INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2024IPPAT011

Thèse de doctorat

Internal methods for the generation and inpainting of images and videos

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Signal, Images, Automatique et robotique

Thèse présentée et soutenue à Palaiseau, le 25/03/2024, par

NICOLAS CHEREL

Composition du Jury :

Simon Masnou Professeur, Université Claude Bernard Lyon 1	Président/Examinateur
Julien Rabin Maître de conférences, ENSICAEN (GREYC)	Rapporteur
Jean-Michel Morel Chair Professor, City University of Hong Kong	Rapporteur
Julie Delon Professeure, Université Paris Cité (MAP5)	Examinateur
Coloma Ballester Professeure, Universitat Pompeu Febra	Examinateur
Patrick Pérez CEO, Kyutai	Examinateur
Yann Gousseau Professeur, Télécom Paris (LTCI)	Directeur de thèse
Alasdair Newson Maître de conférences, Sorbonne Université (ISIR)	Co-encadrant de thèse
Andrés Almansa Directeur de recherche, CNRS (MAP5)	Co-encadrant de thèse
Jean-Marc Thiery Senior Research Scientist, Adobe	Invité

Remerciements

Je tiens tout d'abord à remercier les membres du jury et particulièrement les rapporteurs. Merci à Julien Rabin pour ses nombreuses questions très pertinentes et son invitation au GREYC, Jean-Michel Morel pour sa mise en perspective vis-à-vis de l'état actuel du traitement d'images. Merci aux examinateurs et examinatrices, experts renommés en inpainting et traitement d'images, Julie Delon, Coloma Ballester, Simon Masnou et Patrick Pérez. Je tiens à remercier Jean-Marc Thiery pour sa participation au jury mais aussi pour son encadrement pendant mon stage à Adobe où j'ai eu l'occasion de travailler avec de fantastiques collègues sur un sujet complexe et passionnant.

Merci à mes directeurs de thèse; Yann, Andrés et Alasdair, vous avez su m'encadrer avec beaucoup de bienvaillance et me guider pour accomplir un très beau travail sur des problématiques où rien n'est évident, tout en me laissant une très grande liberté. Merci Andrés pour tes très nombreuses idées, Yann pour tes réponses à toutes mes questions, et Alasdair pour ces exaltantes sessions de travail et ta disponibilité. Merci surtout Alasdair pour cette thèse qui n'aurait pas eu lieu sans toi.

Je voudrais remercier tous mes collègues de l'équipe IMAGES de Télécom où j'ai passé de fantastiques années. Merci donc aux permanents pour leur proximité et les sympathiques moments passés ensemble le midi ou en cours : Arthur, Pietro, Loïc, Florence, Isabelle, Henri, Michel, Kiwon, Amal, Elsa. Merci spécialement aux plus assidus, même pendant les périodes de confinement, Christophe et Saïd.

Je remercie également les autres doctorants croisés pendant ces années de thèse: Raphaël A., Matthis, Emanuele, Alban, Arthur, Nicolas, Mateus, Giammarco, Chloé, Zoé, Rebeca, Inès, Robin, Marie, Antoine, Raphaël R., Emma, Rassim, Camille, Carlo, Edouard, Khalid, Ali & Ali, Emile, Xu, Elie, Arnaud, Cristiano. Mention spéciale aux mousquetaires Gwilherm, Erwan et Pierrick.

Merci aux ex-membres du BDS de Télécom : Florian, grâce à qui je me suis retrouvé à faire une thèse et courir en montagne pendant bien trop longtemps, Alexis, Bruno, Lucas et Benjamin. Merci aux amis de la HX4, Corentin pour ces nombreux weekends en Ardèche, Martin et Déborah pour tous ces verres dans notre quartier, Jean-Yves pour les aventures mémorables autour de Grenoble, Camille, Coline, Nicolaï, Antoine. Merci aussi aux grimpeurs de Centrale pour leur accueil parmi les leurs : Alexandre, Oksana et Lucie pour leur bonne humeur inconditionnelle et surtout dans la difficulté. Merci à Gaël pour ses visites régulières à Paris,

Sylvain, Damian, Alexandre.

Je vais conclure sur l'importance de la famille tout au long de ce chemin. Merci donc à ma soeur Marie et mes frères Loïc et Gaëtan pour ces moments passés ensemble pendant ces dernières années, en plus de toutes les précédentes. Merci finalement à mes parents.

Résumé en français

Les images sont des moyens de communication et d'expression centraux dans nos sociétés et nos vies. Si nous sommes aujourd'hui capables de capter ces images sous formes numériques et de leur appliquer de nombreux traitements (équilibrage, édition, retouche, etc), il n'existe pas de façon simple de les manipuler amplement tout en préservant leur aspect réaliste. La représentation sous forme de pixels est pratique mais ne reflète pas la réalité de leur complexité et l'interdépendance cachée qui existe entre tous les différents pixels d'une même image. Ce phénomène est exacerbé dans le cas des vidéos où les modifications doivent être précises pour être convaincantes. Les applications sont néanmoins nombreuses pour l'art, le graphisme, ou le cinéma.

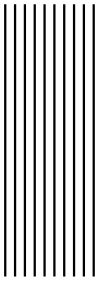
Nous nous intéressons dans cette thèse aux problèmes de la génération d'images et d'*inpainting*, ou complétion d'image. Ces problématiques sont proches, et l'on peut tout à fait voir l'*inpainting* comme une génération contrainte. Les derniers progrès en traitement d'image ont montré des résultats de plus en plus époustouflants pour l'édition et la génération (Ho et al. 2020; Saharia et al. 2022b). Ces méthodes récentes s'appuient sur des modèles complexes, de grands réseaux de neurones, et de larges bases de données d'images. L'apprentissage statistique combine ces éléments et permet de tirer parti de la large diversité d'images pour mieux les générer.

Notre objectif à terme est le traitement des vidéos mais les méthodes récentes sont déjà à la limite des contraintes matérielles pour le simple traitement des images. Nous cherchons donc ici à résoudre les problèmes de génération et d'inpainting de façon plus efficace: avec des petits modèles et très peu d'exemples d'apprentissage. Nous étudions en particulier les méthodes par patchs, les méthodes d'apprentissage *interne* sur une image, et les couches d'attention qui font le lien entre les entre patchs et réseaux de neurones.

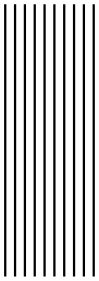
Le Chapitre III présente une application sans apprentissage à la synthèse mono-image par réarrangement de patchs. Alors que Shaham et al. (2019) choisissent d'entraîner un réseau de neurones, nous montrons qu'il est possible d'obtenir des résultats d'une qualité supérieure sans apprentissage avec une approche utilisant explicitement l'information dans l'image de référence. Nous proposons deux version de notre méthode, une version qui favorise la diversité des résultats et une autre version qui utilise une initialisation par transport optimal qui a l'avantage de mieux respecter la distribution de patchs de l'image de référence.

Le Chapitre IV est dédié à l'étude du calcul de l'attention de façon efficace. Les couches d'*attention* (Vaswani et al. 2017) sont une proposition récente pour modéliser les dépendances longue distance dans les réseaux de neurones tout en permettant un apprentissage de bout en bout. On peut notamment les rapprocher des méthodes classiques non-locales dont font partie les méthodes par patchs. Un des problèmes majeurs cependant est que cette approche a un coût d'exécution qui augmente de façon quadratique avec le nombre de pixels. C'est une limite importante pour le traitement d'images haute résolution et potentiellement les vidéos. Nous montrons que le calcul de l'attention se rapproche d'un problème de recherche de plus proche voisin, ce qui permet une approximation efficace. Dans le cas des images, nous proposons une couche d'attention qui s'appuie sur PatchMatch (Barnes et al. 2009) pour calculer l'attention en utilisant très peu de mémoire. Notre couche d'attention est une bonne approximation dans certains cas mais permet surtout d'utiliser l'attention pour le traitement d'images haute résolution ou de vidéos, ce qui est impossible avec le calcul classique de l'attention.

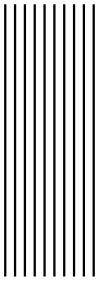
Les chapitres V et VI étudient les nouveaux modèles de diffusion (Ho et al. 2020; Sohl-Dickstein et al. 2015). Ces modèles se sont imposés comme l'état de l'art pour la génération d'images mais aussi pour de nombreux problèmes de restauration d'image. Dans le chapitre V, nous expérimentons leur application au problème de l'inpainting, en limitant l'entraînement à une seule image. Nous comparons en détail cette formulation avec d'autres approches par apprentissage et des méthodes par patchs. Nous évaluons une des caractéristiques centrales de ces réseaux: leur diversité de résultats. Dans le chapitre VI, nous étendons ce travail à la vidéo. La vidéo est parfaitement adaptée à un traitement interne grâce à la redondance temporelle naturelle entre les images composant une séquence. Nous proposons une stratégie d'entraînement particulièrement adaptée à la formulation interne et de l'approche par diffusion en mêlant apprentissage et inférence. Cette stratégie est efficace et permet d'obtenir de bons résultats pour l'inpainting de textures dynamiques complexes.



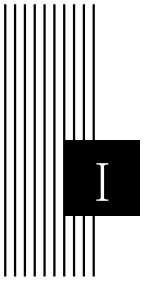
Contents



List of Figures



List of Tables



I Introduction

I.1 Context

Images and video are ubiquitous in today's digital landscape, finding applications in areas as diverse as social media communication, marketing strategies, educational materials, medical diagnostics, and entertainment. From a scientific perspective, we now have a range of tools to take digital pictures of the world in diverse conditions and we store these images in a digital form with millions of pixels. To edit images, the pixel values can be manually or algorithmically modified but doing so in a visually plausible way is still an open challenge. In this thesis we are interested in two different problems. The first one is image *inpainting* *i.e.* how to remove an object from an image, and replace it by generated content that fits visually. The second problem is image generation, how to design algorithms that generate real looking images. These questions are of interest for multiple reasons. First, image editing is an end goal in itself for content creation and artistic purposes. It is desirable to make images easier to edit and create. Second, developing accurate models of complex data is a difficult problem that occurs in many research areas. Some of the tools and techniques developed for image modeling can be applied to different fields. An obvious advantage of images over other types of data, is that anyone can evaluate a natural image model. We are all experts and we do not need a scientific third party to look at the data and judge whether it is realistic or not. This makes the development easier than in other cases, but at the same time this evaluation is not objective, which makes it difficult to measure or optimize.

Image processing aims to improve the visual quality of images or to extract information from them. Improving visual quality covers a wide range of operations: denoising, demosaicing, editing, etc, but they all have one thing in common: producing images. We can further distinguish two subcategories: image restoration, such as denoising, demosaicing, for which we have an objective criterion, and image editing / synthesis, which is more open-ended and subjective. Other high-level problems like classification and semantic segmentation aim to give a human-like understanding of an image beyond the pixels *i.e.* "what objects are seen in this image? how many? where? etc". The work in this thesis falls into the former category, we are interested in image editing and image generation.

More generally, we focus on methods for producing real-looking images with or

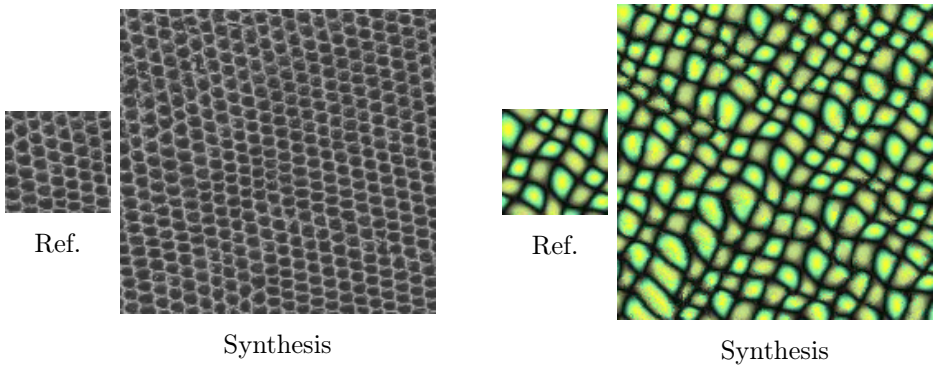


Figure I.1: Texture synthesis examples from Efros and Leung (1999). A new large texture is created using only a reference image as source material.

without constraints. This is a difficult task, given their high dimensionality and our familiarity with natural images. First, images consist of thousands to millions of pixels and the interdependence between the pixels is not obvious. Second, as human beings, we see images all day, all our lives, so we are very demanding when it comes to images, and any defects can be easily detected. For these reasons, creating images that can fool a human being is an extraordinary feat.

We first present the problem of texture synthesis and its natural extension to non-stationary images. Image generation techniques can be adapted for image inpainting. We then focus on the second main topic of this thesis: image and video inpainting.

I.2 Texture synthesis

Textures are easily identified but difficult to define. A texture is a variable but visually homogeneous pattern. Textures vary in scale, stochasticity, regularity, directionality, etc. The aim of exemplar-based texture synthesis is to generate from scratch samples with the same visual identity as a reference image. This reference is limited in size, and the challenge is to extrapolate its properties to a large sample. See Figure I.1 for examples.

This process has applications in image editing and computer graphics. In editing, synthesis can be used to fill-in a background or an occlusion. In computer graphics, textures are used to decorate 2D and 3D objects of all sizes, which then appear in films or video games.

A good synthesis should be visually similar to the reference image, without being identical. It must avoid internal repetition, where appropriate, and external repetition, as the results must be different for different samples of the same texture. For some applications, it is desired that the process is very fast or scales to very high resolutions.

Figure I.2: After a learning phase on a dataset of human faces, the method of Karras et al. (2019) can generate high resolution images of faces. Source: Karras *et al*.

I.3 Image generation

Image generation is an extension of texture synthesis to non-stationary processes *i.e.* any image. In this section, we consider image generation as unconditional generation (without explicit constraints). The problem can be framed mathematically as sampling from an unknown data distribution where our knowledge of the distribution comes from a collection of samples.

Image generation has applications in computer graphics, art, and content creation. Sometimes, such techniques can also extend a limited training set in machine learning where it serves as a complex data augmentation method. Recently, generative models have also shown great succcess in regularizing inverse problems.

A good generative algorithm produces diversified, high-quality, unseen samples that could believably belong to the original training data. These qualities are difficult to measure objectively and quantitatively, requiring either human intervention or approximate evaluation methods. Incredible results are already achieved in human face generation (Figure I.2).

I.4 Inpainting

Inpainting, in-filling, image completion, or disocclusion, is the process of filling a desired region in an image such that the result is visually pleasing. The occlusion usually covers a defect or unwanted content from the picture, and inpainting allows a graceful removal while retaining most of the picture's identity. Applications include art and old photo restoration as well as professional and consumer image editing (Figure I.3). Over time, the physical supports such as canvases, negatives, and films deteriorate as do the images themselves. Inpainting can then be used to restore a digitalized version. Image editing on the other hand is used to alter

photos for artistic purposes, information manipulation, or privacy concerns. The film and photography industries enhance images by removing distracting and unsightly elements such as passers-by, signs, or sound booms. Consumer software now also offers the same capabilities for better vacation photos. These applications are our primary motivation, but we cannot hide the fact that questionable derivative uses are possible. Inpainting can be used to change the meaning of an image and threaten its veracity. This has long been a concern, even with analog photography, as it was already possible (and done) to airbrush someone out of an image (Fineman 2012; King 1997). More recently, *deepfakes* - convincing fully generated images or videos of celebrities - have completed the erosion of trust in visual content. A final application is to remove people from images for privacy reasons, rather than simply blurring their faces. Inpainting them completely preserves their anonymity.

These applications can be extended to video, although the problem is more difficult due to higher dimensionality and temporal coherency. At the same time, a moving occlusion can sometimes reveal the true completion in the case of videos. We present a situation and a complex one in Figure I.4. Algorithms should handle both cases: when the content is truely unknown and when a reasonable guess is possible. The time dimension is very important because the human eye is very sensitive to small temporal inconsistencies.

Finally, while the original problem of inpainting was just to produce a visually pleasing solution, the recent developments have made it possible to describe the desired completion with text (*Adobe Firefly* 2023).

As an image processing problem, we assume that we are given an image and a binary mask indicating the position of the *occlusion* (the region to be filled-in). In any case, we do not use the occluded content of the original image. The difficulty of the task depends strongly on the size of the occlusion. Small independent missing regions are easier to inpaint than large regions for which the problem looks more like a loosely constrained generation problem. In our work, we are more interested in image editing where occlusions can be large. Inpainting is also a very ill-posed problem *i.e.* many different solutions are possible.

It is difficult to evaluate inpainting algorithms for several reasons. The first reason is that we do not have an absolute quantitative metric for the quality or realism of an image that perfectly matches human judgement. Among the metrics we do have, each has different characteristics and limitations. The human eye is still the best option for evaluating the results, but it cannot be used for thousands of images. The second difficulty is that multiple solutions are possible and there is no ground truth available.

I.5 Contributions

We define internal methods as methods that work with very limited data *i.e.* a single image / video. In this thesis, we focus on internal methods for the generation and inpainting of images and videos. The idea is to exploit the self-similarity of images to

Figure I.3: Left: the "inpainting" term comes from the restoration operation on paintings. Right: example of inpainting in digital photography for removing distracting elements. Sources: Bertalmio et al. (2000); Criminisi et al. (2004)

Figure I.4: In video inpainting, a moving occlusion (top) can be easier to inpaint than a static one (bottom) because the background is visible at some point in the sequence. In the second case, the completion must be generated.

inpaint, or generate similar images. Internal methods include patch-based methods as well as (deep) internal learning approaches. Recent attention mechanisms can be seen as a combination of patch-based methods and deep networks. In this thesis, we have investigated several of these aspects for image inpainting or generation.

In chapter III, we develop a patch-based method for single-image generation. The problem of single-image generation was introduced and first adressed by Shaham et al. (2019). They train a neural network on a single reference image to generate variations of the same image. We show that a patch-based algorithm can be devised and perform extremely well. We draw inspiration from early work on exemplar-based texture synthesis, and more recent work on optimal transport. Optimal transport is important for controlling the patch distribution but is computationally expensive, so we only use it at low resolutions. An advantage of our method is that it can generate new samples immediately because no initial learning stage is required.

In chapter IV we investigate the use of attention mechanisms (Vaswani et al.

2017). Attention brings together patch-based methods and deep networks, combining their respective advantages. A limitation of the attention layers is their computational cost, especially for images and videos. Therefore, we propose an efficient attention computation that scales to high resolution images. We exploit the connection between attention layers and the problem of nearest neighbor search. We adapt the PatchMatch algorithm (Barnes et al. 2009) to quickly identify the nearest neighbors, and compute an approximation of at a fraction of the original computational cost. We ensure that our method is differentiable which is the main difficulty in our approach. We validate our approximation on several image editing and image restoration tasks.

In chapter V, we look at a new framework for image generation and inpainting: diffusion models (Ho et al. 2020). The diffusion framework is based on iterative denoising and sampling, that provides diverse and high quality samples. We present an application to internal image inpainting, where a lightweight neural network is trained on a single image. We show results competitive with several state-of-the-art methods: patch-based methods, deep learning approaches, and large diffusion models, on texture images. We also compare different evaluation metrics used in inpainting, and highlight their shortcomings.

In chapter VI, we extend the previous application to videos. Instead of relying on a carefully designed deep architecture, we train a simple neural network to minimize the diffusion loss on a single video. Our method is able to inpaint both the easy case of static backgrounds and, more importantly, the difficult cases of dynamic textures and complex motion. We propose a specific training strategy adapted to diffusion and internal learning. The inference procedure in diffusion models consists of many iterations, we train on specific subsets of iterations instead of all of them. Learning is thus separated for each subset, and consequently simpler, leading to better results. We report a significant improvement over the state-of-the-art for dynamic texture inpainting.

