibility being a major reason to select this method. Other than exemplar matching, I also use the inbuilt **Naive Stokes** based inpainting and the **TELEA method** using the openCV library. These are also approached from the perspective of traditional image processing and statistical distribution based models to benchmark the NN techniques.

V. Deep Image Priors

The **Deep Image Prior** paper aims to show that a randomly initiliased neural network can be used to create untrained priors and these priors can help make trainable

priors redundant computationally. Specifically, the authors adopt the use of an untrained convolutional neural network and postulate that a simple CNN with randomly initialised parameters is enough to model the corrupted image along with some changes in the architecture of the CNN.

In this way, they theorise the problems of image reconstruction specifically denoising, superresolution, inpainting and others to be a form of a conditional image generation task. They specifically replace the prior term $R(x)$ in the conventional energy minimisation optimisation with the prior of a CNN which they postulate models the inductive bias that trained CNNs learn. Optimising this θ using gradient descent leads to a minimiser that they use to model latent distribution of the image generation task. The paper interestingly also shows why the proposed architectural techniques work emprirically. Specifically, they show that the solution space searched by the proposed models favours natural images. It is also found that natural images converge faster and the number of iterations needed to make noisier images increases with an increase in the noise level.

In my implementation, I use both the barebones masking using the COCO dataset as well as the Mask-RCNN objects to create an inpainted model using DIPs. An important aspect to note is the fact that **DIPs are completely untrained**. In fact they only need the masked image and the mask itself to recreate the scene. In this way, they are not really competing with GANs trained on millions of instances but rather the more traditional methods such as Nearest Neighbour and other interpolation based approaches. However this is not to say that DIP is a weak technique. In fact, it has been empirically shown that DIp results in inpainted images that are much closer to the GAN images that any other untrained technique. Thus DIPs lead to significant improvements in untrained methodologies. Note that the DIPs are still trained but wrt θ not the training samples therefore the usage of the word untrained.

VI. LAMA Fourier Convolutions

The second major inpainting technique I implement is called **Large Mask inpainting with Fourier Convolutions**. The adaptability of the inpainting technique to large masks is why it is called "Large Mask" inpainting. The need for this technique arose out of quite a practical problem. Images nowadays have very high resolution and therefore are very detailed. Through the course of implementing Deep Image Prior method, I found that the original images in the curated dataset (some of which are 4K!) take quite a long time to train and need to be rescaled before the technique can be performed. Motivated by this issue, I wanted to look for a more involved SOTA technique that can handle these issues. With the advent of high bit rates, high resolution images have become a reality and this is an important case I need to accomodate for.

As rightly quoted in the paper, "Modern image inpainting systems, despite the significant progress, often **struggle with large missing areas**, **complex geometric structures**, and **high-resolution images**". The actual logic behind the LAMA-Fourier technique is not very obtuse. The authors postulate that for learning the inpainting area in the case when the masks are large sized, there is a need for incorporating more global contexts. Therefore, they adopt the usage of a spectral transform based method. Specifically, they use something called an **FFC or Fast Fourier Convolutional Layer** that splits the convolution pipeline into two branches. Specifically, they argue for the usage of the conventional spatial convolutions in the local branch to capture local dependencies and a more global FFT transform for the global branch. Emperically, they find that this reduces the decaying effective receptive field effect often found in ResNet based architectures for inpainting.

VII. DALL-E based Inpainting

The last and perhaps the most onvolved implementation is that of a DALL-E based Image Inpainting system. DALL-E is a system created by OpenAI that generated photorealistic image based on an input text. This is done using the technique OpenAI coins "Guided Language to Image Diffusion for Generation and Editing" or GLIDE for short. The motivation for this problem is to extend the traditional image based questions to multiple modalities. Incorporating text and Image generation into the same system has thus been called one of the most creative software developed in our lifetime.

The GLIDE technique takes inspiration from both text-conditional image models as well as unconditional image models. Through prior work, it has been found that text conditional image techniques are still developing and are unable to generate photorealistic models. On the other hand, there are many SOTA techniques for unconditional image synthesis such as SinGANS, etc. Among these unconditional models however, the category of models that prevails are called Diffusion Models. Diffusion GANs have really come up to be one of the strongest tools in image synthesis. They work by corrupting the training data by progressively adding Gaussian noise. This removes details in the data till it becomes pure noise. Then, it trains a neural network to reverse the corruption process. Running this reversed corruption process synthesises data from pure noise by gradually denoising it until a clean sample is produced. These diffusion models when incorporated with *classifier guidance* (conditioning on classifier labels) allow the GANs to reach photorealism. In contrast, there has also been work on *classifier guidance free diffusion models* through interpolation between predictions. The GLIDE model is trained using a classifier guidance system with a text encoder submodule to condition on the text. In this way, it achieves an integration of the modalities required.