In summary, we make the following contributions:

- in chapter III, we propose a fast patch-based algorithm for single-image generation that does not require a learning phase. We combine optimal transport and a global patch-based energy term for fidelity to the reference image and diversity. We compare it to a deep learning approach and show superior visual results.

- in chapter IV, we develop a very efficient attention layer. It is based on a nearest neighbor approximation of attention. The nearest neighbors are efficiently identified using PatchMatch. The result is an attention layer with a very small memory footprint, which no longer limits the architectures of neural networks and their input sizes. We demonstrate its application to various image editing and restoration tasks.

- in chapter V, we investigate the use of small neural networks in diffusion models. Even with a small number of parameters, we get very good results in

texture inpainting, with diverse and high quality results. The training time is also minimal compared to other deep learning approaches. We discuss several metrics for image inpainting.

- in chapter VI, we extend the work on diffusion to video inpainting. We train a small network on a single incomplete video and inpaint it. This approach handles static and dynamic backgrounds, as well as moving objects. We design a specific training and inference scheme that significantly improves the results over the baseline. We show superior results for dynamic texture inpainting, as verified quantitatively and qualitatively.

These contributions were shared in the form of 4 publications: 1 at an international conference, 1 in a journal, and 2 at national conferences. Publications for the chapters V and VI, on image and video inpainting using diffusion models, are in preparation.

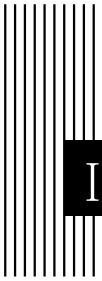
- Chapter III: N. Cherel, A. Almansa, Y. Gousseau, and A. Newson (2022a). "A Patch-Based Algorithm for Diverse and High Fidelity Single Image Generation". In: *2022 IEEE International Conference on Image Processing (ICIP)*. Oral presentation, pp. 3221–3225. DOI: 10.1109/ICIP46576.2022.9897913, URL: https://hal.science/hal-03822204, Code: https://github.com/ncherel/psin

- Chapter IV: N. Cherel, A. Almansa, Y. Gousseau, and A. Newson (2024). "Patch-Based Stochastic Attention for Image Editing". In: *Computer Vision and Image Understanding* 238, p. 103866. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2023.103866, URL: https://arxiv.org/abs/2202.03163, Code: https://github.com/ncherel/psal

- Chapter IV: N. Cherel, A. Almansa, Y. Gousseau, and A. Newson (2022b). "Attention stochastique basée patchs pour l'édition d'images". In: *28° Colloque sur le traitement du signal et des images*. 001-0302. Nancy: GRETSI - Groupe de Recherche en Traitement du Signal et des Images, p. 1209–1212, URL: https://www.gretsi.fr/data/colloque/pdf/2022_cherel952.pdf

- Chapter V: N. Cherel, A. Almansa, Y. Gousseau, and A. Newson (2023b). "Modèle de Diffusion Frugal Pour l'inpainting d'images". In: *GRETSI 2023 :XXIXème Colloque Francophone de Traitement Du Signal et Des Images*. Grenoble, France, URL: https://hal.science/hal-04199282

- Chapter VI: N. Cherel, A. Almansa, Y. Gousseau, and A. Newson (2023a). *Infusion: Internal Diffusion for Video Inpainting*. In preparation. arXiv: 2311.01090 [cs], URL: https://arxiv.org/abs/2311.01090



II Related works

This chapter covers the related works and is organized along 3 methodological perspectives: patch-based methods, deep learning models, and attention mechanisms. This is not a partition of the space of image processing techniques: this will not cover every method related to works presented in this thesis and some methods may also belong to more than one family. In each chapter, the relevant bibliography is recalled and expanded to account for the blind spots of our partition. The goal is really to present the major techniques of interest for us through 2 main applications: *inpainting* and *image synthesis*.

This chapter follows the chronological developments of the field, with first the patch-based methods, then the deep learning models, and finally the attention mechanisms.

II.1 Patch-based methods

II.1.1 Definition

A *patch* is a small square region of an image, a set of pixels. Patches are local descriptors of an image, they locally represent its appearance and can easily be compared and combined. It is easy to construct images from patches and extract patches from images. Formally, for an image $u \in \mathbb{R}^{H \times W}$, a patch of size $n \times n$ at position p[1] is a vector of $\mathbb{R}^{n \times n}$ that we note Ψ_p^u.

It has been observed that natural images are *self-similar*: similar local patterns (and patches) are found at different positions both in-scale and across-scale. We will see that many methods are based on this property either to overcome degradations or to create self-similar images. Patch-based approaches fall into so called *non-local* methods, due to patch operations not being restricted to a local neighborhood.

II.1.2 Exemplar-based Texture synthesis

For texture synthesis, one option is to use a parametric model such as the one of Portilla and Simoncelli (2000). An initial white noise image is optimized so that

[1] It is common to consider patches of odd size so that the central pixel of the patch coincides with the pixel at position p in the image.

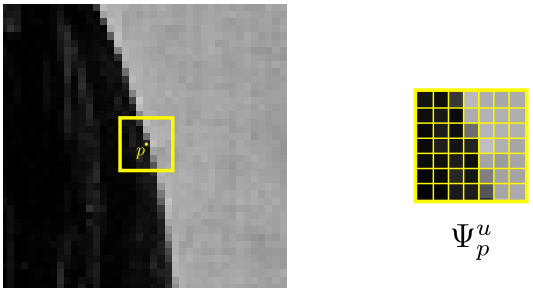


Figure II.1: A image u and a 7×7 patch from this image at position p

the statistical descriptors extracted from the wavelet coefficients are similar to those of the reference texture. An alternative is to rearrange patches from the reference image into a new texture sample. The obvious redundancy of a texture is exploited through its patches by Efros and Leung (1999). They progessively synthesize a texture by finding patches similar to the synthetic neighborhood, and directly paste pixels of the reference image. This algorithm has been refined (Wei and Levoy 2000), accelerated (Liang et al. 2001), and extended to surfaces (Turk 2001). To avoid *verbatim* copies, Raad et al. (2014) estimate a Gaussian model from multiple patches and sample from it.

Kwatra et al. (2005) proposed to shift from greedy algorithms to optimize a global patch-based energy term. Let $\tilde{u}$ be the reference image and u be the new, synthesised image, defined over the image domain Ω. A patch centred on a pixel p in image u is denoted as Ψ_p^u. At each scale, the energy writes:

$$E(u) = \sum_{p \in \Omega} \min_{\tilde{p} \in \Omega} \|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2, \tag{II.1}$$

Their energy is hard to minimize as it requires many iterations compared to the previous greedy approach but produce very good results. They also include a multi-scale strategy which can handle larger structures without increasing the patch size.

Lefebvre and Hoppe (2006) change the patch description to include appearance features. Kaspar et al. (2015) propose several heuristics to tune the list of parameters of texture synthesis (number of scales, patch size) and add a few constraints to avoid repetitions.

One limitation of Kwatra et al. (2005) is that nothing prevents unbalanced synthesis where only a small region of the reference texture is replicated. Optimal transport is one way to solve this shortcoming. Gutierrez et al. (2017) tackle the problem of optimal transport on patches using a linear sum assignment and solve it with the Hungarian algorithm. Next, Galerne et al. (2018) proposed to learn a semi-discrete transport plan which enables multiple synthesis once established. This has been refined next by Leclaire and Rabin (2019) in a two-step approximation which is faster than the previous approach. Finally, Houdard et al. (2021) proposed a semi-discrete differentiable formulation which can also be optimized by

a neural network.

In our work, we are interested in image generation, not limited to textures. However, the problem is more complicated because larger structures are harder to create with patches. It has been explored for images and videos in Kwatra et al. (2003) using GraphCut. Since the recent introduction of the single-image generation problem (Shaham et al. 2019), multiple patch-based approaches have been proposed for images (Granot et al. 2022) or videos (Haim et al. 2022). They extend the work of Kwatra et al. (2005) with more scales and a regularization term.

Surveys are available for state-of-the-art patch-based texture synthesis (Barnes and Zhang 2017; Wei et al. 2009).

II.1.3 Patch-based Inpainting

The extension of texture synthesis to image inpainting is straightforward. In the case of a homogeneous texture to inpaint, the algorithm of Efros and Leung (1999) can be used, the non-occluded content acting as the reference image and the occluded content the synthesized output. It turns out that the self-similarity of natural images is not limited to textures which greatly expands the possibilities of inpainting (Bornard et al. 2002). Drori et al. (2003) propose a multiscale greedy algorithm that computes a confidence map to decide which patch to inpaint first. Criminisi et al. (2004) change the order of the inpainting to first extend the existing lines. This is important for reconstructing the main structures in the image. Sun et al. (2005) constrain the generation with user-defined line guidance.

Wexler et al. (2007) propose an extension to video inpainting using spatio-temporal patches and a global energy minimization similar to Kwatra et al. (2005). The energy is:

$$E(u) = \sum_{p \in \bar{\Omega}} \min_{\tilde{p} \in \Omega} \|\Psi_p^u - \Psi_{\tilde{p}}^u\|_2^2, \tag{II.2}$$

but this time, there is only one image: u, and the domains of definition Ω and $\bar{\Omega}$ correspond to the visible and masked region respectively.

One issue of this energy is the computational complexity of their fundamental step: nearest neighbor search. This problem is solved with the PatchMatch algorithm (Barnes and Shechtman 2010). PatchMatch is a very efficient nearest neighbor search technique very adapted to images because it relies on the self-similarity property of images. It allows nearest neighbor search in log-linear computational complexity compared to a quadratic complexity for naive approaches.

Simakov et al. (2008) introduce a bidirectional similarity measure to optimize completeness and coherence for image and video retargeting. Pritch et al. (2009) optimize the *shift-map* directly instead of alternating it with optimizations over the image pixels. Formulated as a graph-cut problem, this can be solved quickly. Cao et al. (2011) combine an exemplar-based inpainting with a solution obtained with Partial Differential Equation (PDE). The exemplar-based approach allows for a better inpainting of the textures and the PDE of the structures. Bugeau et al.

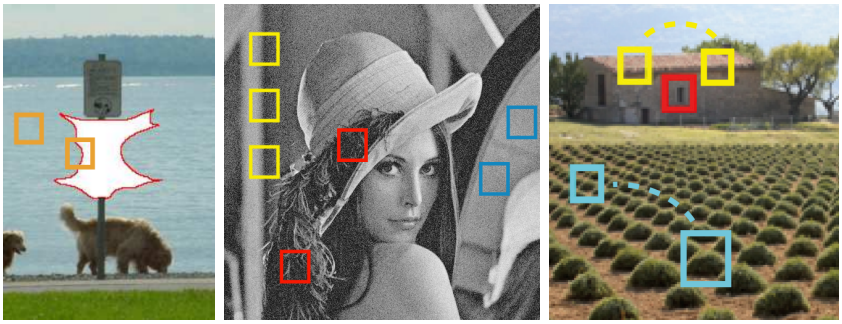


Figure II.2: Patches can be used in different applications due to self-similarity: inpainting, denoising, single image super-resolution. Images adapted from (Criminisi et al. 2004), (Buades et al. 2005), and (Glasner et al. 2009).

(2010) minimize an energy which is a sum of 3 terms: one relying on patches, one on PDE, and one on image coherence. This covers all situations, structures and textures.

This variational non-local inpainting approach has been formalized and analyzed by Arias et al. (2012). Patches can be used to reconstruct the RGB values directly or the gradients (Darabi et al. 2012), which have been found to improve coherence (Pérez et al. 2003).

He and Sun (2012) find that the shift-map is sparse and thus the nearest neighbor search can be drastically accelerated while enforcing coherence more easily. It has been observed by Liu and Caselles (2013), that the textures are better inpainted in multiscale approaches when using the gradients computed at the original image resolution in the distance function. Lee et al. (2016) use a Laplacian pyramid for the multiscale approach to find more accurate neighbors.

Recently, patches have also been used jointly with deep learning, with patches of neural features by Yang et al. (2017). Samuth et al. (2023) generate realistic face images using a small dataset using patches in a latent space. For small datasets, this non-parametric approach is more interesting than learning-based approaches. An advantage of patch-based methods is that they scale to very high resolutions more easily than methods using deep neural networks (Zhang et al. 2022d).

II.2 Deep learning methods

II.2.1 Introduction

As a brief introduction to the topic, deep learning uses neural networks which are compositions of simple operations such as convolutions, or matrix multiplications. We call these operations *layers*. These *layers* have a set of parameters, such as the kernel for a convolution. The main idea is to optimize these parameters θ, called the *weights* of the neural network, for a given dataset.

Convolution	Matrix multiplication
$x \to k * x$	$x \to Wx$
$\theta = k \in \mathbb{R}^{s \times s}$	$\theta = W \in \mathbb{R}^{n \times m}$

Nonlinear operations are also introduced in between these linear layers. These include Rectified Linear Unit (ReLU): $x \to \max(0, x)$, sigmoid $x \to \frac{1}{1+\exp(x)}$ and many others. A neural network is a composition of multiple layers and non-linearities, we note f_θ this composition, where θ is thus the union of all parameters from each layer.

The parameters are optimized on a dataset of pairs $\{(x, y) \in \mathcal{X} \times \mathcal{Y}\}$, where in each pair (x, y) we find the input x and the desired output y. A user-defined differentiable loss function is used to measure the error between y and $f_\theta(x)$. The gradient with respect to the weights is used to optimize the parameters of our neural network. For image restoration, the ℓ_2 error between the predicted pixel values and the ground truth is often used. The most common technique for training a neural network is to optimize the parameters using stochastic gradient descent, using only a small subset of examples at a time. $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the loss function, we optimize the following:

$$\min_\theta \mathbb{E}_{(x,y)}[\mathcal{L}(f_\theta(x), y)] \tag{II.3}$$

The layers can be optimized to solve nearly any problem, provided there is enough data and computing power. When it comes to image restoration, such as denoising, deblurring, or inpainting, the training data can be generated using non-degraded images and a degradation operator. When non-degraded images are not available or the degradation operator is unknown, the problem is more complicated. Some works (Batson and Royer n.d.; Lehtinen et al. 2018) have overcome this difficulty of requiring paired data for deep learning.

II.2.2 Generative models

The distribution of natural images is complex due to its high dimensionality in pixel space. At the same time, it is believed that natural images can be represented in a low-dimensional manifold. Many models have been proposed to find this representation space and the mapping from it to the pixel space. An advantage to this problem is the large amount of natural images we have at our disposal, which are as many samples from this unknown and complex distribution.

Generative Adversarial Networks (GANs) were invented by Goodfellow et al. (2014) as a simple way to learn a generative model on a given dataset. They introduce two neural networks: a generator, whose goal is to generate real-looking images from noise, and a discriminator, whose goal is to distinguish between the images from the real dataset and from the generator's output. This procedure is formalized by the dual optimizations of the generator and the discriminator:

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{II.4}$$

While elegant, such a loss function can be difficult to optimize, and the training may not converge to an optimum in practice. This idea has been extended beyond

image generation, as a real-looking objective function can be of interest for many applications. Indeed, in image restoration, the goal is to recover an image that minimizes both the data fitting term and a regularization term describing natural images. This regularization is learned by a discriminator. Several improvements have been made for their stability (Arjovsky et al. 2017) or the architectures used (Karras et al. 2019).

We can also mention the Variational AutoEncoder (VAE) by Kingma and Welling (2014) in this category, a neural network is trained to encode the images into a Gaussian latent space and reconstruct them from the latent representation. Once trained, images can be obtained by sampling latent variables and decoding them. It is easy to edit existing images, because an encoder explicitly projects images into the latent space, unlike GANs, which typically do not have this component. Compared to GANs, VAEs do not suffer from the mode *collapse* where only a subset of the distribution is represented, nor are they unstable during training, but they do have other problems. The images they produce are often less detailed than those produced by a properly trained GAN.

In both cases, GANs and VAEs, the latent space is an interesting tool for interpolation (Karras et al. 2019), or image editing (Yao et al. 2022). It provides a compact representation space to semantically modify an image.

Among the other deep generative frameworks, we now briefly introduce Normalizing Flows (Rezende and Mohamed 2015). They have a similar idea to VAEs, *i.e.* encoding/decoding images to/from a Gaussian latent space, but rely on learned invertible functions to map the images to the latent space. The invertible property is nice because it does not require training an encoder and decoder separately. In fact, the decoder is the (mathematical) inverse of the encoder. However, this framework limits the design of the neural network because all operations must be differentiable and invertible. In particular, for bijectivity, the latent space must be the same size as the original image space, which is often impractical.

Inspired by text modeling, it is also possible to view an image as a sequence of pixels (van den Oord et al. 2016; Van Den Oord et al. 2016). Generative models for sequences can be trained by predicting the next element given the previous ones over a set of discrete outputs.

Diffusion models. We present in detail diffusion models, which are extensively used in Chapters V and VI. Diffusion models (Ho et al. 2020; Sohl-Dickstein et al. 2015) are a recent improvement over the other generative alternatives in quality and stability. The idea is to introduce a long Markov chain of T states from $q(x_0)$ to $q(x_T)$. Here $q(x_0)$ represents the distribution of clean natural images and $q(x_T)$ is a pure noise distribution and in between, there is a series of intermediate states (Figure II.3). The *forward* process describes the transition from x_0 to x_T through the following kernel:

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right), \tag{II.5}$$

We define $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ which are useful variables in the

following because $q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$. Associated with the forward process, is the *reverse* process that defines the data distribution $p_\theta(x_t|x_{t+1})$ which is also a Gaussian of unknown but learnable mean and variance.

$$p_\theta\left(x_t|x_{t+1}\right) = \mathcal{N}\left(x_t; \mu_\theta(x_{t+1}, t), \sigma_t^2\mathbf{I}\right)$$

A neural network predicts the mean $\mu_\theta(x_{t+1}, t)$ whose parameters θ are optimized to maximize a lower bound of the log-likelihood of the data. In the case of Gaussian variables of known variance, which is a simpler case, then this lower bound has a simple formula which is finally for each timestep t:

$$L_t = \mathbb{E}_{q(x_0,x_t)}\left[\frac{1}{2\sigma_t^2}\frac{\bar{\alpha}_{t-1}\beta_t^2}{(1-\bar{\alpha}_t)^2}\|x_0 - f_\theta(x_t, t)\|^2\right] \tag{II.6}$$

This loss is easier to minimize than the one used for GANs which results in more stable trainings. The training and inference algorithms are detailed in Figure II.4.

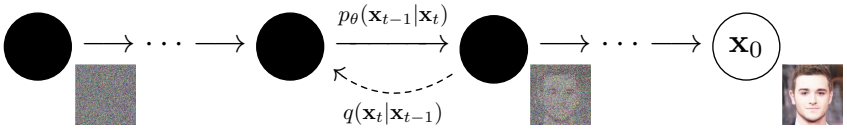


Figure II.3: Diffusion model - Source: Ho et al. (2020)

The iterative nature of diffusion models make them particularly slow for inference. Improvements of different kinds have been proposed. Nichol and Dhariwal (2021) propose adjustments to the loss function and time schedule. Subsequently, they proposed a non-Markovian formulation of the process to skip some steps that they call Denoising Diffusion Implicit Models (DDIM) (Song et al. 2021a). To reduce the complexity of training and inference, Rombach et al. (2022) investigate training a diffusion model in another representation space. They train a VAE to drastically reduce the resolution of the input images and train the diffusion model in this new latent space. Similarly to GAN-based approaches, high-resolution images can be produced by chaining multiple diffusion models (Ho et al. 2022a).

Score-based methods (Song and Ermon 2019; Song et al. 2021b) are very close to diffusion models and precede them temporally by less than a year. Similar to diffusion models, the process is iterative and uses a stochastic refinement starting from pure noise. In this case, the network is explicitly trained to model the score *i.e.* the gradient of the log-likelihood. Given the score at every point of the space, it is possible to design inference schemes to sample high probability samples through Langevin sampling for instance.

Diffusion models have been shown to be very flexible and usable in many contexts: image (Dhariwal and Nichol 2021), audio (Kong et al. 2020), videos (Ho et al. 2022c). Diffusion models can also easily be conditioned on very different variables: texts (Xie et al. 2023), segmentation maps (Zhang et al. 2023), images (Zhang et al. 2023). We refer the interested reader to the survey of Yang et al. (2023b).

Algorithm 1 Training

repeat
$x_0 \sim q(x_0)$
$t \sim \text{Uniform}(\{1, \dots, T\})$
$\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Take gradient descent step on
$\nabla_\theta \|x_0 - f_\theta(\underbrace{\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}}_{x_t}, t)\|^2$
until converged

Algorithm 2 Sampling

$x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
$\mu_{t-1} = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t} x_t + \frac{\beta_t \sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t} f_\theta(x_t, t)$
$x_{t-1} = \mu_{t-1} + \sigma_t \mathbf{z}$
end for
return x_0

Figure II.4: Training and sampling algorithm for Denoisig Diffusion Probabilistic Models.

II.2.3 Texture synthesis with neural networks

To complete the very general framework of data generation and sampling, we present here some techniques specifically designed for texture synthesis. The seminal work of Gatys et al. (2015) has introduced the idea of using neural networks as texture descriptors. These descriptors are derived from the Gram matrices of the feature maps, and have been shown to be very accurate for replicating the fidelity of textures. The associated loss function is also called the *style loss*, and is used both in texture synthesis and sometimes for image inpainting, or other image generation tasks. However, each sample image must be optimized individually. To speed up the synthesis, Ulyanov et al. (2016) have proposed to train a feed-forward generative network for each texture while maintaining a diversity in the output. GANs can also be used to train such a network (Jetchev et al. 2016). An alternative to bypass the pixel optimization is to train a VGG (Simonyan and Zisserman 2015) decoder that can go from the features to the pixels, the texture style is transferred through a whiten-and-color transform, swapping the feature maps statistics (Li et al. 2017). To force the textures to respect a given periodicity, Bergmann et al. (2017) introduce periodic sine maps into the inputs, as an addition to the noise. Recently, Chatillon et al. (2023) encode the texture in a controllable latent space, and decode these vectors with a Style-GAN (Karras et al. 2019) inspired architecture.

II.2.4 Neural Inpainting

The first application of neural networks for image inpainting is the work of Xie et al. (2012) which performs blind image inpainting. An autoencoder is trained to denoise images and is then used for inpainting. Next, Pathak et al. (2016) use a regression network to predict the missing pixel values in a central square region of an image. They add a discriminator to produce less blurred images. In fact, training a neural network for image inpainting is simple: take complete images, and randomly mask some regions in the image. The reconstruction error alone leads to blurry outputs,

so they add a discriminator and its associated loss. The loss function is eventually:

$$\mathcal{L} = \ell_2 + \lambda \mathcal{L}_{\text{adv.}}$$

where $\lambda \in \mathbb{R}$ controls the balance between the two terms. Later, Iizuka et al. (2017) use both a local and a global discriminator to improve image quality on two different levels. Previously a single global discriminator guided the generator, the local discriminator is helpful for better local reconstruction and details. Yang et al. (2017) combine a first inpainting step using a neural network and a second step optimizing a patch-based energy. Other approaches tackle the inpainting problem in two distinct steps: a first rough reconstruction and only then the addition of details. Liao et al. (2018); Nazeri et al. (2019) use line reconstruction, once the edges and object boundaries are fixed, the second network only needs to add the texture and colors. As a first step, Ren et al. (2019) use a smoothed image, without any texture, but preserve the boundaries. Yeh et al. (2017) propose a semantic inpainting approach. Other options include a first step for inpainting the foreground and background (Xiong et al. 2019).

Yu et al. (2018) introduce an attention layer inspired by the traditional methods to take advantage of the existing information. After a coarse inpainting, the result is refined by an attention-based network (Figure II.5). The attention layer is used to fill-in the occluded region using an aggregation of patches from the visible region of the image.

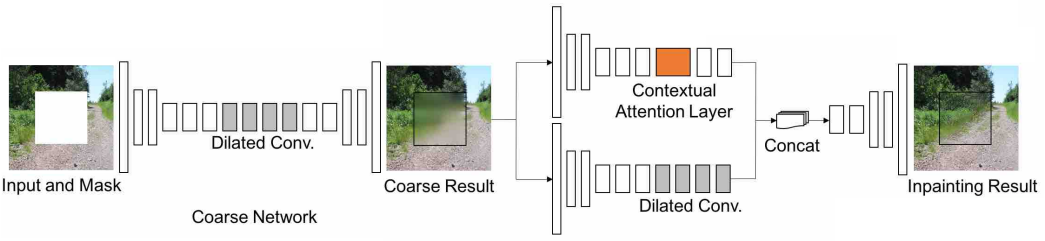


Figure II.5: ContextualAttention: after a first coarse inpainting, the result is refined by a two-branch neural networks. One branch is using Attention. Adapted from Yu et al. (2018)

Shift-net (Yan et al. 2018) uses an attention layer with a single matched patch outside the occlusion. Until now, the mask information was simply introduced into the network by concatenation at the input layer which is suboptimal (Liu et al. 2018b; Yu et al. 2019). In fact, it is possible to propagate this mask in the network to mask invalid features. Inpainting with deep learning networks, and a discriminator have seen several improvements: Wang et al. (2018c) use multiple levels of convolutions, Xie et al. (2019) use a bidirectional attention map. TransFill (Zhou et al. 2021) tackles the problem of reference guided image inpainting, they use a neural network to combine multiple images.

One problem with deep neural networks is that it is more difficult to use them for high resolution images due to increased memory requirements and limited GPU

memory. One solution is to do the inpainting in multiple steps, at different resolutions similar to the Gaussian pyramid approach. Yi et al. (2020) perform a two step inpainting at different resolutions. Zhang et al. (2022d) use a first step with a neural network and a guided patch-based refinement step.

Diverse image inpainting has been a long lasting problem, and one solution has been proposed by Zheng et al. (2019). They use a conditional VAE for introducing diversity. For a single image, multiple solutions can be sampled. Peng et al. (2021b) have a diverse structure generation and deterministic refinement step. Zhao et al. (2023) use a diffusion model for image inpainting, which may then produce different samples using different latent vectors.

To better handle high-frequency details, Kim et al. (2022) use a multi-scale architecture and multi-step processing. On their side, Suvorov et al. (2022) introduce a Fourier transform that allows local convolutions in the spectral domain to have a global effect. Fourier networks are well suited for inpainting repetitive structures whose spectrum is unique.

With the development of diffusion models (Ho et al. 2020), the inpainting problem has seen a great leap in the quality of results. Diffusion models produce great and diverse results (Lugmayr et al. 2022; Saharia et al. 2022b) and the completion can be conditioned on text (Xie et al. 2023) or class (Ho and Salimans 2021). It is even possible to use an unconditional model that is not trained for inpainting but this requires additional attention to converge to a coherent scene (Chung et al. 2022; Kawar et al. 2022a; Song et al. 2023; Trippe et al. 2023). Large off-the-shelf diffusion models can be finetuned for a specific scene (Tang et al. 2023).

II.3 Attention mechanisms

II.3.1 Definition

In this section, we present a new type of deep learning layer: the attention layer. Introduced by Bahdanau et al. (2015) and popularized by Vaswani et al. (2017), attention has been introduced to overcome one crucial limitation of other operations: long-range dependencies. Indeed, convolutions are local operations, multi-layer perceptrons require a fixed input size. Recurrent neural networks can model long-range dependencies, but are inefficient in practice due to their sequential nature and do not take full advantage of the parallel architecture of GPUs. Intuitively, attention mechanisms compute a weighted sum of *all* the input vectors, and their weights depend on pairwise similarities.

We first give the mathematical definition of the Attention layer, the classical dot-product attention mechanism introduced in Vaswani et al. (2017). Let $Q \in \mathbb{R}^{m \times d}$ denote a set of m *queries* packed into a matrix, each query being a vector of $\mathbb{R}^d$. Intuitively, these queries correspond to the different elements for which we want an attention vector. In the context of images, this may be a set of patches. The queries are compared to a set of n *keys*, packed into the matrix $K \in \mathbb{R}^{n \times d}$. These keys correspond to elements that we want to use as a reference to give more or less

importance to the queries. In the image case, the keys may be a set of patches (not necessarily the same as the queries). Given a vector $V \in \mathbb{R}^{n \times d'}$, the attention output is the following weighted sum:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V \tag{II.7}$$

where the softmax function is applied to each row of the matrix QK^T. Recall that the softmax of a vector $x \in \mathbb{R}^n$ is defined by $\text{softmax}(x)_i = e^{x_i} / \sum_{j=1}^{n} e^{x_j}$. Thus, the final result of the attention for each query is a vector, containing a weighted average of values, weighted by the dot product of the query with the elements in K (the keys). This definition solves the problem of long-range dependencies, since the the result for a single query depends on *all* the keys and values. The non-local behavior is thus obtained by choosing the approriate set of keys.

The link between attention and non-local operations from patch-based methods (section II.1) and especially Non-local Means (Buades et al. 2005) is obvious when the dot-product similarity is replaced by the ℓ_2 distance.

The attention mechanism is very general and can be applied to any kind of data. In fact, the queries, keys are just sets of vectors compared by a similarity function and aggregated. Any order is lost. It can be applied to non-Euclidean data as long as an aggregation operation and a similarity measure are defined. Thus the attention layer can be used with queries, and keys from different images, different modalities such as text and images for example (Xu et al. 2018b).

However, the spatial information, *i.e.* the position of the queries relatively to the keys, is lost. It can be artificially restored with positional encoding (Vaswani et al. 2017). A common choice is to represent the positions using Fourier features which have been shown to better reconstruct high frequencies (Tancik et al. 2020).

In the next sections, we will see applications of attention layers in inpainting and generation. Attention has been introduced in deep vision architectures in two different ways. The first one was to interleave attention and convolutions, keeping the usual vision architectures such as UNet or ResNet. This solution allows for long-range dependencies while preserving many local and efficient operations: convolutions. The second one is to use Transformers, a novel neural network architecture based on the attention mechanism (Vaswani et al. 2017). The architecture alternates between attention layers, dense layers, and non-linear operations (Figure II.6). There is no convolution. The query and key descriptors are enriched with a positional information.

II.3.2 Attention-based Neural Image Generation

For texture synthesis, Lu (2022) proposed an extension of the approach of Gatys et al. (2015) using a Transformer. Liu et al. (2020a) compute the attention through transposed feature maps, which can be seen as an exemplar-based approach to texture synthesis, with soft assignments. Guo et al. (2022) have a multi-scale hierarchical Transformer for texture synthesis.

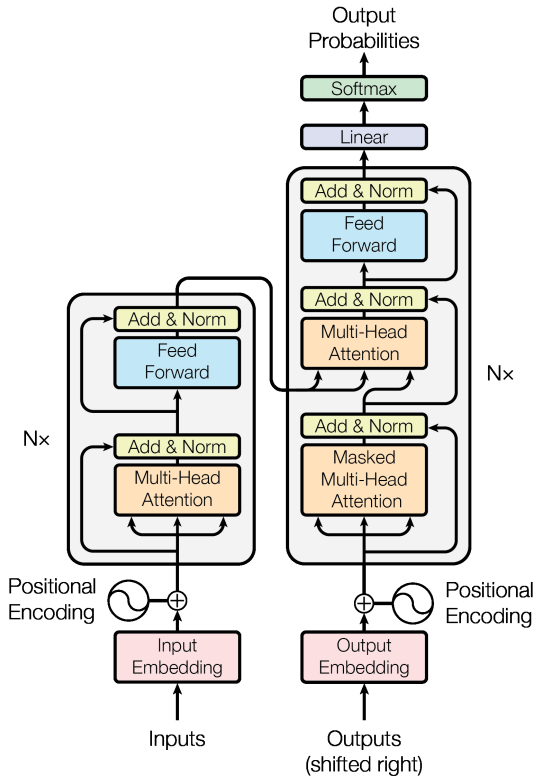


Figure II.6: Transformer block as introduced by Vaswani et al. (2017)

For image generation, attention can be used as a complementary operation to convolutions (Zhang et al. 2019a). The GAN framework can thus be extended to include attention layers, improving the results by simply changing the architecture. For example, Jiang et al. (2021) train both a generator and a discriminator with Transformer architectures. To reduce the computational burden, they propose a multi-scale grid attention which divides the image into small subregions in which attention is computed. Zhao et al. (2021) have a different approximation: they rely on different stride patterns so that the attention of each pattern is computed quickly, but their diversity compensates their individual imprecision. Lee et al. (2022) modify the Vision Transformer (ViT) (Dosovitskiy et al. 2020) architecture and adapt it for GAN training: they use overlapping patches, enforce the Lipschitz constant of the discriminator, and improve the spectral normalization. For high-resolution image generation, Zhang et al. (2022a) use the SWinTransformer (Liu et al. 2021c).

Another application of Transformer architectures for image generation is to use them as autoregressive predictors, similar to what is done in Natural Language Processing (NLP) in 1D. Pixels are predicted successively in raster order. This is experimented by Parmar et al. (2018). They combine a 1D view of the image and a 2D local view to compute the attention. Chen et al. (2020) take inspiration from NLP and use a generative pretraining for downstream classification tasks. Esser et

al. (2021) use an autoregressive model with transformers. The input is first encoded in a discrete latent space, which is optimized by a VAE. Instead of predicting RGB pixels, the Transformer is trained to predict latent pixels which are then decoded into an image. This greatly reduces the dimensionality of the data, simplifying the learning task without losing detail.

In recent diffusion models, it is common to find attention layers that are important for maintaining global coherence (Ho et al. 2020; Rombach et al. 2022; Saharia et al. 2022c). Lately, a Transformer-only architecture has been proposed by Peebles and Xie (2023). Transformers are interesting in very large networks, where they have been found to work better than convolutions and are more parameter efficient (Zhai et al. 2022).

Attention has also changed the way text-to-image generation is done by using cross-attention between visual features and text tokens. This was first used by AttnGAN (Xu et al. 2018b), and is now popular in diffusion-based text-to-image models such as StableDiffusion (Rombach et al. 2022). Attention, and more specifically the attention map, is another tool for controlling the generation. With a constrained attention map, it is possible to force the layout of a generated image (Hertz et al. 2023). Similarly, Chefer et al. (2023) leverage the attention map to ensure that the description of an image fully matches the visual content in text-to-image.

Finally, we have recently seen Transformers been used in a pretraining inpainting task in Masked AutoEncoders (He et al. 2022). Masked Autoencoders trained in this way can be used for discriminative tasks such as classification and object detection, or for generative tasks such as inpainting, or generation. Chang et al. (2022) adapt the framework with a bidirectional Transformer instead of an autoregressive formulation for generation. Tong et al. (2022) extend this application to video, masking an entire temporal tube instead of individual spatio-temporal pixels. The reason for this is to ensure that the inpainting task is hard enough to force the network to learn meaningful representations.

II.3.3 Attention-based Inpainting

For image inpainting, attention layers have bridged the gap between the traditional patch-based methods (Criminisi et al. 2004; Wexler et al. 2007) and neural networks (Pathak et al. 2016). We see early ideas of combining patches and deep networks in Yang et al. (2017). They have a first inpainting step with a neural network which is then refined by optimizing a patch-based energy on features. Song et al. (2018) also split the inpainting into two steps. A first network extracts and completes the features from the input image. A patch-based inpainting is then performed in the feature space. A decoder finally projects the result into image space, but the two networks cannot be trained end-to-end because the patch-based operation is not a proper attention layer.

Yu et al. (2018) introduce the first neural network for image inpainting that uses an attention layer. After a coarse inpainting, the image is refined by a fully convolutional network on one side, and convolution and attention on the other side.

Yan et al. (2018) perform inpainting with a single nearest neighbor, replacing the attention by an argmax. Ren et al. (2019) first reconstruct the image structure, and then sample neighbors locally for the texture. This aggregation is based on the pairwise similarities like in attention. Liu et al. (2019) use an attention layer but with an additional coherence step that refines the weight in the attention matrix based on the neighbors of each patch. Xie et al. (2019) add bi-directional attention map that focuses only on reconstructing the masked region and not the known region. Zheng et al. (2019) introduce short and long term attention for pluralistic completion. They combine short self-attention and long attention between the current layer and the corresponding layer in the encoding path of a UNet.

Zheng et al. (2022a) use restrictive convolutions and a Transformer architecture for inpainting. The attention is computed by distinguishing between masked and visible regions. Wan et al. (2021) start with an autoregressive Transformer for inpainting a low-resolution image and then guide the upsampling with a convolutional network. The first inpainting is diverse due to the sampling step during autoregressive generation. Yi et al. (2020) use contextual attention at lower resolutions and transfer the attention maps to the higher resolutions. Their method handles images with resolutions up to 8K. Zeng et al. (2021) learn an autoencoder to produce images with known self-similarity, this improves the results and avoid computing the attention over all points from the feature map.

Li et al. (2022a) use a Transformer with an iterative filling of the masked region. They achieve diversity by adding noise as one of the inputs in the network. Shamsolmoali et al. (2023) combine a detection Transformer for predicting incomplete objects and then a Texture Enhancement Network that combines convolutions and Transformer blocks. Ko and Kim (2023) use a Transformer for inpainting but with a non-binary mask value for missing regions not covering a full patch. The network can learn to leverage information from these partially incomplete patches. Similar to onion peeling, the information is gradually propagated from the boundaries to the center of the occlusion.

III Patch-based Single Image Generation

III.1 Introduction

Single-image generation is the problem of generating new images that preserve the original patch distribution from a single reference image. This problem was recently introduced by Shaham et al. (2019) and can be seen as an extension of texture synthesis (Efros and Leung 1999; Kwatra et al. 2005; Portilla and Simoncelli 2000), which uses only a single reference but without the assumption of stationarity. The images are not restricted to textures and thus have a "structure", which makes some texture approaches unusable here. Examples of synthesis can be seen in Figure III.1. The two main goals of such image synthesis are to produce results with both high visual fidelity with respect to the original image, but that also have enough diversity. Indeed, it is trivial to achieve fidelity by always producing the same image, and conversely it is trivial to produce high diversity by producing noise. Thus, attaining both goals is a great challenge.

While Shaham et al. (2019) trained a deep neural network to solve this task we show that a simpler approach can also be used. We thus propose an efficient, fully patch-based method for single-image synthesis, that requires no training and produces examples of both high fidelity and diversity. We promote fidelity by minimizing a patch-based energy in a multi-scale approach, and we ensure diversity by carefully choosing the initialization of the example, which turns out to be crucial for diversity. In particular, we propose an initialization based on optimal transport, which is designed to respect the patch distribution of the original image.

This task is useful for several reasons:

- Generating new samples is important for some computer vision problems with limited data availability. Single-image generation can thus generate new samples that are very different from the usual data augmentations (rotations, scaling).

- The second aspect of the problem is modeling the patch distribution from a single image. This is important for other downstream image restoration problems that use this internal prior. Applications include denoising, deblurring, super-resolution, inpainting.