VIII. Experimental Observations

Experimentally, it is obvious that deep learning techniques are very powerful for image inpainting. In cases of watermark removal, traditional techniques such as Exemplar Matching and Naive-Stokes are not able to clear the words but DL techniques such as DIP and LAMA perform satisfactorily. It is however to be noted that on pictures with repeated linear patterns and symmetry, the classical Image processing techniques do perform closer to SOTA methods. Their limitations however are clear as day. For large 3D objects, the classical techniques fail. It is however surprising that even DIP fails in this area and this motivated the need for LAMA and GLIDE to a certain extent.

It is also important to notice the efficacy of the object extraction and masking systems. In some of the figures shown, Mask RCNN is able to isolate upto 6 different objects from the same scene leading to robust solutions to problems. It is breathtaking that it is even able to detect the person in a painting and perform semantic segmentation.

GLIDE is perhaps the most interesting and creative technique implemented here. The fact that it can condition the masked part of the image on a text through NLP based encoding is miraculous. In cases where the object is present partially (the bird case) it is able to interpolate very well and reconstruct a photorealistic image from the part of the wings of the bird. Furthermore, it is able to take in the sky from that image and create an image of an ocean which I found very very interesting.

IX. Evaluating Performance

It is important to note the key learning of this project. Therefore a comparison/tabulation of the strong points as well as the skills learnt is necessary.

1) VERSATILITY : Through the choice of a curated dataset corresponding to different samples, I was able to analyse the effectiveness of SOTA techniques on different image inpainting problems.

2) COMPLETENESS : I implemented an object extration module that gave this project a completion aspect. This can now be used as an end to end system where the user inputs the image that requires an object to be inpainted into along with the choice of the object.

3) CLASSICAL v/s DEEP LEARNING : The comparison of classical and deep learning techniques give an important insight on where the traditional methods fail along with applications where there is no necessity of applying the more involved computationally expensive techniques arising from DL. This insight is important for an end to end system since real time considerations need to take place.

Fig. 1. Sample COCO data sample

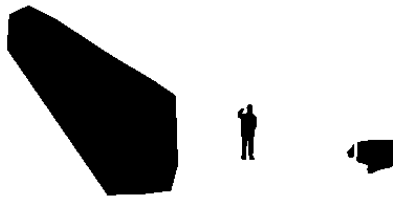

Fig. 2. COCO sample's binary mask

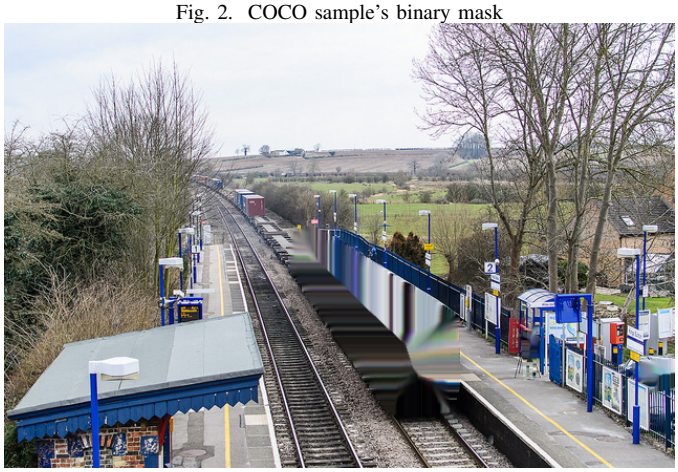

Fig. 3. COCO train with Naive Stokes inpainting

X. Key Conclusions

I would like to conclude by stressing upon a couple of things.

- MOTIVATIONS : The motivation for the project was to look at image inpainting qualitatively. Image inpainting

Fig. 4. COCO train with Telea's inpainting

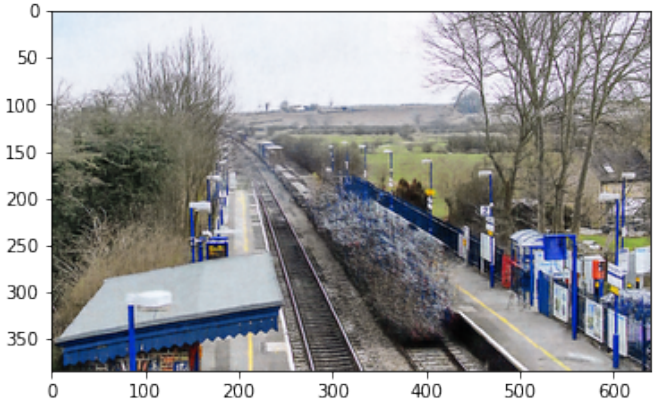

Fig. 5. COCO train with DIP inpainting

is an important benchmarking problem which many image synthesis and augmentation architectures use for evaluation purposes. This field is growing at such a fast pace that it regularly has a good chunk of papers at the premier computer vision conferences such as CVPR, ECCV and BMCV. Through analysis of the field I aim to introduce the reader to the past, present and the future of this important problem.

- THINGS LEARNT : This project allowed me to learn the important concepts of *Image restoration* and *Image reconstruction* along with *Object detection and masking* problems. I was also able to work with *manifold learning* and *texture synthesis* issue through more rigorous image processing techniques relying on geometrical ideas.