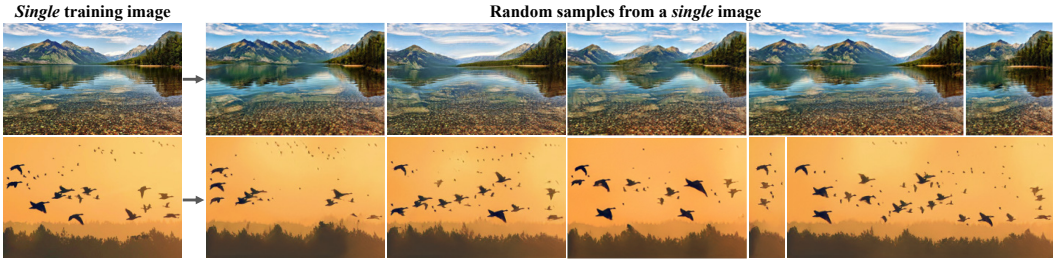


Figure III.1: Single image generation can be defined as generating diverse image samples, visually similar to a reference image but nonetheless different. Source: Shaham et al. (2019)

- Finally, in this setting we also have the perfect opportunity to compare patch-based methods with deep learning approaches for a generative task. Image generation using a large database of images is natural for deep learning methods (Goodfellow et al. 2014), and much less natural for patch-based methods, although possible (Hays and Efros 2007).

III.2 Related works

III.2.1 Patch-based methods

The corresponding literature was already presented in chapter II and we here recall some of the most related works. It is possible to directly reuse the pixel values of the reference image with a non-parametric formulation (Efros and Leung 1999). Kwatra et al. (2005) optimize a global energy term instead of the greedy approach. Early works in image inpainting use the same exemplar-based tools to complete both textures *and* structures (Criminisi et al. 2004; Wexler et al. 2007). The patch distribution in texture synthesis can be controlled by optimal transport (Galerne et al. 2018; Gutierrez et al. 2017; Houdard et al. 2021; Leclaire and Rabin 2019).

For single image generation, Generative Patch Nearest-Neighbor(GPNN) of Granot et al. (2022) is a new patch-based algorithm proposal to avoid the learning phase of SinGAN. They minimize a multi-scale energy similar to the one of Kwatra et al. (2005), but add a regularization term for the distributional aspect. However, this regularization term is computationally expensive and lacks guarantees for distribution convergence. Haim et al. (2022) extend the 2D patch-based version of Granot *et al.* to videos. Their method can synthesize high quality and diverse videos. To accommodate the large size of the input data, they use the PatchMatch algorithm (Barnes et al. 2009) to find the nearest neighbors. The distance is modified to include a completeness score and to promote distribution fidelity.

III.2.2 Internal learning approaches

In this section, we review internal learning approaches, *i.e.* internal methods, that train a deep neural network on a single image. It was shown by Shaham et al. (2019), who proposed SinGAN (Single Image Generative Adversarial Network), that the GAN formulation can be applied to a distribution of patches rather than a distribution of images. Using a lightweight convolutional neural network at multiple scales, they show that they can easily generate variations of a reference image. At the coarsest scale, the generator directly predicts pixels from noise. For the other scales, the generator combines noise with the upsampled image from the previous scale. Hinz et al. (2021) proposed some changes in the SinGAN architecture to allow faster training and better results. Sushko et al. (2021) extend SinGAN to multiple reference images (*e.g.*, from a video sequence) and also better preserve the global layout with a special branch in the discriminator. Zhang et al. (2021) use three different GANs for the different components of an image (texture, structure, semantic). Xu et al. (2021) show that the spatial bias in neural networks due to padding in convolutions is of utmost importance for structured images. They explicitly add the Cartesian coordinates to the pixels during generation. For other modalities, Gur et al. (2020) have proposed to combine a Patch-VAE and Patch-GAN (Isola et al. 2017) to synthesize new videos from a single example. Wu and Zheng (2022) adopt the framework of Shaham et al. (2019) for 3D volumes. However, they use 2D projections to significantly reduce the complexity. More recently, several diffusion approaches have been proposed for single-image generation by Kulikov et al. (2023); Nikankin et al. (2023); Wang et al. (2022).

Few works have investigated the use of neural networks on a single image for image restoration. Ulyanov et al. (2018) have introduced Deep Image Prior, a neural network as a regularization component for solving inverse problems. In this work, they show that the convolutional architecture is key in modeling natural images. Chatillon et al. (2022) extend SinGAN to images with strong self-similarity across scales for single-image super-resolution. Shocher et al. (2019) learn the distribution of patches in an image, enabling smart image retargeting that synthesizes new regions instead of interpolating. Alkobi et al. (2023) derive an inpainting network from the work of Shaham *et al.* that learns on a single occluded image. The completion is refined at multiple scales by a generator. The coarsest initialization is done with the internal approach of Ulyanov *et al.* For denoising, internal learning has been popularized following the developments of Noise2Noise (Batson and Royer n.d.; Huang et al. 2021; Krull et al. 2019; Lehtinen et al. 2018). Extensions to videos have been successful due to their high degress of self-similarity for temporal consistency (Lei et al. 2020; 2023), and video inpainting (Ouyang et al. 2021; Zhang et al. 2019b).

III.3 Method

III.3.1 A patch-based approach for single-image generation

We propose **PSin**, a **P**atch-based algorithm for **Sin**gle image generation. Our algorithm exploits the Gaussian pyramid, minimizing an energy from the coarsest scale to the finest scale. We avoid costly learning stages by copying patches from the reference image.

We now introduce the patch-based optimization problem that we solve to produce the output image. Let $\tilde{u}$ be the reference image and u be the new, synthesised image, defined both over the image domain Ω. A patch centred on a pixel p in image u is denoted as Ψ_p^u. At each scale, we minimize a global energy similar to the one of Kwatra et al. (2005):

$$E(u) = \sum_{p \in \Omega} \min_{\tilde{p} \in \Omega} \|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2, \tag{III.1}$$

where $\|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2$ is the ℓ_2 distance between the pixels of the patches Ψ_p^u and $\Psi_{\tilde{p}}^{\tilde{u}}$. This energy specifies that a good solution is one where each patch is similar to its nearest neighbor (NN), with respect to the ℓ_2 patch distance, in the reference image.

This energy is efficiently minimized by alternating a nearest neighbor search step and a reconstruction step, which were identified as two steps of a hard Expectation-Maximization (EM) by Kwatra et al. (2005). Let $\psi : \Omega \to \Omega$ represent the nearest neighbor mapping, the energy can be rewritten as:

$$E(u, \psi) = \sum_{p \in \Omega} \|\Psi_p^u - \Psi_{\psi(p)}^{\tilde{u}}\|_2^2, \tag{III.2}$$

The minimization of this energy is done by alternating the minimizations on u and ψ, using the following for ψ:

$$\psi(p) = \arg\min_{\tilde{q} \in \Omega} \|\Psi_p^u - \Psi_{\tilde{q}}^{\tilde{u}}\|_2^2 \tag{III.3}$$

The reconstruction step is given, for each pixel p by:

$$u(p) = \sum_{q \in \Psi_p} e^{-\|\Psi_q^u - \Psi_{\psi(q)}^{\tilde{u}}\|_2^2} \, \tilde{u}(\psi(q) + (p - q)) \tag{III.4}$$

Using the efficient approximate nearest neighbor algorithm PatchMatch (Barnes et al. 2009) for an approximation of Equation III.3 makes generation possible in seconds. No training is required at any time. Our full algorithm is described in Alg. 3.

This energy is minimized at each scale starting from the coarsest to the finest scale, adding more and more details. The coarse structure *e.g.* position of the main objects and structures, is defined at the very beginning similarly to SinGAN. The initialization and first scales are thus crucial steps in our algorithm (Figure III.2), and must be carefully considered. We have two strategies for initialization.

Figure III.2: Images at the end of the optimization process at each scale. We already see the main structure of the image after the very first scale.

Algorithm 3 PSin / PSinOT

```
u ← init()                                  ▷ Random noise or optimal initialization
for s ∈ [2, 1, 0] do
    u ← rescale(u, scale = s)
    for i = 1..10 do
        ψ ← NN-Mapping(u, ũ)                ▷ Expectation
        u ← Reconstruction(ψ, ũ)            ▷ Maximization
    end for
end for
```

III.3.2 Random Initialization - PSin

In the simplest approach that we first consider, Gaussian noise can be used as an initialization. We refer to this method as **PSin**. While simple, this can lead to interesting structures provided that the starting resolution is low enough with respect to the patch size. Figure III.3 illustrates this phenomenon: with 3 scales, the generated image has poor global coherency. In general, this approach ensures some diversity but has limited fidelity, in the sense that *e.g.* it does not respect the distribution of patches in the reference image. To address this problem, we turn to another initialization.

(a) Reference (b) 3 scales (c) 4 scales (d) 5 scales

Figure III.3: PSin results with different number of scales. With 3 scales, the general structure is not coherent; this is addressed by initializing at a lower scale.

Algorithm	Learning-Free	Distribution	Scalable
SinGAN (Shaham et al. 2019)	✗	✓	✓
Patex (Houdard et al. 2021)	✓	✓	✗
GPNN (Granot et al. 2022)	✓	✓	✗
PSin	✓	✗	✓
PSinOT	✓	✓	✓

Table III.1: Combining PSin with a good initialization gives an algorithm that does not require learning, respects the original distribution, scales to higher images and has limited runtime

III.3.3 Optimal distribution - PSinOT

In this approach, we turn to tools from optimal transport to build a loss that accounts for the distance between the probability distribution of patches from the input and the one of the synthesized image. This enables us to produce an initialization which has a similar patch distribution to the reference image. This loss, the Wasserstein-2 distance, is minimized at a coarse scale to produce the desired initialization. Our method is inspired by the work of Houdard et al. (2021), who proposed a patch-based optimal transport algorithm for texture synthesis. Minimizing the Wasserstein-2 distance ensures that the generated samples have the correct patch distribution, *i.e.* the distribution of patches in the reference image.

Using our notations, the semi-dual problem at a single scale is the following:

$$OT(u) = \max_{\beta \in \mathbb{R}^{|\Omega|}} \sum_{p \in \Omega} \min_{\tilde{p} \in \Omega} \left(\|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2 - \beta_{\tilde{p}} \right) + \sum_{\tilde{p} \in \Omega} \beta_{\tilde{p}} \tag{III.5}$$

using β as the dual variable. The cost is minimized by alternative optimizations on u and β.

This process is long and computationally expensive. It scales quadratically with the number of patches which makes it unpractical for single-image generation. To combine the strengths of both approaches, we propose to first create a coarse initialization by optimization and then switch to our fast generator for performance. We call this method **PSinOT**.

III.3.4 Fast nearest neighbor search

Our algorithm spends most of its computational effort finding the nearest neighbors. Unfortunately, a naive approach to this search does not scale well, with a complexity of $\mathcal{O}(n^2)$ for n patches. Therefore, we turn to PatchMatch (Barnes et al. 2009) for a fast computation of nearest neighbors. This makes PSin and PSinOT scalable algorithms. Table III.1 summarizes the advantages of each method. In practice, PSin can generate a new sample in 15 seconds on the CPU, while SinGAN first requires 1 hour of training on GPU. GPNN takes 6 seconds to generate a sample on GPU. The optimal initialization in PSinOT adds 15s (see Table III.2 for running times).

Algorithm	Runtime CPU (s)	Runtime GPU (s)
SinGAN	-	3700
GPNN	32	6
PSin	15	-
PSinOT	72	30*

Table III.2: Runtimes of each algorithm on CPU / GPU. PSin is the fastest on CPU. The optimal initialization slows down the total run time but is still faster than training SinGAN. *Initialization on GPU and refinement on CPU

(a) u_{l-1}

(b) u_l - image upsampling

(c) u_l - ψ upsampling

Figure III.4: For the same image at a coarser scale, the results are very different when upsampling through u or through ψ. Details are lost when using a simple bilinear upsampling of the image u (lower left).

III.3.5 Upsampling the shift-map

Our generated image is described by two different variables: the pixel image u and the nearest neighbor mapping ψ. When we change scale, we then have the option of upsampling either u or ψ. While upsampling the image is the easiest operation, it is not the most accurate. In Figure III.4 we show the results of two different upsampling operations. The first method is to upsample u, optimize ψ and then reconstruct of u from ψ. The second method is to upsample ψ and then reconstruct u from ψ. The first option causes a large loss of detail because the nearest neighbors have to be recomputed and are not preserved.

In practice, upsampling on ψ is done by enlarging the shift map. Remember that $\psi_l : \Omega \to \tilde{\Omega}$ is the nearest neighbor map of scale l. If the patch positions are represented by integers and we use of scales of ratio 2, then:

$$\forall p \in \Omega_l, \psi_l(p) = 2\psi_{l+1}(p^\star) + p - 2p^\star \tag{III.6}$$

where $p^\star = \arg\min_{p' \in \Omega_{l+1}} \|p - 2p'\|^2$ is the nearest (spatial) neighbor at the lower scale. The careful reader will notice that the reconstruction step requires both ψ, which we just defined *and* the distances at the current scale $\|\Psi^u_p - \Psi^{\tilde{u}}_{\psi(p)}\|^2$, for simplicity we also upsample these using nearest neighbor interpolation.

Optimal transport to PSin The similar formulations of Equation III.1 and Equation III.5 suggest an easy way to transition between the two (on the same scale):

$$\psi(p) = \underset{\tilde{p}\in\Omega}{\arg\min} \|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2 - \beta_{\tilde{p}} \tag{III.7}$$

The shift-map is a better representation of the image than the pixel image u which helps preserve the details from the optimal initialization to the rest of the algorithm.

III.4 Results

III.4.1 Experiment details

We implement our algorithm in Python and speed it up with Numba[1] on CPU. We typically use 4 scales with factor 2 and set the patch size to 11, which is comparable to the receptive field of SinGAN (Shaham et al. 2019). With images of original size around 256×256 this brings the initialization size to 32×32, close to the patch size. We do 10 iterations of EM at each scale before switching. Upscaling is done by interpolating the shift map ψ rather than interpolating the image u. For PSin, we use a Gaussian noise $\mathcal{N}(0.5, 1)$. For PSinOT, optimal transport is employed for the first two scales and then the standard EM is used. Our patch distance includes the RGB difference and the norm of the horizontal and vertical gradients of the intensity images, which have been shown to improve texture synthesis (Liu and Caselles 2013; Newson et al. 2014). We have an additional parameter $\alpha = 5$ which balances the weight of this term in the patch distance:

$$\|\Psi_p^u - \Psi_{\tilde{p}}^{\tilde{u}}\|_2^2 = \sum_{r\in\mathcal{N}_p} \|u(r) - \tilde{u}(r)\|_2^2 + \alpha \cdot \|\nabla u(r) - \nabla \tilde{u}(r)\|_2^2$$

Our code is available online: `https://github.com/ncherel/psin`. For comparisons, we have used the official implementation of each work with their default parameters[2]. ($\sigma = 0.75$ for GPNN).

III.4.2 Quantitative results

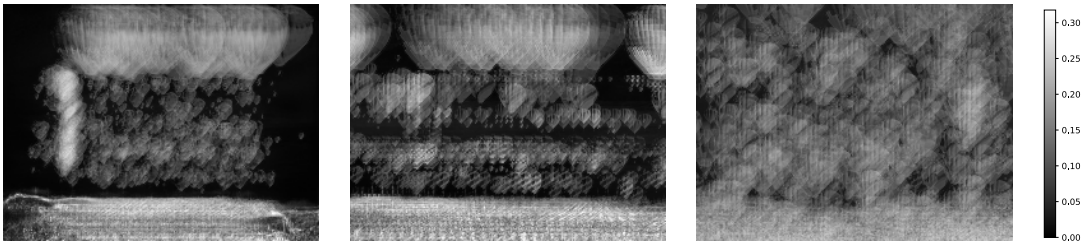
Evaluating image generation is challenging in itself. Therefore we rely on several metrics to compare the methods. We use the Fréchet Inception Distance (Heusel et al. 2017) which measures the distance between Gaussian distributions of features and its adaptation to single-image generation (SIFID) (Shaham et al. 2019). SIFID uses lower level features that FID, to accomodate for the small number of data points. A low SIFID means that images have the same feature distribution and contain the same visual objects. For fidelity, we include the optimal transport cost on patches (derived from the work of Houdard et al. (2021)) which measures the true distance between patch distributions at the finest scale. In practice this is done

[1] Numba is a compiler for Python `https://numba.pydata.org/`

[2] GPNN: `https://github.com/iyttor/GPNN`, SinGAN: `https://github.com/tamarott/SinGAN`

Algorithm	SIFID ↓	Optimal Transport ↓	Diversity ↑
SinGAN	0.12	1.34	0.34
GPNN	**0.02**	*0.52*	0.40
PSin	0.45	0.94	**0.62**
PSinOT	*0.06*	**0.36**	*0.53*

Table III.3: PSin is very diverse but has limited fidelity (SIFID, optimal transport). PSinOT combines high diversity and similar distribution. **best**, *second best*.



SinGAN PSinOT PSin

Figure III.5: Standard deviation images starting from the *balloon* reference image. SinGAN has very low variance / diversity along the edges, while PSinOT has higher variance with some recognizable shapes. PSin has a uniform variance, indicating a low structural component.

by optimizing Equation III.5 in the dual variable β only, with 1000 iterations of gradient ascent.

We also measure the diversity of generated images. The entropy of the distribution is intractable and we use the measure of diversity given in Shaham et al. (2019). For each image, the pixel diversity is the standard deviation of pixel-wise intensities when stacking all generated images $(u_i)_{i=1..n}$:

$$\begin{aligned} \text{Diversity} &= \frac{1}{|X||Y|} \sum_{x,y} \sigma(x,y) \\ &= \frac{1}{|X||Y|} \sum_{x,y} \frac{1}{n} \sqrt{\sum_{i}^{n} (u_i(x,y) - \mu(x,y))^2} \end{aligned} \tag{III.8}$$

We compare with the results of SinGAN, GPNN (Granot et al. 2022), PSin, and PSinOT. We use a dataset of 50 images from Places (Zhou et al. 2017), the same as in Shaham *et al.*, and compute our metrics on 50 samples for each image. Table III.3 confirms that PSin produces very variable results with limited fidelity. SinGAN produces less diverse output with a lower SIFID. Finally GPNN and PSinOT both have good diversity and fidelity scores but our approaches yield significant improvements both in the fidelity of patches distribution and in diversity.

Figure III.6: SinGAN (left) and PSinOT (right). Our method does not have network artifacts.

III.4.3 Qualitative results

We also present visual results in Figure III.7 which are representative outputs. SinGAN's results are visually pleasing from a distance but suffer from network artifacts when viewed closely (Figure III.6). However, SinGAN's results are diverse globally and locally. GPNN produces visually coherent results but may reproduce the same image multiple times. Finally PSinOT has a coherent structure and satisfying details. More results are shown in Figure III.8. In these examples, the difference between PSin and PSinOT is clearly visible: the patch distribution is not respected in PSin. In these 3 examples, the sky is missing in the PSin samples while it is correctly placed in PSinOT. However, both PSin and PSinOT results are of high quality. Results from GPNN are very close to the reference image. See our website for more examples: link to website.

We show in Figure III.10 that some neural network artifacts never appear with our method however our method suffers from other defects. Blurry borders can happen when no suitable patch is found to join two different regions. At the same time, our method cannot create new content.

Diversity We show in Figure III.5 the variance map before spatial averaging. We see that in this simple diversity measure highlights the limits of SinGAN: as mentioned by the authors, diversity is very low along the edges of the images and the corners. This is because of the convolutions in the neural network which have to use a default value for the outside region. It is then very easy for the generator and the discriminator to specifically learn these patterns which include an unusual number of zeros. On the other side, PSinOT has a variance more uniformly distributed and PSin's variance has even less structure, indicating a high diversity of results.

III.4.4 Other applications

SinGAN's paper proposes multiple applications once the network has been trained for single-image generation. These applications include: single-image super resolution, sketch to image, image animation, image harmonization.

We show that our approach can be used for sketch-to-image. The idea is similar

Figure III.7: Reference image and 6 uncurated samples for each algorithm to showcase diversity. SinGAN and PSin produce diverse shapes, however the visual quality of SinGAN is clearly lacking. GPNN introduces very little diversity, keeping the main structure of the reference image (the single rock arch). PSinOT produces original geometries while maintaining better visual fidelity than SinGAN.