- FUTURE SCOPE : I think the future is in the deep learning techniques. Although image processing techniques do well for single object images and images with more linear background interpolation requirements, they fail in complicated setups. This is where the ability to sample from the true underlying distribution through GANs and other generative model really shines.

Fig. 6. COCO bus detected using Mask-RCNN

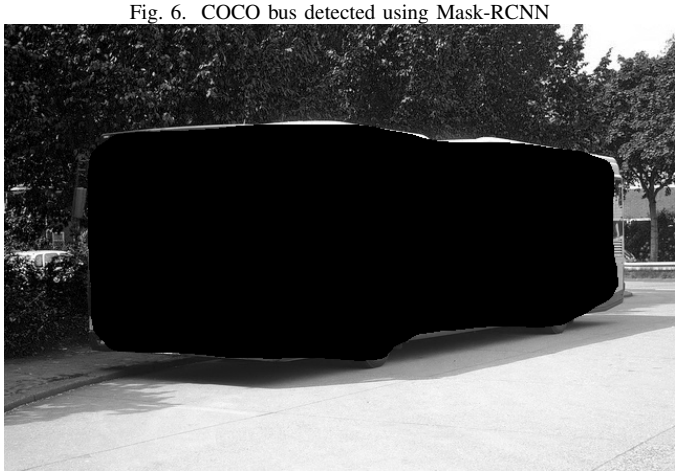

Fig. 7. Negative masking using Mask-RCNN

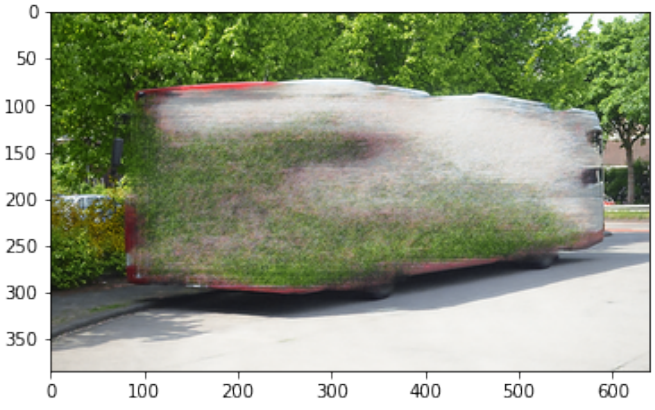

Fig. 8. DIP's poor performance on large objects

With DALL-E, I think the image inpainting world has been revolutionised and the future looks promising. It remains to see what new architectures will be developed and whether they would be able to keep up with a human's ability to speculate about the inpainted area.

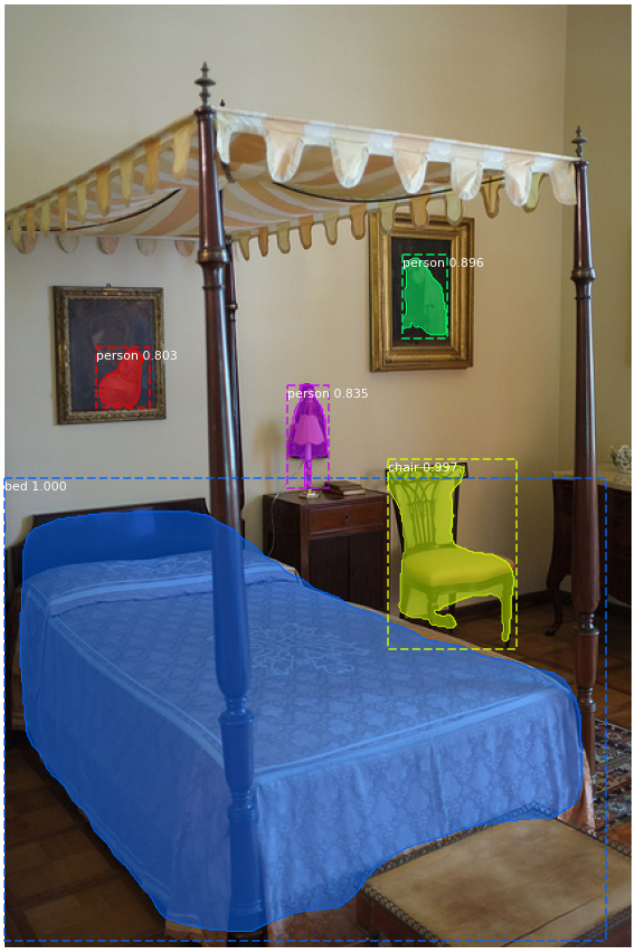

Fig. 9. Mask-RCNN isolating small objects

Fig. 10. Masking the person inside painting

Fig. 11. DIP's good performance on small objects

Fig. 12. Mask-RCNN on lamp

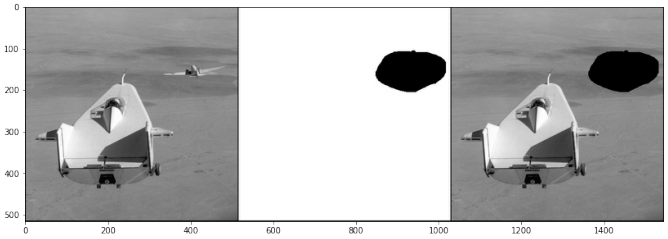

Fig. 13. Airplane and masked Airplane from curated dataset

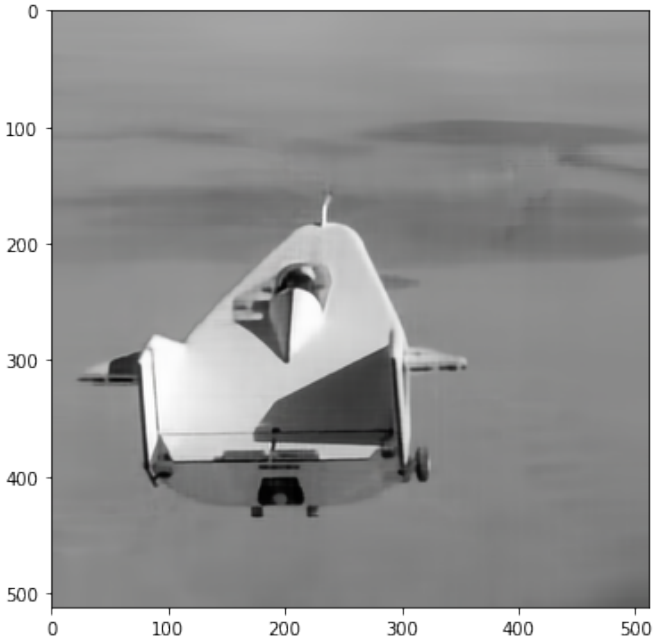

Fig. 14. DIP on Airplane

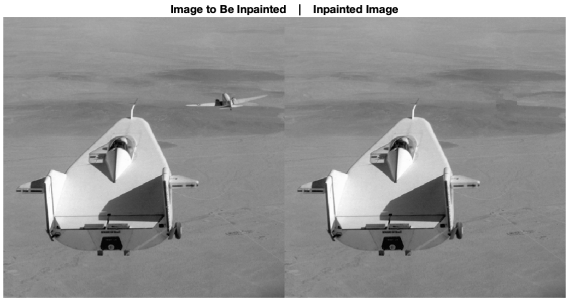

Fig. 15. Exemplar Matching on Airplane

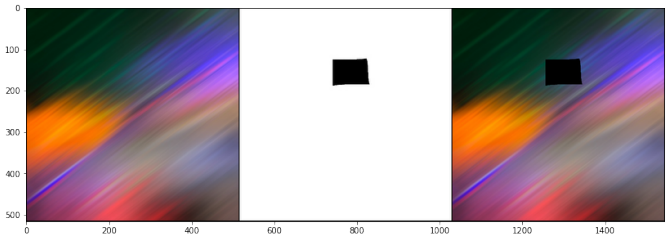