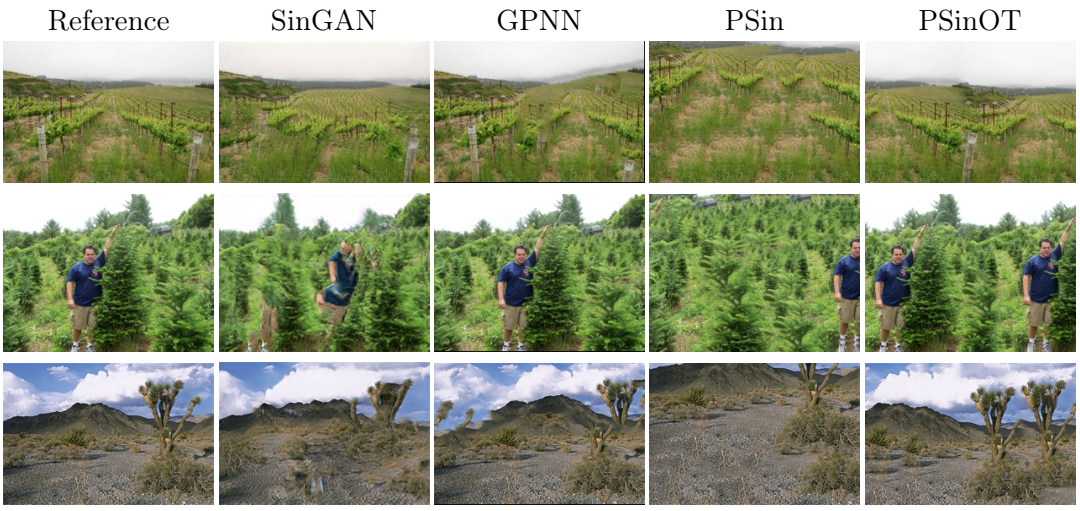


Figure III.8: Results from our different algorithms.

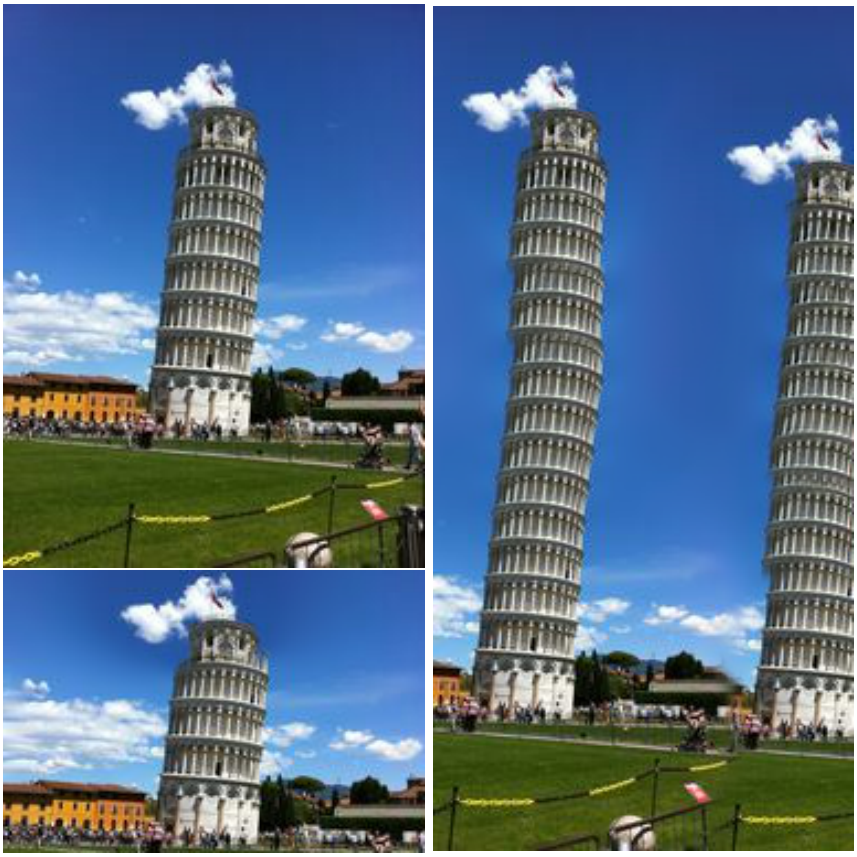


Figure III.9: Reference (top left) and two generated images with different aspect ratios. Original image by Jainam Mehta on Unsplash

Figure III.10: Limitations of patch based methods vs deep networks: first row shows that our results guarantees good looking patches compared to a neural based network but lacks originality which may be desired in the second row.

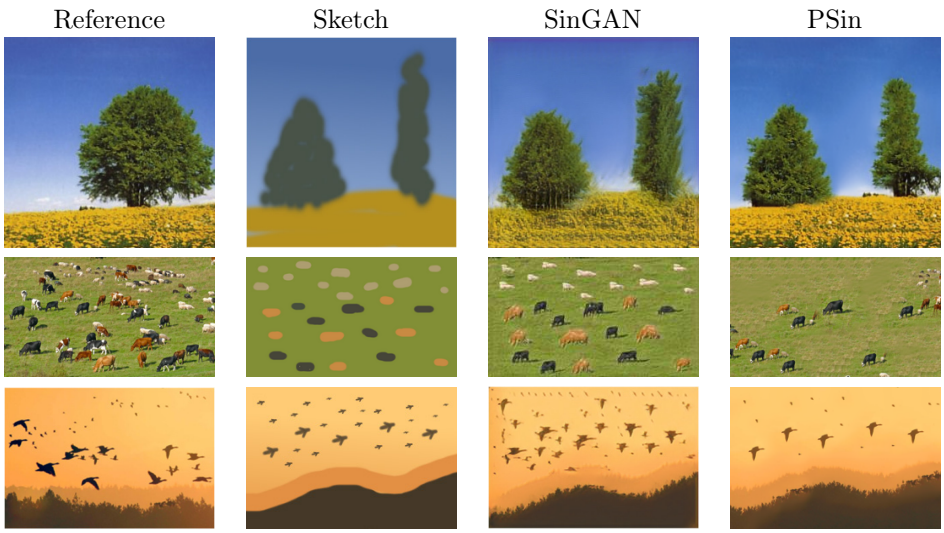


Figure III.11: Paint-to-image: our approach allows partially to generate an image from a sketch, we compare our results with those of SinGAN. The results are comparable. Sometimes it fails to respect the constraints but often produces sharper results.

to Image Analogies (Hertzmann et al. 2001) but does not require a full pair, the color is used for matching. In this setting, a simplified image is used to guide the generation. In practice, instead of using random noise as initialization, we can take an image and subsample it to start from a template. So we start with a first optimization on ψ. While there is no guarantee that all the structure will be preserved, is can work in simple settings, sometimes better than SinGAN (Figure III.11).

In our approach we can't offer the other applications of SinGAN due to our patch-based approach. We do not have access to a latent space that can be navigated to create animations, for example. In our case, navigating through the initial noise produces different solutions without any transitions.

We have not explored other applications suggested by the Shaham *et al.*, such as super-resolution or harmonization, and we must admit that while these applications are possible with our patch-based approach, they are not straightforward in their implementation. Glasner et al. (2009) have investigated single-image super resolution using patches and a single image. Image harmonization could be built by replacing the Gaussian pyramid with a Laplacian pyramid to preserve the content present in the lower scales.

III.5 Conclusion and perspectives

In this chapter, we introduced a patch-based approach to single image generation. Contrary to SinGAN, it does not require training but still generates diverse and visually pleasing images. Our algorithm is based on the minimization of a patch

energy, which encourages fidelity to the reference image. In order to ensure that the patch distribution of the reference image is respected, we propose an initialization based on optimal transport. We have compared our results quantitatively and qualitatively with the original SinGAN and another patch-based method, showing that our approach achieves better fidelity and diversity than the previous two. Although patch-based methods work well here, in future work, we may want to preserve some of the advantages of convolutional architectures, replacing only the discriminator with a patch-based approach.

Our approach also has some limitations: it can only combine existing patches in a pleasant way and cannot create new content. Another problem with our algorithm is that while it requires no learning, it is slower at inference. In some cases, where many inferences are needed, this can be a problem.

Matching space and reconstruction space. Our method works in the RGB space, using familiar image processing tools such as the Gaussian pyramid. But we began to observe that these design choices, while sound, were sometimes not optimal: for example the preferred descriptor space is to use the horizontal and vertical gradients on top of the RGB information from the patches. Other choices can be made to improve the compactness of patch descriptors (Korman and Avidan 2011). To avoid handcrafting these parameters and descriptors, it is possible to use a learning-based approach. Many classical methods have a set of parameters that can be optimized on a given dataset rather than blindly tested (Monga et al. 2021).

The first application of learning descriptors would be to start integrating semantics that are available in a neural network pyramid rather than a Gaussian pyramid. Most popular latent spaces are often derived from networks trained for object classification which correctly reflect the human perception (Gatys et al. 2015; Zhang et al. 2018a), but no associated decoder is available to go from the feature level to the pixel level. Changing the matching space would not lead to drastically different results if we still perform the reconstruction step simply. Another option would be for instance to look for a different reconstruction space.

The reconstruction space can be built manually: for example, Darabi et al. (2012) aggregate the gradients and then reconstruct the image from them using following the Poisson equation. Another option is to use a neural network to perform the reconstruction step: it is expected that averaging deep patches and decoding them would lead to different results than averaging patches of pixels.

Finally, we can compare this approach to attention layers or Transformers (Vaswani et al. 2017) where the base operation is very similar to a non-local (or patch-based) operation. These attention layers use projection layers in a set of *queries*, and *keys* for matching and a set of *values* for the reconstruction, these layer parameters are fully learned. However, it is unclear what the objective function optimized by such a neural network should be.

Transformed patches. A second way to create new content would be to use a larger patch dictionary, a source of patches, by introducing augmentations of patches and recombining them together. This has been done in several papers for rotations and scalings to overcome the inherent limitations of patch-based methods (Barnes

and Shechtman 2010; Darabi et al. 2012; HaCohen et al. 2011).

For the transformations to consider, a natural choice is the set of rotations and scalings, studied by Darabi et al. (2012) for image melding. A new direction could be to incorporate non-linear augmentations such as deformations (Simard et al. 2003) or neural augmentations. In fact, neural networks are flexible enough to modify images or patches in non-linear ways (Karras et al. 2019; 2020).

IV Patch-based Stochastic Attention

Attention mechanisms (Vaswani et al. 2017) have become of crucial importance in deep learning in recent years. The use of attention has helped deep learning introduce long-range dependencies. This addresses a drawback of the commonly used convolutions, which are *local* operations. Even if deeper networks and dilated convolutions can address this drawback by extending the receptive field of the network, they nevertheless fall short when non-local behaviors are important. This happens in text processing where words referring to a same subject can be far apart, in video classification (Wang et al. 2018b) when the action is changing position in time and space, or for image editing to maintain global coherence (Yu et al. 2018; Zhang et al. 2019a).

However, computing the full attention matrix is an expensive step with heavy memory and computational loads. These limitations curb network architectures and performances, in particular for the case of high resolution images. Among the popular proposals for efficient attention mechanisms, we find drastic approximations like local attention (Parmar et al. 2018), attention subsampling (Yu et al. 2018), sparse strided attention (Child et al. 2019), clustered attention (Vyas et al. 2020), or linear approximations of the non-linear softmax (Choromanski et al. 2020; Katharopoulos et al. 2020; Wang et al. 2020). However these methods are not sufficient for high-resolution image editing where these approximations are too restrictive. Localization is unable to model long-range interactions, and subsampling is unable to provide pixel-precise attention maps, leading to blurred results or jagged edges where they should be smooth.

In this work, we aim to drastically reduce the memory and computational requirements of the attention layer through a novel approach that avoids the artifacts of other attention approximations. It turns out that the attention layer is closely related to the problem of Nearest Neighbor (NN) search. The softmax in the attention computation in effect biases the distribution of weights towards a handful of similar points. We show that when dealing with images, attention mechanisms can be efficiently estimated via an Approximate Nearest Neighbor (ANN) search. For this search, we turn to the prominent PatchMatch algorithm (Barnes et al. 2009), a fast algorithm for ANN search that is especially efficient for comparing similar images. In order to overcome the computational limits of the traditional attention layer,

we propose an attention layer which employs the PatchMatch method, specifically designed for the case of images, which we name "Patch-Based Stochastic Attention Layer" (PSAL).

Furthermore, we propose different approaches, based on patch aggregation, to ensure the differentiability of PSAL, thus allowing end-to-end training of any network containing our layer. PSAL has a small memory footprint and can therefore scale to high resolution images. It maintains this footprint without sacrificing spatial precision and globality of the nearest neighbors, which means that it can be easily inserted in any level of a deep architecture, even in shallower levels. PSAL has a small memory impact, scaling linearly with the input image size. As a result, it can be applied to large-size two-dimensional inputs and in particular allows us to apply the attention mechanism to high resolution images or to both shallow and deep 2D feature maps. Such situations are out of reach for classical attention mechanisms, because of memory limitations, and require the use of a sub-sampling strategy or a restriction to very deep features. These approaches are problematic for image editing in several situations: handling high resolution images, using low level features closer to the original image or applying attention mechanisms at the pixel level.

IV.1 Related work

IV.1.1 Image editing

We briefly recall some of the important references for image editing, and single-image super-resolution. As already presented in chapter II, image editing has long used patch-based approaches for inpainting (Criminisi et al. 2004; Wexler et al. 2007), retargeting (Barnes et al. 2009), style transfer (Frigo et al. 2016), and other image editing tasks (Darabi et al. 2012), making heavy use of NN patches. More recently, the work of Yu et al. (2018) has pioneered inpainting with attention layers.

Self-similarity has also been identified as a key component for single-image super-resolution (Freeman et al. 2002; Glasner et al. 2009; Michaeli and Irani 2014). For single-image super-resolution, deep networks have progressively replaced these methods with the introduction of large datasets (Zhang et al. 2018b). Attention layers have been integrated within recent image restoration networks, increasing their modeling power with non-local operations (Dai et al. 2019a; Liu et al. 2018a). The work of Mei et al. (2020) highlights the importance of attention across multiple scales.

IV.1.2 Efficient attention mechanisms

The attention layer is very flexible at the cost of high computational complexity; attention for a single query is a function of *all* keys. This is a major limitation because the computational cost grows quadratically $\mathcal{O}(n^2)$ with the input size, as opposed to linearly for fully convolutional architectures. This is problematic for Natural Language Processing (NLP), but even more so for image and video processing, which

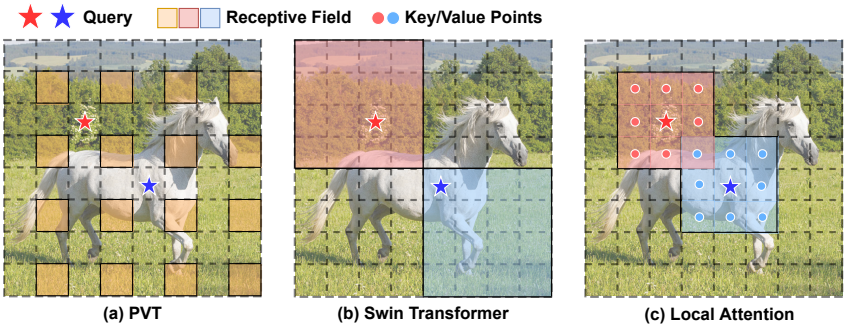


Figure IV.1: Efficient attention by subsampling (a) (Wang et al. 2021d), or local attention (b-c). Source: Pan et al. (2023)

deal with large inputs.

Some works have focused on reducing the amount of pairwise distances to compute. The immediate idea is to split the input into several small chunks and only compute the attention locally. This is done by Parmar et al. (2018); Plötz and Roth (2018), and we refer to it as Local Attention. Note that the non-local behavior of attention is lost with this approximation. These local windows can be shifted so that two successive attention layers use different query and key sets, this is especially interesting for images because they have 2 dimensions that can be used to shift the window along (Liu et al. 2021c). Vaswani et al. (2021) introduce HaloNet with a spatial subdivision of queries and keys, they show improved results in classification error and throughput. The Local Attention can be refined with multiple scales and regional attention (Chen et al. 2022). Pan et al. (2023) extract the local vectors with a deformable convolution.

Grouping queries and keys into small subsets for comparison is an interesting approach to efficiency, but is not limited to spatial proximity. It is possible to first regroup similar vectors based on similarity and then compute the attention in that group. Roy et al. (2021) learn a sparse attention routing operation with k-means. Wang et al. (2021b) combine a clustering approach with a sliding window attention. Similarly, Vyas et al. (2020) use a first clustering step to identify similar keys and queries and reduce the overall cost of attention. Tay et al. (2020b) use a differentiable sort as a first step and then compute the attention locally. Lample et al. (2019) design an attention layer around key stores, the attention approximation is then a problem of finding nearest keys in these stores. Kitaev et al. (2020) introduce a layer based on Locality Sensitive Hashing (LSH), they hash the keys and queries to different buckets, and compute the attention only in these buckets. Calian et al. (2019), propose an attention layer derived from PatchMatch to compute the attention, however in their preprint the authors do not verify the validity of their layer in a practical deep learning setting, meaning that there is no way of knowing if the layer functions correctly.

It is also possible to reduce the number of computations by using sparse or strided extraction patterns. Child et al. (2019) propose the Sparse Transformer

which factorizes the attention matrix with fixed strides and patterns. Correia et al. (2019) learn a sparse pattern, their multihead attention can also have different levels of sparsity. Beltagy et al. (2020) combine a local strided attention and global attention. Zaheer et al. (2020) combine random attention, local attention, and global attention.

Another possibility is to approximate the softmax with a linear operation. In this case, reordering the matrix operations avoids computing the huge QK^T matrix as long as we have a correct approximation using the auxiliary functions f and g:

$$\text{softmax}(\underbrace{QK^T}_{n\times n})\underbrace{V}_{n\times d'} \approx \underbrace{f(Q)}_{n\times d}\underbrace{g(K)V}_{d\times d'}$$

Katharopoulos et al. (2020) project the queries and keys to a different vector space in which the dot product is an approximation of the softmax. Peng et al. (2021a) use random projections which are known to approximate the Gaussian kernel. Performers (Choromanski et al. 2020) follow the same idea, but with only positive orthogonal features which reduce the unstability of the previous approach. Wang et al. (2020) propose the Linformer, which factorizes the attention into low-rank matrices. Another solution is the Nyström approximation, computed by Xiong et al. (2021) by sampling the queries and the keys.

Long-range dependencies can be achieved by iterating multiple smaller attention layers with different contexts every time. Jaegle et al. (2021) use a drastic compression step and then iterate over multiple cross-attention layers. Dai et al. (2019b) compute the attention over local chunks but recursively which brings a larger context than the standard approach. Similarly, Rae et al. (2020) use a memory bank for caching previous states and compute the attention using the local window and this memory tokens.

For videos, Arnab et al. (2021) investigate different factorizations of the attention. Attention can be decoupled in an intra-frame spatial attention and inter-frame temporal attention (Liang et al. 2022). Local spatio-temporal windows are another solution (Liu et al. 2022).

Finally, an alternative to attention layers that allows for long-range dependencies without the cost is LambdaNetworks (Bello 2021). In the attention linguo, they project the keys to a fixed size context with a preprocessing softmax operation.

The interested reader will find in Tay et al. (2020a) a comprehensive survey of efficient Transformers and attention layers.

IV.2 Patch-Based Stochastic Attention

IV.2.1 Full Attention

We recall the standard definition of dot-product attention introduced in chapter II. We refer to it as the *Full Attention layer* (FA layer). The attention is defined for a set of m *queries* each in $\mathbb{R}^d$ so that $Q \in \mathbb{R}^{m\times d}$. We also have a set of n *keys*,

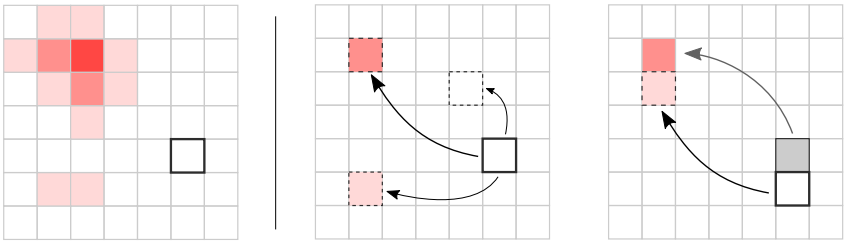


Figure IV.2: Illustration of patch NN search. (a) Full Attention computes a complete attention matrix but many elements have negligible weight. (b) Patch-Based Stochastic Attention only probes randomly a few elements. (c) Good matches are propagated to neighbors

vectors of $\mathbb{R}^d$, with the matrix $K \in \mathbb{R}^{n \times d}$ the collection of all these vectors. The last element is the set of *values* $V \in \mathbb{R}^{n \times d'}$. Given these, the attention is the following weighted sum:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T)V \tag{IV.1}$$

where the softmax function is applied to each row of the matrix QK^T. Here, we concentrate on the case where these contain image patches. For simplicity, in all that follows, we will consider that $m = n$. However, we note that our approach is equally applicable in the general case where $m \neq n$.

The dot-product attention in Equation (IV.1) requires the computation of the full matrix QK^T with n^2 entries. This results in a computational complexity of $\mathcal{O}(n^2 d)$, and a memory complexity of $\mathcal{O}(n^2)$, n being the input size *e.g.* sequence length or number of pixels. For 1-dimensional vectors, this can be implemented with simple matrix multiplications. For 2-dimensional vectors *i.e.* patches, dot products can be computed as 2D convolutions as remarked by Li and Wand (2016).

This memory requirement is the most problematic limitation of the FA layers. To address this problem, Yu et al. (2018) subsample the set of keys. Another approach, proposed by Liu et al. (2018a) for image restoration, is to compute the attention restricted to a local neighborhood. While such approximations are useful, they nevertheless rely on many pairwise distances to be computed. In practice, we remark that after the softmax operation in Equation (IV.1), *only a few elements actually matter*. One way of viewing the attention layer is as a "soft" NN search layer. Consequently, in order to limit algorithmic complexity, we propose to switch to a *sparse* layer, keeping only a single non-zero value in each row in the matrix QK^T, corresponding to the NN. The crux of the problem is now to solve the NN search quickly and with little memory overhead. For this purpose, we propose to employ an efficient ANN algorithm, designed specifically for images: the PatchMatch algorithm (Barnes and Shechtman 2010; Barnes et al. 2009).

IV.2.2 Patch-based Stochastic Attention Layer (PSAL)

Recall that our goal is to replace the traditional attention layer, which is cumbersome in terms of memory, with a more efficient approach, designed for images. As we have noted above, attention layers are closely related to the search of NNs. We start by defining the NN mapping ψ, between the query vectors and the key vectors:

$$\psi(i) = \underset{j \in \{1,\dots,n\}}{\arg\max} \langle Q_i, K_j \rangle \,, \tag{IV.2}$$

where Q_i is the ith line of the matrix Q, corresponding to the vector i, and likewise for K. In the attention literature, the dot product is commonly used to compare patches, but for more generality, we introduce a patch similarity function $s(Q_i, K_j)$, which is high when patches are similar.

Finally, we introduce the associated sparse matrix $A \in \mathbb{R}^{n \times n}$ defined as:

$$A_{i,j} = \begin{cases} 1 & \text{if} \quad \psi(i) = j \\ 0 & \text{otherwise} \end{cases} \,. \tag{IV.3}$$

Our definition of attention, which can be seen as a rewriting of Equation (IV.1), is simply

$$\text{Attention}(Q, K, V) = AV. \tag{IV.4}$$

The next step is to provide a fast and light way to approximate ψ. In the general case, this can be implemented using ANN algorithms, like kd-trees or Locality Sensitive Hashing as done by Kitaev et al. (2020). For images and image-like tensors such as feature maps, where vectors are patches, PatchMatch is an efficient alternative. It accelerates the search for NNs by drawing on a specific regularity property of images: the *shift map* (ie. the values $\psi(i) - i$) between NNs in different images is approximately piece-wise constant. Note that this implicitly requires a spatial organisation (1D, 2D, etc) of the data, which is the present case of images.

PatchMatch is an efficient, stochastic, algorithm for searching for ANNs of patches in images and videos, between a query image and a key image. PatchMatch starts out by randomly associating ANNs to the query patches. In general, these ANNs will be of poor quality, however from time to time a good association $\psi(i)$ will be found. The algorithm then attempts to propagate the shift $\psi(i) - i$ given by this ANN to other query patches in the spatial patch neighborhood of i, with the hypothesis that these shifts are piece-wise constant. This happens, for example, when a coherent object is found in both the query image and the key image.

After the random initialization, the PatchMatch algorithm relies on two alternating steps:

1. Propagation, in which good shifts are propagated to spatial neighbors

2. Random search for better ANNs for each query patch

Algorithm 4 Propagation step using Jump-Flooding

Require: queries Q, keys K, ANN field ψ
Ensure: Updated ANN field ψ

parfor $p = 1 \dots m$ **do**
  for $l = 1, 2, 4, 8$ **do**
    for δ in $[(-l, 0), (l, 0), (0, -l), (0, l)]$ **do** ▷ Look up, down, left, right
      $q \leftarrow \psi(p + \delta) - \delta$ ▷ Candidate position q
      if $s(Q_p, K_q) > s(Q_p, K_{\psi(p)})$ **then**
        $\psi(p) \leftarrow q$
      end if
    end for
  end for
end parfor

The random search is carried out by randomly looking in a window of decreasing size, around the current ANN. An illustration of the idea of PatchMatch can be seen in Figure IV.2.

A drawback of PatchMatch is that it is an iterative algorithm that is not naturally parallelizable, with the propagation step being inherently sequential, thus making it problematic for use in deep learning. However, we employ a semi-parallel approximation to this propagation step known as jump-flooding (Barnes et al. 2009; Rong and Tan 2006) described in Algorithm 4. A significant advantage of PatchMatch is that it keeps only the current ANN, which vastly reduces the memory requirements.

IV.2.3 Complexity

The computational complexity of our proposed PSAL is $\mathcal{O}(ndN \log(n))$ with N the number of iterations of propagation / random search. The derivations are the following: the PatchMatch algorithm consists of N iterations of propagation and random search for each of the n patches. Following the usual practice for random search, we sample 1 candidate in windows of geometrically decreasing size of ratio 0.5, leading to $\mathcal{O}(\log(n))$ candidates. For the propagation step, the parallel approach (Algorithm 4) uses $4 \times 4 = 16$ candidates. Finally, the cost of a similarity computation (dot product, cosine similarity, or ℓ_2 distance) between two patches has a complexity $\mathcal{O}(d)$ for d the dimension of the patch. Putting all together, we have a computational complexity of $\mathcal{O}(ndN \max(\log(n), 16))$. The memory complexity is simply $\mathcal{O}(n)$, as we only have to store the shift map ψ, our mapping for each patch. This is to be compared with the Full Attention layer, whose complexities are $\mathcal{O}(n^2 d)$ and $\mathcal{O}(n^2)$, respectively.

In particular, the memory complexity is linear with respect to the number of queries, while that of Full Attention is quadratic. This has important consequences, in particular on the maximum resolution of images that can be processed by deep

learning architectures which employ attention layers.

IV.2.4 Differentiability

Unfortunately PatchMatch, using a single NN, is not differentiable with respect to Q and K because of the argmax operator in equation (IV.2).

We propose to overcome this limitation by the use of multiple neighbors, and the adaptation of the softmax operator. Note that another work, by (Plötz and Roth 2018), also proposed the use of nearest neighbours. However, it is based on a continuous relaxation of the k-Nearest Neighbor (KNN) attention matrix, and thus has the same computational and memory complexity as Full Attention, which is precisely what we wish to avoid.

We now assume that we have access to multiple NNs instead of only one so that the support of our rows, or the shift-map, is now a set $\psi(i)$ with k elements. The matrix A writes:

$$A_{i,j} = \begin{cases} \frac{\exp(S_{ij})}{\sum_{j' \in \psi(i)} \exp(S_{ij'})} & \text{if} \quad j \in \psi(i) \\ 0 \quad \text{otherwise} \end{cases} . \tag{IV.5}$$

The matrix $S \in \mathbb{R}^{n \times n}$ is an intermediate matrix storing the similarities between elements of Q and K. We will give further details on how S is established later in the paper.

We propose two ways to construct A in a computationally- and memory-efficient way: one based on using several NNs, and one based on patch aggregation. Our theoretical findings are confirmed by the experiments (Section IV.3.3.3), in which the importance of the differentiability property will be shown.

We show that this restores differentiability with respect to Q and K as long as the number $k = |\psi(i)|$ of NNs is larger than 1.

IV.2.4.1 Derivatives

We give the derivatives with respect to all input variables in the case of scalar values *i.e.* $Q \in \mathbb{R}^{n \times 1}, K \in \mathbb{R}^{n \times 1}, V \in \mathbb{R}^{n \times 1}$. We have : $\forall i, j$:

$$\frac{\partial (AV)_i}{\partial V_j} = A_{ij} \tag{IV.6}$$

$$\frac{\partial (AV)_i}{\partial Q_j} = \mathbb{1}_{i=j} \sum_{k \in \psi(i)} A_{ik} V_k \left[\frac{\partial S_{ik}}{\partial Q_j} - \sum_{j' \in \psi(i)} A_{ij'} \frac{\partial S_{ij'}}{\partial Q_j} \right] \tag{IV.7}$$

$$\frac{\partial (AV)_i}{\partial K_j} = \mathbb{1}_{j \in \psi(i)} \frac{\partial S_{ij}}{\partial K_j} \sum_{k \in \psi(i)} A_{ik} \left[\mathbb{1}_{j=k} - A_{ik} \right] V_k \tag{IV.8}$$

The derivatives with respect to Q, K involve the derivatives of the similarity function used $S_{ij} = s(Q_i, K_j)$ with respect to its inputs but are of no difficulty

when s is the Euclidean distance or the dot product. In the case where $|\psi(i)| = 1$, the derivatives with respect to Q_j and K_j are equal to 0 for all j, confirming the need for multiple neighbors in our method.

IV.2.4.2 k-Nearest Neighbors differentiability

In order to overcome the aforementioned differentiability problem which arises when we have only a single NN, we propose to enrich the matrix S using several NNs for each patch, which can be done with a modified version of PatchMatch (Barnes and Shechtman 2010). In this case, each $\psi(i)$ is now a *set* of k NN correspondences. We start by redefining the matrix $S_{i,j}$ as:

$$S_{i,j} = \begin{cases} s(Q_i, K_j) & \text{if } j \in \psi(i) \\ -\infty & \text{otherwise.} \end{cases} \tag{IV.9}$$

Then by applying a softmax operation along the rows, we obtain the differentiable $\tilde{A}_t = \text{softmax}_t(S)$ with the same behavior than the original implementation. It can be seen that the Full Attention implementation corresponds to $\tilde{A}_t$ in the case where $k = N$. There is only a light computational overhead to manage the k nearest neighbors using a max heap and the associated memory extension. Finally PSAL-k has $\mathcal{O}(ndN \log(n) \log(k))$ computational complexity and $\mathcal{O}(nk)$ memory complexity.

IV.2.4.3 Patch aggregation differentiability

The second approach we propose for achieving differentiability/enriching the patch NNs is to perform spatial aggregation. Intuitively, we enrich the list of NNs for a given patch by using the NNs of the *spatial* neighbors of this patch. To put it more colloquially, the neighbor of my spatial neighbor is my neighbor. In this case, we redefine S in terms of a spatial neighbor i' and its patch-space neighbor j' as[1]:

$$S_{i,j} = \begin{cases} s(Q_{i'}, K_{j'}) & \text{if } \begin{cases} i' \in \mathcal{N}_i \text{ and } j' \in \psi(i') \\ \text{and } i' - i = j' - j \end{cases} \\ -\infty & \text{otherwise.} \end{cases} \tag{IV.10}$$

where $\mathcal{N}_i$ is the spatial patch neighborhood of i. The condition in Equation (IV.10) basically says that, for a patch i, we are analyzing its *spatial* neighbor i' and the NN of i', $\psi(i')$. We then check that the spatial shift between the patch i and i' is the same as the NN patches j and j'. The last condition is necessary to link j to j'. In practice, we choose the spatial neighborhood to be the patch neighborhood. This aggregation can be useful to other sparse attention layers. PSAL Agg. has $\mathcal{O}(p^2 n)$ memory complexity and $\mathcal{O}(p^2 n + ndN \log(n))$ computational complexity for a patch size of p.

[1] Note that this also impacts the definition of A in Equation (IV.3)

These previous two differentiability methods can be combined as desired. We present experiments in the next Section that show that without these approaches, networks have great difficulty learning, and produce poor results.

IV.2.5 Similarity function

One of the key components of the attention mechanism is the similarity function used to build the similarity matrix S . The original work of Vaswani et al. (2017) used the dot product with a scaling parameter, however other options such as the cosine similarity (Yu et al. 2018) are better suited in some situations. PSAL can be used with any similarity metric. This is important for providing a flexible attention layer, especially given than the similarity function can significatively impact the performance of a whole network. For instance, we found out that the dot product similarity performs poorly compared to the ℓ_2 distance in the reconstruction and colorization tasks (Sections IV.3.2 and IV.3.3). Linear Attention (Katharopoulos et al. 2020) and Reformer (Choromanski et al. 2020) are linear approximations and are thus limited in some cases.

IV.3 Results

We now present quantitative and qualitative results showing the advantages of our proposed attention layer. We compare our layer in 5 different situations:

1. We analyze the memory consumption of different attention layers. We observe in particular that PSAL requires orders of magnitude less memory than alternatives

2. Image reconstruction. We show that by replacing the FA layer with the proposed PSAL, the NN patches reconstruct an image well. This evaluation is motivated by the fact that the examples in the matrix V should reconstruct or approximate the initial queries

3. Image colorization. PSAL performs better than other attention layers in the context of guided image colorization, an image editing task for which attention is crucial

4. Image inpainting. We show that it is possible to replace a classical FA layer directly with a PSAL, without affecting the inpainting quality, allowing for inpainting of high-resolution images

5. Single-image super-resolution. Similarly to inpainting, classical FA layer can be replaced with PSAL without affecting the performance but allowing high resolution processing which is beneficial

We compare our work with Full Attention and other state-of-the-art attention layers: Local Attention (Parmar et al. 2018), Performer (Choromanski et al. 2020),

Attention Method	Mem. complexity	Memory (GB) for images of size			
		64x64	128x128	256x256	512x512
Full Attention	$\mathcal{O}(n^2)$	0.30	4.98	15.26	250.04
Local Attention	$\mathcal{O}(w^2 n)$	0.19	0.64	3.23	13.12
Linear Attention	$\mathcal{O}(n)^*$	0.11	0.46	1.85	7.47
Performer	$\mathcal{O}(nd \log d)^*$	0.71	2.84	11.55	17.68
Reformer	$\mathcal{O}(128n)^*$	0.47	1.86	7.42	21.47
PSAL 3	$\mathcal{O}(3n)$	0.01	0.01	0.04	0.18
PSAL Aggreg.	$\mathcal{O}(p^2 n)$	0.05	0.19	0.74	2.95

Table IV.1: Memory (mem.) (GB) required by the attention layer when the input size is increasing. n is the number of pixels. For Local Attention, the window size w is set to 50. For Performer, we use the recommended parameter $M = d \log d$. Tested in conditions with patch size $p = 7$ and 16 channels. PSAL and PSAL Aggreg. have a low enough memory footprint to make batches larger than 1 possible. * Linear Attention, Performer, and Reformer use 1D vectors, gathering patches and unrolling them consumes a significant amount of memory.

Reformer (Kitaev et al. 2020), and Linear Attention (Katharopoulos et al. 2020). We compare these with two differentiable PSAL approaches: PSAL 3 which uses k-NNs with $k = 3$ and PSAL with aggregation, (PSAL Aggreg.). The code for our PSAL layer is available at `https://github.com/ncherel/psal`.

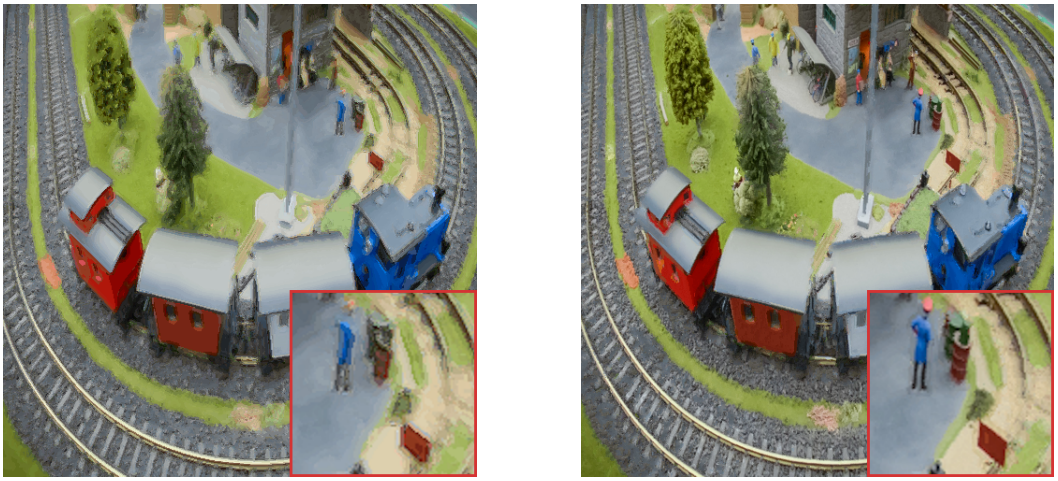
IV.3.1 Memory benchmark

We recall that one of our initial motivations is to develop an efficient attention layer that can be used in any situation. In particular, we designed our layer to not be memory-bounded when applying it to mid to large images. In this section, we measure the memory footprint of various attention layers for different feature maps size. We also compare the theoretical complexity with the true memory occupation. These results can be seen in Table IV.1. We see that our PSAL requires vastly less memory than FA and competing methods. For example, in the case of 512×512 images, FA requires 250GB, whereas PSAL 0.18GB or 2.95GB (PSAL 3 or PSAL Aggreg.).

IV.3.2 Image reconstruction task

We now compare the performances of PSAL and that of a FA layer on the task of image reconstruction using patches. The goal of this experiment is to check that using ANNs does not induce any loss of quality with respect to the "standard" attention. In the case of images, we can directly view this quality, contrary to the case of deep features.

We reconstruct an image by applying the attention layer using the ℓ_2 distance, Q the set of patches from image A, K the set of patches from image B and V the pixels of image B. Image A and B are taken from the same video sequence.



(a) Full Attention (b) PSAL

Figure IV.3: Full Attention (left) and PSAL (right) reconstruction using another frame of the same video. The memory constraints and the subsequent subsampling step make it impossible for the Full Attention to capture all details

Attention Method	ℓ_2 loss
Full Attention	0.0022
Local Attention	0.0119
Linear Attention	0.0576
PSAL 3	0.0011
PSAL Aggreg.	0.0008

Table IV.2: Reconstruction error of PSAL layers vs other attention layers.