Fig. 16. Color gradient and its mask from curated dataset

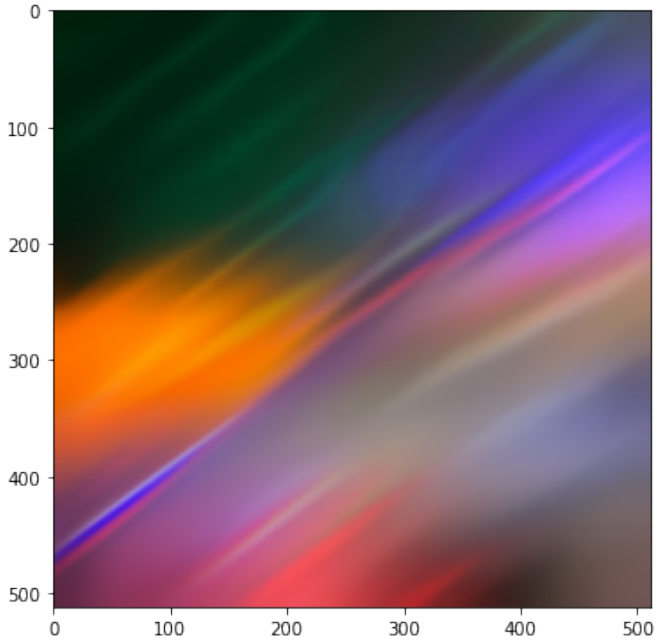

Fig. 17. DIP on color gradient

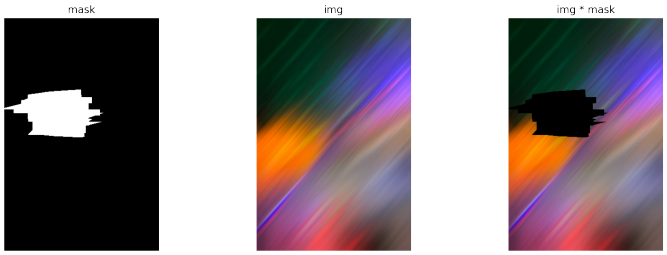

Fig. 18. LAMA mask on color gradient

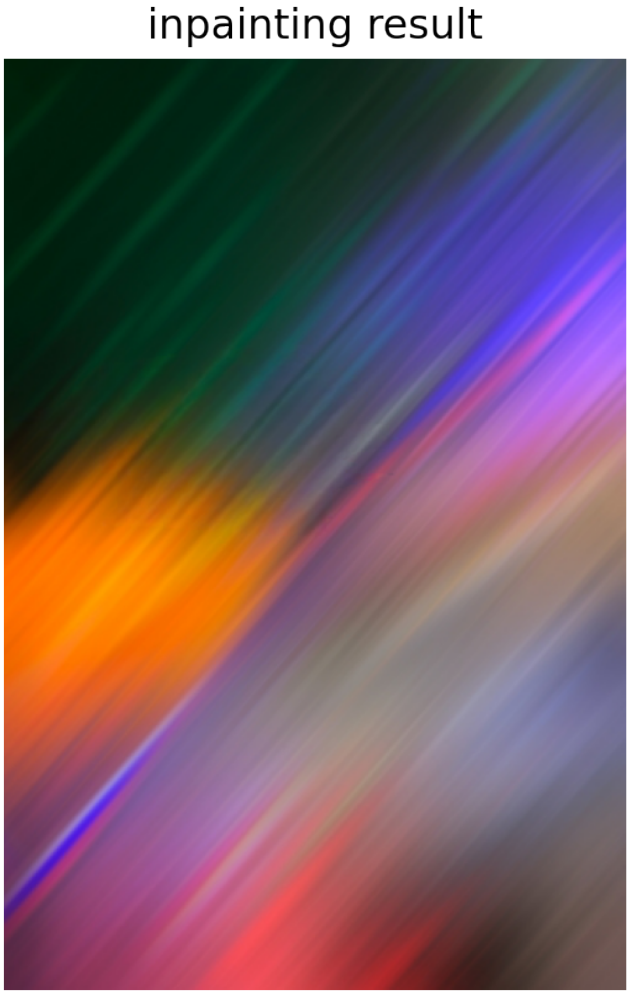

Fig. 19. LAMA inpainting on color gradient

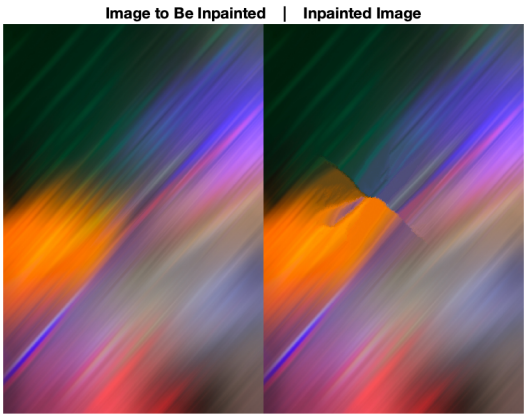

Fig. 20. Exemplar matching on color gradient

Fig. 21. GLIDE mask for Colors gradient

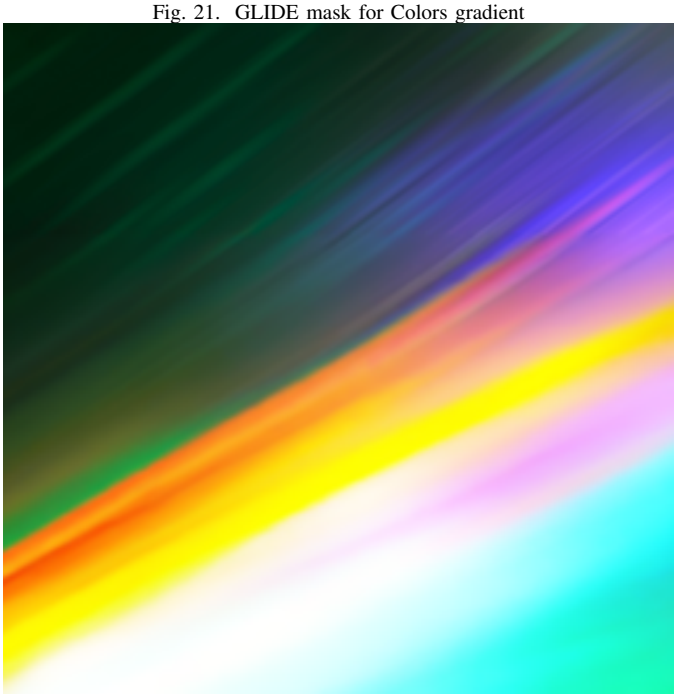

Fig. 22. GLIDE inpainting with prompt "**color gradient**"

Fig. 23. GLIDE inpainting with prompt "**northern lights**"

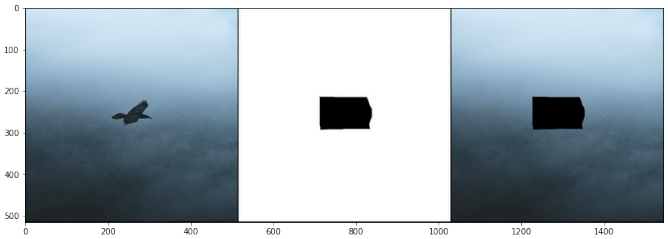

Fig. 24. Bird from curated dataset

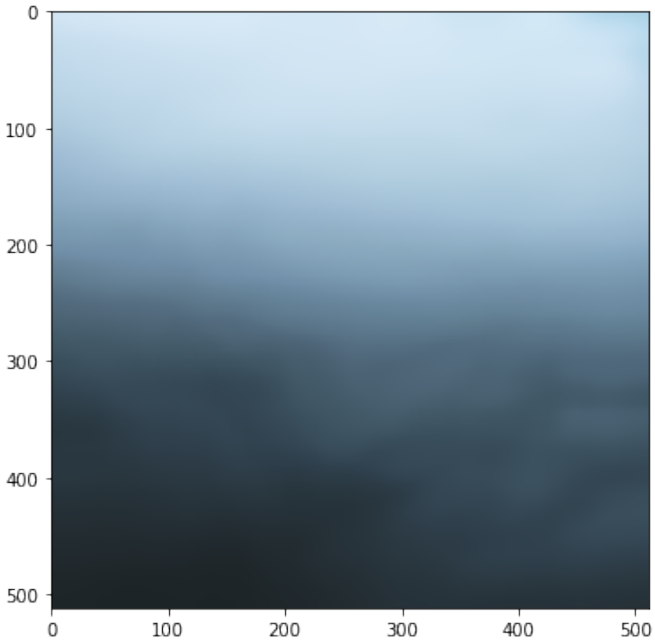

Fig. 25. DIP inpainting on Bird

Fig. 26. GLIDE mask on Bird for background interpolation

Fig. 27. GLIDE mask on Bird for bird reconstruction

Fig. 28. LAMA inpainting on Bird

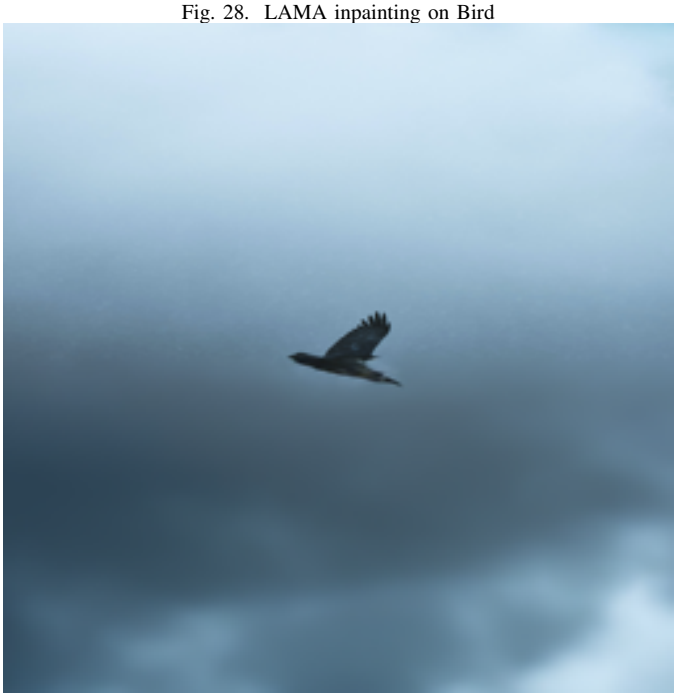

Fig. 29. GLIDE inpainting with prompt "**bird flying**"