We first show that PSAL can efficiently appproximate Full Attention on this reconstruction task, using patches of size 7×7. Table IV.3 indicates that the performances of PSAL and Full Attention (with stride 1) are equivalent but limits the size of the input to 64×64. As the resolution is increased, an approximation has to be used for Full Attention. Among the different options, we decide to keep all the queries Q but the keys K are obtained using a stride to decrease the memory pressure. We observe better results for PSAL in these cases.

We then consider a realistic case where images are of size 512×512. Quantitative results for 30 pairs of images 10 frames apart are shown in Table IV.2. The metrics confirm that the naive approximation of Full Attention via subsampling is more damageable to the performance than the approximation of PSAL. For Linear Attention, the approximation proposed by Katharopoulos et al. (2020) is an approximation of the dot product which is not well-suited in this case. Similarly Local Attention performs poorly as the displacement between the reference and the image to reconstruct is often larger than the local window.

Figure IV.3 shows the results of the image reconstruction task. We observe that

Image width	Full Attention		PSAL 3
	Stride	ℓ_2 error	ℓ_2 error
512	10	0.0022	0.0011
256	5	0.0031	0.0017
128	2	0.0038	0.0029
64	1	0.0048	0.0048

Table IV.3: Reconstruction error of PSAL layer vs Full Attention.

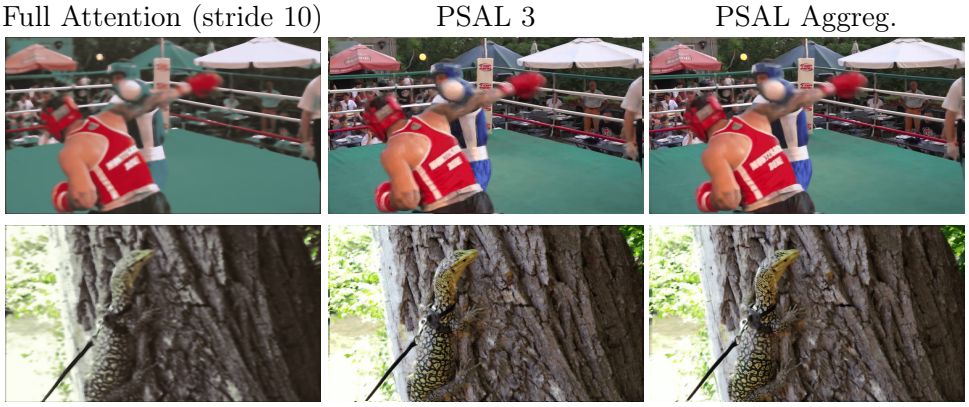


Figure IV.4: Results on the reconstruction task. Best seen with zoom-in.

not only does PSAL maintain good reconstruction, it in fact performs better than FA, which is not able to reconstruct fine details (see the zooms in the red squares on the bottom right of the images). This is a result of the stride induced by strong memory requirements. On the other hand, PSAL reconstruction has crisp details. This ability of PSAL to recover details is interesting when attention layers are used for tasks such as single image super-resolution (Parmar et al. 2018).

IV.3.3 Guided Colorization

We evaluate PSAL on the task of guided image colorization. Given a grayscale image and a colorful reference image, we train a network to recover the color information. We use as simple a network as possible, to isolate the contribution of the attention layer. This architecture can be seen in Figure IV.6. Because we are using similar images, but not identical, attention is a crucial component to identify the best regions from which to copy the colors. Note that the goal of this experiment is not to get good colorization results, but to compare attention layers in a challenging but realistic task. In this setting, the limitations and advantages of each method are easier to interpret than in a large neural network where attention layers are added for unclear reasons.

We compare our results with the other attention models by keeping the same architecture, but changing only the attention layer. We use images of size 256x256

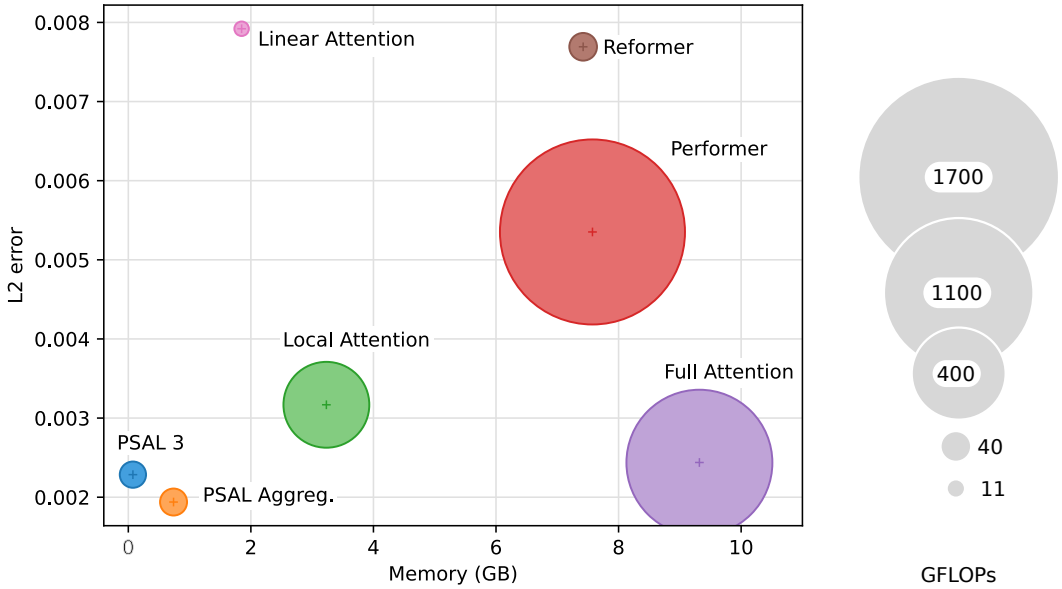


Figure IV.5: Performance vs computation constraints (memory and GFLOPs) in the colorization task. Radius is proportional to FLOPs for a single forward pass (smaller is better). Full Attention (Vaswani et al. 2017) performs well at the cost of high memory and many GFLOPs. Local Attention (Parmar et al. 2018) is an efficient approximation of Full Attention with a limited drop in performance. Performer (Choromanski et al. 2020) does not perform well. Reformer (Kitaev et al. 2020) and linear Attention (Katharopoulos et al. 2020) are linear approximations which are not adapted to this problem. PSAL 3 and PSAL Aggreg. have better performance, largely reduced memory usage, and require less FLOPs than alternatives. *Full Attention still requires a subsampling step to fit into GPU memory.

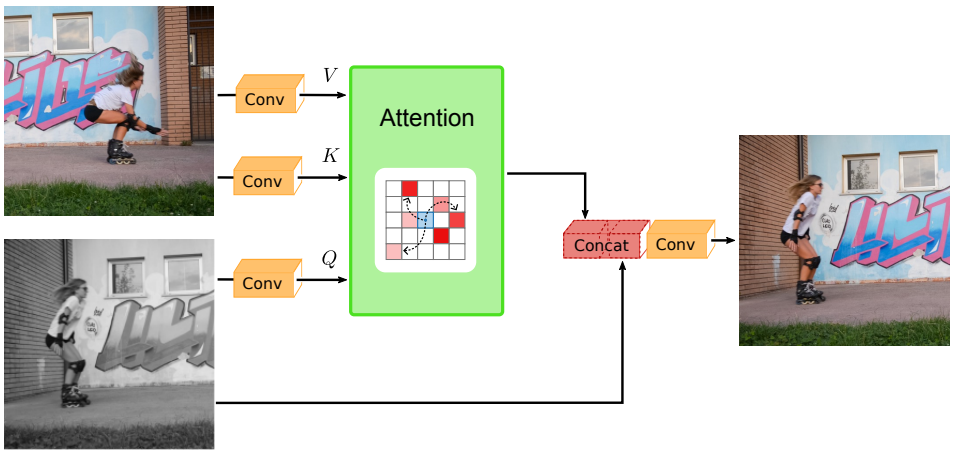


Figure IV.6: Architecture for our colorization network. We have used as simple an architecture as possible to isolate the contribution of the attention layer.

(a) Ground truth (b) Full Attention (c) Local Attention

(d) Performer (e) PSAL 3 (f) PSAL Aggreg.

(g) Ground truth (h) Full Attention (i) Local Attention

(j) Performer (k) PSAL 3 (l) PSAL Aggreg.

Figure IV.7: More results on the colorization task. Performer and Full Attention produce bland results. Because of large displacements between frames local attention is not enough to recover the true colors resulting in artifacts. PSAL 3 and PSAL Aggreg. have good results despite some wrong matches.

which are large by feature map standards but low resolution in the modern context. If we plot the ℓ_2 error as a function of the memory usage, and resize the points proportionally to the number of Floating Point OPerations (FLOPs), (Figure IV.5), it is clear that PSAL does not trade performance for memory or computations, and performs favorably against all other considered methods. Full Attention has close results but is limited by memory constraints: to fit into memory a subsampling step (of stride 3) is necessary which limits the set of keys considered. For Local Attention (Parmar et al. 2018), the model performs well for frames with little or no displacement but degrades abruptly when distant neighbors are required. PSAL 3 reaches good results but may sometimes match patches incorrectly. PSAL Aggreg. helps with this aspect, the aggregation step helps smoothing out irregularities. As in

Attention Method	ℓ_2 error
PSAL 1	0.00832
PSAL 3	0.00228
PSAL 7	0.00234
PSAL 15	0.00237
PSAL 31	0.00232
PSAL1 + Aggreg.	**0.00194**

Table IV.4: Colorization error of PSAL layers with different parameters. PSAL-1 is not fully differentiable which limits learning and performance. These experiments confirm that the solutions we proposed in Section IV.2.4 do indeed allow for efficient learning

the case of image reconstruction (Section IV.3.2), the choice of either ℓ_2 distance or dot product when comparing patches has a significant impact on performance. Our method is compatible with any metric. For FA and Local Attention (Parmar et al. 2018), we replaced the dot product with a ℓ_2 operation. For the Performer, which is based on softmax approximations of the dot product, we left them as described by their authors, since there is no obvious way to adapt their algorithm to the ℓ_2 case. They indeed produce poorer results generally. Reformer is very similar to our solution by using LSH to efficiently identify the nearest neighbors. For cross-attention, the proposed implementation of Kitaev et al. (2020) is far from perfect as it may create buckets containing only keys or queries, hence it has difficulty matching neighbors together. We also compare our method to a linear approximation of the attention : Linear Attention (Katharopoulos et al. 2020). In this case, we observe that non linearity is needed and the approximation is thus severely hurting the performance.

Figure IV.7 shows the visual results. We observe that FA and Performer produce faded results, while Local Attention has visible square artifacts because of local windows. PSAL Aggreg. produces the best results in this case.

IV.3.3.1 Training details

The colorization network is minimal, only containing: a single layer of 3x3 convolutions (16 output channels), the attention layer, and another 3x3 convolution (3 output channels). The goal is to put the emphasis on the attention layer and not on the problem of colorization, which is complex with an extensive literature. We add a residual connection so that the attention layer only has to provide color information.

Training is done on DAVIS dataset (Perazzi et al. 2016): train + test-dev. Testing is done on test-challenge. Images are resized to 256x256. The patch size is set to 7. We use the Adam optimizer with a learning rate of 0.001 for 200k iterations and a batch size of 1.

Similarity function	ℓ_2 loss
Dot product $s(q,k) = \langle q, k \rangle$	0.0064
ℓ_2 distance $s(q,k) = -\|q-k\|_2^2$	0.0024

Table IV.5: ℓ_2 vs dot product colorization results when using Full Attention. The choice of similarity metric is very important for the task. We see that ℓ_2 performs much better.

Attention Method	ℓ_1 loss ↓	ℓ_2 loss ↓	PSNR (dB) ↑	TV loss ↓	SSIM ↑
ContextualAttention*	11.8%	3.6%	16.4	**6.6%**	53.7
PSAL (ours)	**11.6%**	**3.6%**	**16.6**	6.9%	**54.1**

Table IV.6: Average inpainting metrics on Places2 validation set. Quantitative measures show similar performance using our layer with reduced memory requirements. * retrained.

IV.3.3.2 ℓ_2 vs dot product

For image colorization, we observed a large discrepancy between methods employing an ℓ_2 distance and a dot product similarity measure in the attention *e.g.* $s(q,k) = \langle q, k \rangle$ vs. $s(q,k) = -\|q-k\|_2^2$. Indeed, all methods using an ℓ_2 distance (Full Attention, Local Attention, PSAL) perform better than the methods based on dot product (Performer, Reformer, Linear Attention) in Figure IV.5. Table IV.5 shows this gap for the Full Attention layer for the same hyper parameters.

We hypothesize that in this bare-bones experiment without additional layers and especially normalization layers, the dot product is not well adapted. In fact, when comparing patches with the dot product similarity, the average value of a patch (query or key) does not directly intervene compared to the case of the ℓ_2 norm. It is then not surprising that the similarity between patches with very different average values can be high, which is often undesirable in traditional patch-based methods.

IV.3.3.3 Differentiability and neighbors

We now show experimental evidence that our proposed strategies for PSAL's differentiability are effective and are crucial for end-to-end training. Looking at Table IV.4, we see a large performance gap between PSAL-1 and PSAL-3/PSAL Aggreg. PSAL-1 is indeed not fully differentiable with respect to all its parameters and cannot learn the projection matrices Q and K. The attention does not help in this case and performances are poor. The performance does not improve beyond $k \geq 3$. PSAL with aggregration uses more neighbors and aggregates values differently than PSAL-k. We often use a number of neighbors of form $2^l - 1$, this comes from our internal representation of the set of neighbors as a binary tree of depth l.

PSAL-3, on the other hand, employs a 3-NN PSAL layer which makes optimization possible. Similarly, PSAL Aggreg. performs extremely well. This experiment confirms that our two approaches for making PSAL differentiable are effective. This

also shows that the adaptation of PatchMatch to feature maps is not as straightforward as it looks.

IV.3.4 Image inpainting

A prominent field of application of attention models is image inpainting. This is the process of automatically filling in unknown or damaged regions in an image. Deep learning inpainting algorithms are able to inpaint semantic objects, such as positioning an eye in a face. However, one of the blindspots of such approaches to image inpainting is the correct reconstruction of textures and fine details. It turns out that this problem is, in turn, well addressed by using patch-based approaches (Criminisi et al. 2004; Wexler et al. 2007). Thus, the ideal inpainting method would be able to unite the strengths of both deep learning and patch-based approaches in a single algorithm. This has motivated the introduction of attention layers in deep inpainting networks.

Recently, Yu et al. (2018) have introduced an attention layer with great success in their inpainting network, which we refer to as ContextualAttention (CA). After a first coarse inpainting, the image is refined flowing into 2 different branches: a fully convolutional network, and an attention-based network. The outputs are then merged. At its core, CA is a Full Attention layer based on 3x3 patches in feature maps. Unfortunately, even with mid-level features, the remaining spatial size is too large to avoid a memory explosion especially during training. Yu *et al.* limit the number of patches to be computed using a downsampling scheme. Once again, this illustrates the practical need for an attention layer which scales to large images. Given the setting, the proposed PSAL is a good fit in the network. We directly replace the FA layer with our PSAL 3. We use a patch size of 7 for PSAL, which is equivalent to a patch size of 3 plus a downsampling with a factor of 2 as used by CA.

We compare both approaches on different metrics: PSNR, SSIM, as well as Total Variation. We include the Total Variation (TV) loss which measures the smoothness of the inpainted image and does not depend on the ground truth:

$$TV(x) = \sum_{i,j} |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|$$

The quantitative results shows no significant differences between PSAL and ContextualAttention for different metrics (Table IV.6). This confirms that the PSAL can indeed replace the CA layer, with no quality loss, but with a great reduction in memory requirements.

Note that we do not need to compare with other attention approaches, since our goal here is not to improve the quality of inpainting, but rather to show that our PSAL can be easily inserted into any existing architecture, without any loss in quality, while greatly reducing memory requirements (which we showed in Table IV.1).

Figure IV.8: Here, we show the results of the ContextualAttention (Yu et al. 2018) and our version of the algorithm where the FA layer is replaced with a PSAL. We observe that the results are of very similar quality, validating the direct and straightforward replacement of FA with PSAL.

IV.3.4.1 Inpainting results

For completeness, we produce visual comparisons of inpainting results. Figure IV.8 shows very similar results between our method and the one from Yu et al. 2018. This is exactly what we aimed for, since we want to achieve the same inpainting results with our more memory efficient PSAL. This is reflected by close quantitative inpainting metrics.

IV.3.4.2 Training

We retrain Contextual Attention (Yu et al. 2018) using the same hyper-parameters than the authors and an implementation by Du Ang `https://github.com/daa233/generative-inpainting-pytorch`. Specifically, we train our networks for 800k iterations with a batch size of 16 on Places2.

For PSAL, we train for the same number of iterations. We use a patch size of 7, remove the downsampling step in the attention, reconstruct only from the central pixel, and use 5 iterations of PatchMatch.

IV.3.5 High resolution inpainting

Finally, in Figure IV.9 we show that with PSAL we can inpaint high resolution images, which is the initial goal that motivates this work (extending attention layers to larger resolution images). The low memory requirements make it possible to handle images of resolution up to 3300x3300 on a NVIDIA GTX 1080 Ti with 11 GB of memory. Processing such a large image using the FA layer without subsampling would require more than 1000 GB of memory.

The images are processed at their native resolutions without memory-saving tricks. The network is still limited by its receptive field which makes it only possible to fill small holes (128x128). Note that the maximum size of the occlusions/holes

Attention Method	Zoom x2	Zoom x3	Zoom x4
Cross-Scale Attention	33.383	29.123	27.288
PSAL	33.375	29.112	27.184

Table IV.7: For single-image super-resolution, Cross-Scale attention (Mei et al. 2020) can be efficiently approximated with PSAL as indicated by similar PSNR scores on the Urban 100 dataset.

(128×128) is imposed by the network architecture of ContextualAttention. We could certainly create another architecture using PSAL for larger occlusion sizes, but to be fair to the original work we kept the same sizes.

IV.3.6 Super-Resolution

Attention can also be very useful for the super-resolution task (Liu et al. 2018a). Patch recurrence in natural images at different scales have been identified as a strong prior for super resolution, providing rich information (Glasner et al. 2009; Michaeli and Irani 2014). Recent adaptations to attention with In-Scale attention and Cross-Scale Attention by Mei et al. (2020) confirm these findings for deep learning.

Recurring patches naturally occur at long range due to perspective, thus limiting the usefulness of local or restrained attention. Looking at the work of Mei *et al.*, we replace the Full Attention layer which they use in the Cross-Scale branch by PSAL. We evaluate this replacement on the task of single-image super-resolution.

Our results (Table IV.7) on the Urban 100 dataset for different zoom factors indicate that we can use PSAL to efficiently approximate the Cross-Scale attention which benefits from long range dependencies. The In-Scale attention branch is more efficiently approximated with a local attention. We test our hypothesis by keeping the same weights and changing only the Cross-Scale attention layer. Similar performance is reached using only a fraction of the memory (Table IV.7), this is of high interest for handling images with repetitive structures at a distance *i.e.* at high resolution.

Early works on patch recurrence (Glasner et al. 2009; Michaeli and Irani 2014) showed that the benefits of Cross-Scale are most spectacular at very large resolutions and zoom levels, which cannot be achieved by full attention implementations such as the one in Mei et al. 2020 due to GPU memory limitations. By replacing this implementation with PSAL in the Cross-Scale branch we expect to be able to retrain the architecture for much larger zoom levels and image sizes, thus showing the full potential of patch recurrence. This will be the subject of future research.

IV.4 Approximation in pretrained networks

One advantage of our attention layer is that it can replace the attention layers in already trained models. More generally, because PSAL-k is an approximation of Full Attention, we can replace attention layers, keeping similar results at a fraction of the

Figure IV.9: A 2700x3300 image inpainted using PSAL. The initial occlusions are outlined in green. Original picture by Didier Descouens - Licensed under CC BY-SA 4.0

Figure IV.10: A 3600x2700 image inpainted using PSAL. The initial occlusions are outlined in green. Original picture by MOSSOT - Licensed under CC BY-SA 3.0

original computational cost. This is particularly useful when performing inference on large inputs.