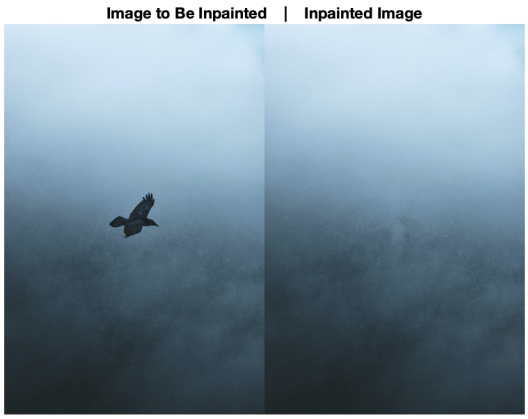

Fig. 30. Exemplar Matching on Bird

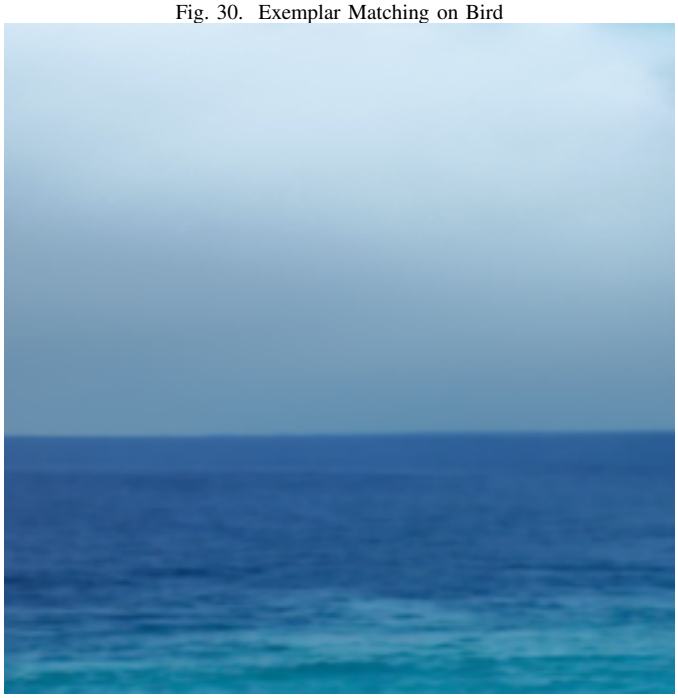

Fig. 31. GLIDE inpainting with prompt "**ocean**"

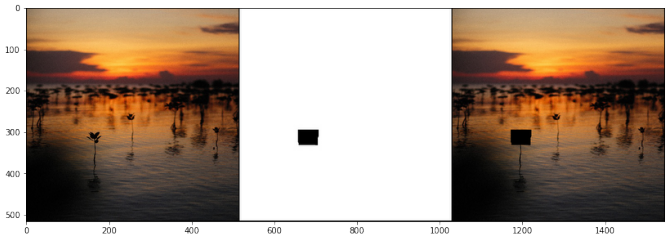

Fig. 32. Flowers from curated dataset and masks

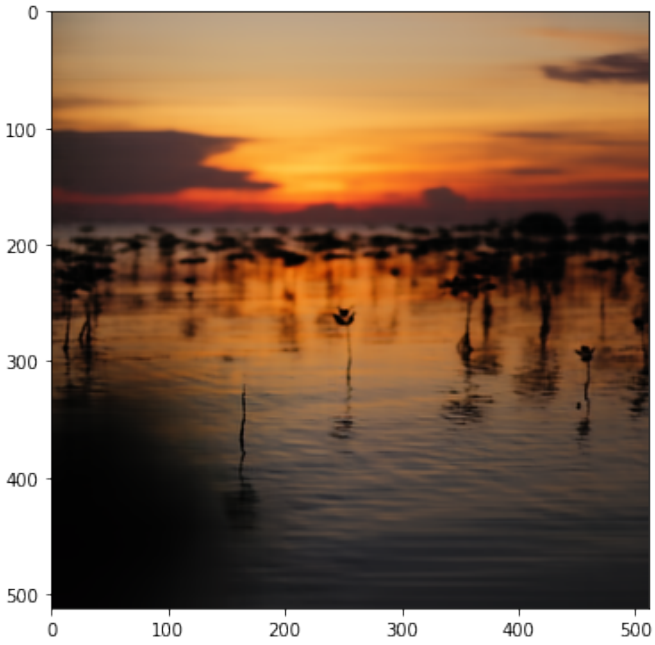

Fig. 33. DIP inpainting on Flowers

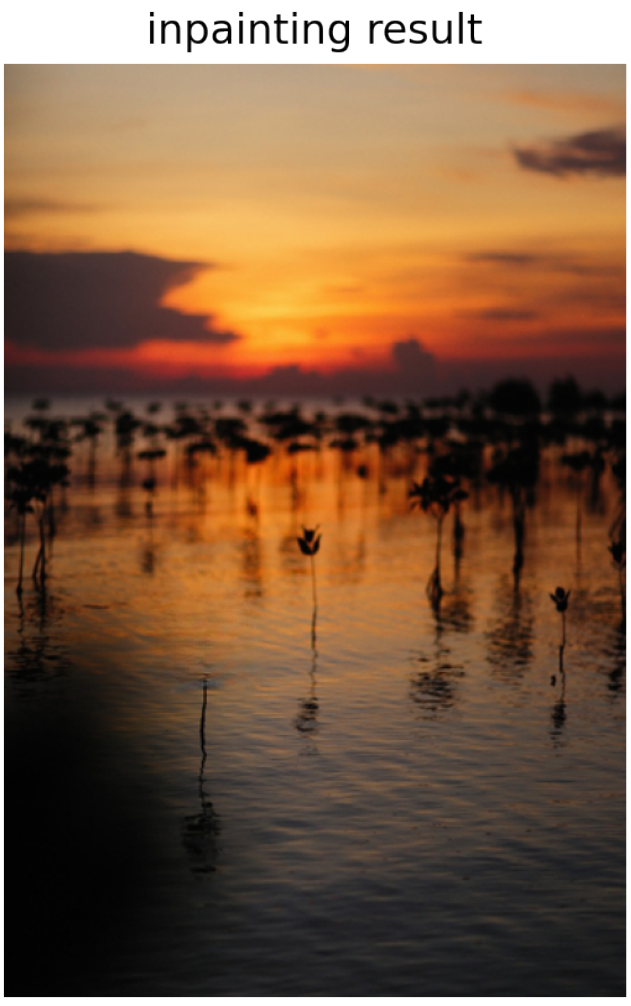

Fig. 34. LAMA inpainting on flowers

Fig. 35. GLIDE mask for Flowers

Fig. 36. GLIDE inpainting with prompt " **flowers in water**"

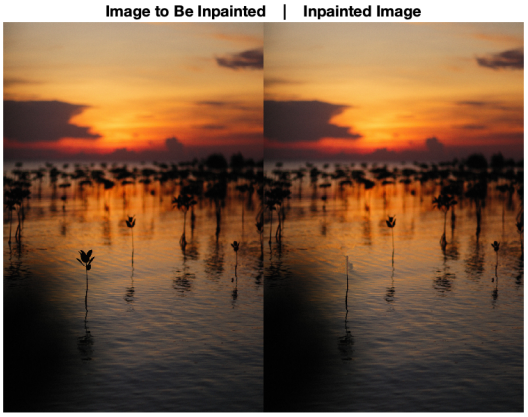

Fig. 37. Exemplar matching on Flowers

Fig. 38. GLIDE inpainting with prompt " **rainforest sunset**"

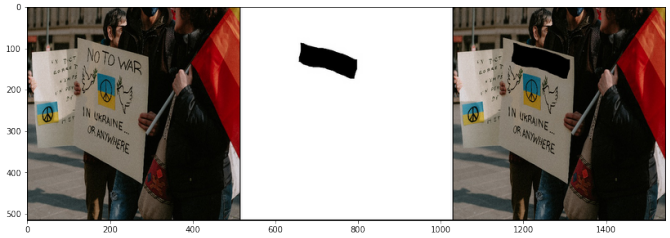

Fig. 39. Watermark based image and its mask

Fig. 40. LAMA mask for watermark inpainting result

Fig. 41. LAMA Inpainting on Watermark

Fig. 42. GLIDE Mask on Watermark

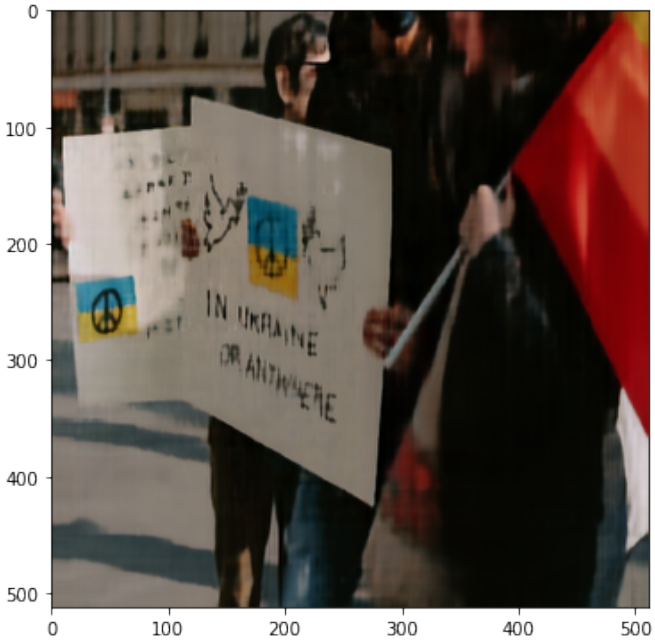

Fig. 43. DIP inpainting on Watermark

Fig. 44. GLIDE inpainting for watermark with prompt "**protest**"

Fig. 45. Exemplar matching on Watermark

REFERENCES

[1] A. Criminisi, P. Perez and K. Toyama, "Object removal by exemplar-based inpainting," 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings., 2003, pp. II-II, doi: 10.1109/CVPR.2003.1211538.

[2] R. Suvorov et al., "Resolution-robust Large Mask Inpainting with Fourier Convolutions," 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022, pp. 3172-3182, doi: 10.1109/WACV51458.2022.00323.

[3] Ulyanov, D., Vedaldi, A. Lempitsky, V. Deep Image Prior. Int J Comput Vis 128, 1867–1888 (2020).doi.org/10.1007/s11263-020-01303-4

[4] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., … Chen, M. (..). doi:10.48550/ARXIV.2112.10741

[5] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980-2988, doi: 10.1109/ICCV.2017.322.

[6] M. Bertalmio, A. L. Bertozzi and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990497.

[7] Telea, Alexandru. (2004). An Image Inpainting Technique Based on the Fast Marching Method. Journal of Graphics Tools. 9. 10.1080/10867651.2004.10487596.