Recall that the attention can be written with a temperature coefficient T:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{T}\right)V \tag{IV.11}$$

This temperature coefficient controls the entropy of the attention matrix by assimilating the rows as probabilities. In the extremes, we have $\lim_{T\to+\infty} \text{softmax}\left(\frac{QK^T}{T}\right)_i = \frac{1}{n}$ which is simply a uniform aggregation. We also call this regime the "high entropy" regime since the entropy of the attention matrix is high. At the other extreme, $\lim_{T\to 0} \text{softmax}\left(\frac{QK^T}{T}\right) = \psi^\star$ for which PSAL-1 is an approximation. This is "low entropy". We therefore see that our approximation is mainly valid in the case where the attention distribution has low entropy. The often recommended value for T is $\sqrt{d}$ where d is the dimension of the vectors.

In the following subsections, we experiment with different values of T to understand in which regime the network operates and understand if our approximation would be detrimental to the performances.

We only test this on *already trained* networks. This does not indicate that our PSAL layer wouldn't work when retrained from scratch. Indeed it is expected that

Attention during training	Attention during test	ℓ_2 loss	Test GFLOPs	Test Memory (GB)
PSAL 1	PSAL 1	0.0083	30	0.05
PSAL Aggreg.	PSAL Aggreg.	0.0019	37	0.74
	PSAL 1	0.0023	30	0.05
PSAL 3	PSAL 3	0.0023	36	0.08
	PSAL 1	0.0023	30	0.05
Full Attention	Full Attention	0.0024	1173	9.32
	PSAL 1	0.0024	30	0.045
Local Attention	Local Attention	0.0032	385	3.23
	PSAL 1	0.0035	30	0.045

Table IV.8: Colorization performance when using weights from pretrained models but replacing the attention layer with PSAL 1 at test time. PSAL 1 can replace the attention layers in trained networks with similar performance but a significant reduction in FLOPs and memory usage.

the weights would be different if we enforce sparsity in the attention map during the training part.

For Natural Language Processing, the Transformer framework has been central to the recent breakthroughs. However we refrain from testing this additional modality, while the search for very long context windows have started to reach the image levels and could benefit from the limited memory impact of PSAL, the PatchMatch hypothesis is also weakened because text is 1 dimensional and therefore benefits less from the spatial propagation step.

One of our hypothesis for our approximation was that it was possible to approximate the attention using only the nearest neighbors. In this section, we investigate if this is true for different types of common networks.

IV.4.1 Colorization

We show this effect on the colorization task using PSAL-1 as a replacement for other attention layers at inference time. While the parameters of PSAL-1 are not all trainable, we can still use it if we do not need to train the network. PSAL-1 is still a fast, light, and good approximation of Full Attention or other PSAL-k layers. In the colorization task, we trained and froze several different networks, and then we switched attention layers to PSAL-1. We observed very good results with this approach. Table IV.8 shows for instance that Full Attention can be approximated with no drop in performance but with 40x less computations and 225x less memory.

IV.4.2 Inpainting with ContextualAttention

Here, contrary to the experiment of subsection IV.3.4, we compare the results replacing the ContextualAttention layer with PSAL in the already trained network of Yu et al. (2018). We evaluate the results using the same validation set than

Attention Method	ℓ_1 loss	ℓ_2 loss	PSNR	TV loss	SSIM
ContextualAttention (2018)	10.8%	**3.2%**	**17.1%**	7.3%	56.0%
PSAL 3 (ours)	**10.7%**	**3.2%**	**17.1%**	**7.1%**	**56.4%**

Table IV.9: Average inpainting metrics on Places2 validation set. Quantitative measures show similar performance using our layer with reduced memory requirements

previously. We report the results in Table IV.9. Once again, we find that PSAL correctly approximate the ContextualAttention layer at a fraction of the memory requirements.

In this case, we recall that the ContextualAttention was computed using the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{1}{0.1}\frac{\langle q, k\rangle}{\|q\| \cdot \|k\|}\right) V$$

On top of using cosine similarity instead of simply scaled dot product, the temperature is already low: 0.1, biasing the attention to a handful of points.

IV.4.3 Transformer for classification

For classification networks, we look at the Vision Transformer (ViT) from Dosovitskiy et al. (2020). The Vision Transformer performs the classification on images from the ImageNet dataset, non-overlapping patches of size 16x16 are extracted from the image and then go through a series of Transformer layers: attention layers, multi-layer perceptrons, normalization layers. In this context, after extraction, the number of patches in the attention layer is 197 which is very low compared to applying attention of size 128x128 (still small) but for which around 16k patches are used.

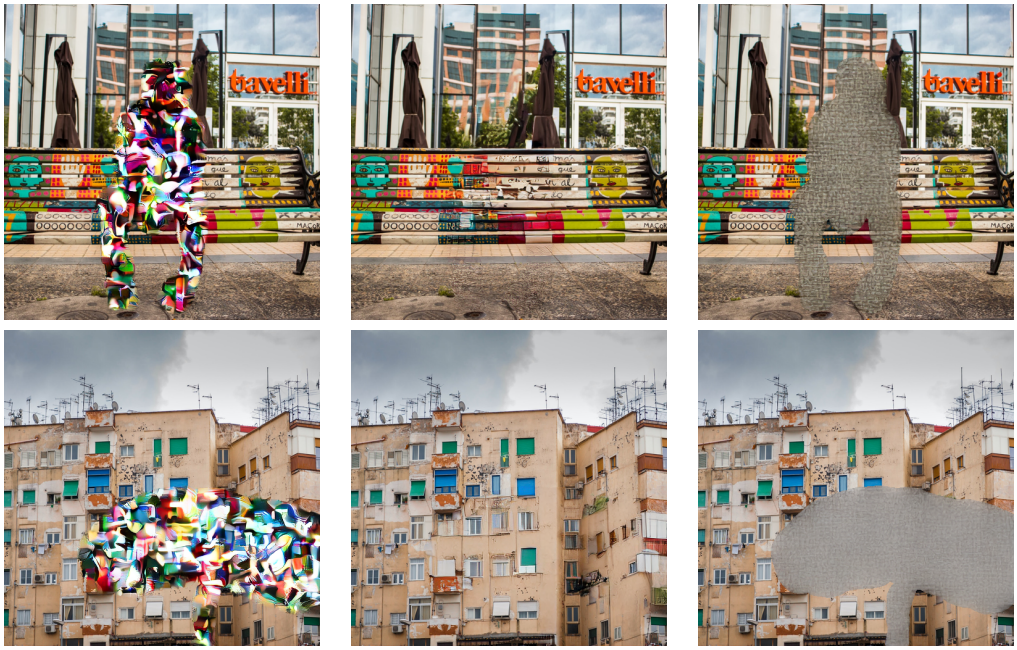
We select a small test set of classification images (4000 images from ImageNet) and test different values of T on ViT-B. It is also possible that the entropy changes through the layers for instance with first layers being more sparse and later layers less so. Recall that the attention is given by:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{T}\right) V$$

We change the temperature and report the classification accuracy in Table IV.10. The quantitative evaluations show a sharp decline in accuracy when using a different temperature coefficient than the training one. In our case, we are interested in the low temperature value, and we see that the results are actually quite bad. In fact, for classification, it could be expected that the attention is not sparse, even more so that there are not many patches in total.

Temperature	1	2	4	8	16	32	64
Top-1 accuracy	0.0100	0.0760	0.6640	**0.8098**	0.6850	0.2435	0.0693

Table IV.10: Changing the temperature coefficient in an already trained classification network.



$T = 0.1$ $T = 1$ $T = 100$

Figure IV.11: Inpainting results in diffusion models when changing the temperature in all the attention layers. We cannot replace the attention by our approximation in these cases because our approximation would be close to the case $T = 0.1$.

IV.4.4 Stable Diffusion

For diffusion models, we test changing the temperatures in the inpainting network of Stable Diffusion (Rombach et al. 2022). Their network is large UNet with attention layers at different positions in the network. We compare the results using different temperatures coefficients through the diffusion process; we use the same temperature coefficient despite the attention layers having different input resolutions. In the baseline, the Full Attention adds a temperature coefficient scaled to the square root of the dimension of the features, thus providing an adaptive temperature based on the feature depth.

For these results, we do not have a quantitative evaluation but a purely qualitative one, that should be enough to convey the main idea. The results are shown in Figure IV.11 and show that in this context, our method cannot be used. When using a low temperature, severe artifacts are appearing. On the other hand, when increasing the temperature, we see that the output lacks details.

IV.4.5 Conclusion about pretrained networks

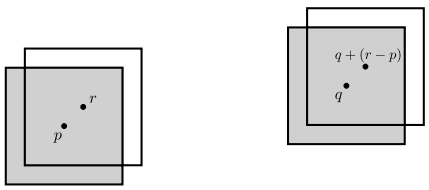
Among the 4 different settings tested: image classification, guided colorization, image inpainting with standard model, image inpainting with a diffusion model, we found that only the guided colorization and the image inpainting with the standard model actually worked with PSAL. Our hypothesis may thus only work for a few networks, it is also possible that our method work better for high resolution images than low resolution images, as relying on a few neighbors is even more true when the number of points grows. Note that we have done these experiments using trained networks, it is possible that retraining with PSAL from the start could work just as well.

IV.5 Self-similarity and the PatchMatch hypothesis

We have assumed in this chapter that the main PatchMatch hypothesis, the fact that natural images are coherent, can be extended without issues to feature maps. This is indeed a property that have been exploited by many authors for style transfer (Hamazaspyan and Navasardyan 2023; Samuth et al. 2022), stereo (Duggal et al. 2019; Wang et al. 2021a), or optical flow (Zheng et al. 2022b). In practice, for our application this self-similarity translates into a piecewise constant shift map: large regions of the image are repeated. We have several reasons to think that this property holds true for feature maps.

First of all, when using overlapping patches with the ℓ_2 distance, independently of the underlying signal, this hypothesis is likely to be respected because the differences are shared. This effect is stronger when using large patches. In fact, it is easy to see that the following is true for all patch positions p, q, r:

$$\|\Psi_p - \Psi_q\|^2 = \underbrace{\|(\Psi_p \cap \Psi_r) - (\Psi_q \cap \Psi_{q+(r-p)})\|^2}_{\text{common}} + \|(\Psi_p \setminus \Psi_r) - (\Psi_q \setminus \Psi_{q+(r-p)})\|^2$$



Using a neighbor patch of p for r, then the intersection is the dominant factor in the distance.

Secondly, features are non-linear transformations of an image but still look like the original image Figure IV.12.

IV.6 Limitations

Our approach is an approximation of the Full Attention based on the assumption that the attention is sparse and the number of points is too large to compute the

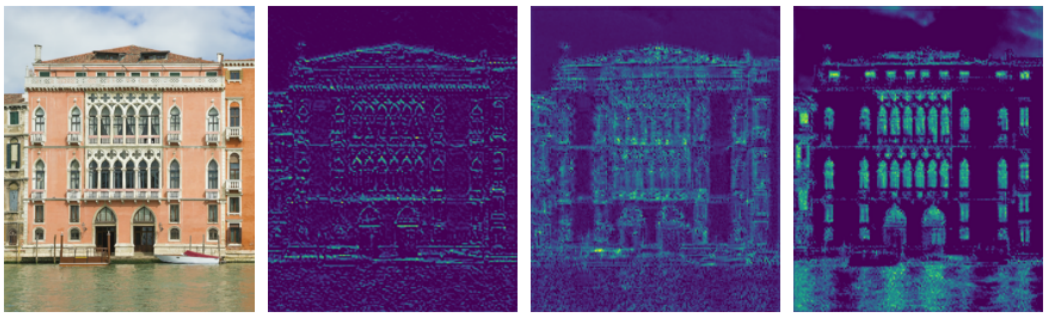


Figure IV.12: RGB image and some feature maps from the attention layer from ContextualAttention. The original image has been downscaled. After the convolutions, the image is still recognizable in the features which validate self-similarity in the features even though the features are activating for very different patterns from the original image.

standard attention due to memory constraints. Our limitations are then, by definition, when the attention is not sparse and memory is not an issue. If the attention is not sparse, then we leave a lot of the energy of the weighted sum on the side which is detrimental to the final output. We saw in section IV.4, that this can be the case for classification networks, where the temperature coefficient is large.

Second, our attention layer is not interesting if it is slower than Full Attention, this happens when the feature map is relatively small. In fact, PatchMatch is mainly a spatial approximation, and is of little use when the spatial size is low. In addition, it is often very likely than our previous assumption is also wrong: when there are not many points, attention is often not sparse. While our custom kernel has an advantageous theoretical complexity, it can be difficult to compete with naive methods with efficient implementations. Hardware optimizations and memory access are not always reflected in theoretical complexity analysis.

Finally, due to a sparse attention, we also have sparse gradients and while we have not encountered this problem in practice, learning with PSAL may be more difficult with only a few points receiving gradients. For disjoint query and key sets, the gradients with respect to K (Equation IV.8) show that a key may not receive a gradient from our attention layer if it is not among the nearest neighbors of any query point.

IV.7 Conclusion and perspectives

In this chapter, we have presented PSAL, an efficient patch-based stochastic attention layer that is not limited by GPU memory. We have showed that our layer gives a much lighter memory load, scaling to very high resolution images. This makes the processing of high resolution images possible with deep networks using attention, without any of the customary tricks currently used (subsampling, etc). Furthermore, new network architectures using attention mechanisms on low level

Figure IV.13: Failure cases of our method for reconstruction (top) and colorization (bottom).

features are now conceivable. We have demonstrated the use of PSAL in several tasks, showing in particular that high resolution image inpainting is achievable with PSAL. Some questions are still open:

Training of neural networks with low level attention layers. Current architectures are limited to using attention layers at very coarse resolution because of memory constraints. Our layer could finally enable architectures with low level attention layers almost at pixel level. This could improve the results as shown by Beyer et al. (2023); Dosovitskiy et al. (2020) for classification and restoration (Zamir et al. 2022) - operating closer to the pixel level often leads to better results.

Extensions to other data. We have proposed to use PSAL when considering 2D patches on images and features maps but the problem of attention is even greater in higher dimensions. Videos and 3D volumes have one more spatial dimension compared to images and thus the attention is even more computationally expensive. While we estimate that the naive Full Attention computations were limited to a resolution of 256x256 for a single image, this corresponds to a 32x32x32 volume in 3D. This dramatically limits the processing of realistic data in these cases. For instance, A high resolution (HD) video of 5 seconds at 24 frames per seconds is already of size 1280x720x120 which would require 1e7 more memory than the limit estimated previously (12e6 Gb of RAM). We see that an extension of PSAL would be welcome to drastically reduce the memory impact of attention layers. PatchMatch for videos have already been proposed by Newson et al. (2014), and a generalized formulation on any graph has been described by Ehret and Arias (2018). One difficulty remains however to fully leverage the GPU architecture, a parallel implementation has to be developed. In the case of videos, the temporal axis is more stable which allows for good approximations with temporal factorization (Liu et al. 2021a) or deformable

attention (Liu et al. 2020b).

Efficient attention layers through attention propagation. In our experiments, we have mainly studied the use of PSAL at a single position in the neural network. It is however common to build neural networks based on attention layers: Transformers. Sequential attention layers can remind us of traditional algorithms reusing the *shift map* multiple times in a row and refining it after the initialization (Newson et al. 2017) or chapter III. A similar mechanism could be proposed in the context of transformers: can we reuse the attention map between two consecutive layers? This could lead to great efficiency improvements. Intuitively, we could expect that the attention map is valid for several features even if splitting the features among different heads could limit the application. PSAL could be extended to reuse both coarser attention maps and previous attention maps at the same resolution (Samuth et al. 2022). This could also benefit diffusion models for instance which iterate the same network many times over slowly changing inputs.

V Internal image inpainting

Generative models based on deep learning have gained considerable momentum in recent years, since the introduction of Generative Adversarial Network (GAN) by Goodfellow et al. (2014). Subsequently, diffusion models (Ho et al. 2020) outperformed these approaches, and established themselves as the state-of-the-art for image synthesis and editing (Lugmayr et al. 2022; Saharia et al. 2022b). In addition, image *a priori* implicitly found by generative models have been exploited for solving inverse problems (Kawar et al. 2022a; Saharia et al. 2022b). However, these diffusion models are massive, possibly comprising up to several hundred million parameters, and require enormous computational resources, making training impossible for most users and questionable from an environmental point of view.

Here, we train a lightweight diffusion model on a single image without using an external database for image inpainting. This framework enables us to investigate the use of small diffusion models which is not the norm in the current literature. In contrast to the state-of-the-art, which proposes massive architectures, we propose very small-scale models. These models rely solely on convolution, subsampling/oversampling and nonlinearity operations, unlike most diffusion models which involve other more complex modules, such as attention modules.

Finally, this setting is also perfect for evaluating the stochastic nature of diffusion models. This ability to produce multiple results is then considered in the evaluation of our method, in particular, we look at the traditional ways of evaluating inpainting and their shortcomings. We also compare in detail the use of diffusion models with patch-based methods which we expect to perform similarly in this context but do not require a learning phase.

V.1 Related works

V.1.1 Diffusion models

The general framework of diffusion models has been presented in chapter II. Briefly, diffusion models are generative models which work on a long Markov chain of noisy examples. First introduced by Sohl-Dickstein et al. (2015), they have been popularized by Ho et al. (2020) recently by reframing it as a "denoising" problem. Multiple extensions have been proposed for conditional generations, from text (Rombach et

al. 2022) or images (Saharia et al. 2022a; b). To specialize the model, Ruiz et al. (2023) finetune a large neural network on a limited set of images. Diffusion models can also be applied to single image generation *i.e.* with limited data, as shown by Kulikov et al. (2023); Nikankin et al. (2023); Wang et al. (2022). Hao et al. (2023) propose a rectification step to enable planar texture synthesis with diffusion models.

V.1.2 Inpainting

As detailed in chapter II, the first approach to image inpainting has been proposed by Masnou and Morel (1998) using Partial Differential Equations (PDE). Bertalmio et al. (2000) then introduced the term of "inpainting" with a discrete smoothness energy term.

Then Efros and Leung (1999) have proposed an exemplar-barsed approach for texture synthesis and inpainting. The assumptions for patch-based texture inpainting can also be applied to non-stationary image inpainting (Bornard et al. 2002). With greedy algorithms, the order of inpainting is crucial and can be optimized (Criminisi et al. 2004). Classical approaches to inpainting with patches (Newson et al. 2017; Wexler et al. 2007) rely on the self-similarity of images to build a plausible solution. Hays and Efros (2007) have an external database to take patches from to complete images.

After the first work by Xie et al. (2012), the next neural network proposed for inpainting was by Pathak et al. (2016). They directly predict the pixel values for the missing region, the objective function uses both a reconstruction term and a discriminator term (Goodfellow et al. 2014). This is then refined by Iizuka et al. (2017), which use a global and a local discriminator. Yu et al. (2018) incorporate an attention layer to integrate a non-local component into their otherwise purely convolutional network. Liu et al. (2018b); Yu et al. (2019) propose to change the way the mask is used in deep networks. Suvorov et al. (2022) use a Fourier convolution module to avoid the use of attention layers while maintaining a global receptive field. Their method is particularly efficient for repetitive structures that have a specific Fourier transform. To account for the multiplicity of solutions in inpainting, Zheng et al. (2019) use a model that can efficiently produce multiple results. More recently, resource-intensive methods based on diffusion models have emerged, such as Repaint (Lugmayr et al. 2022) and Palette (Saharia et al. 2022b). It is possible to use unconditional diffusion models for inverse problems (Chung et al. 2022; Kawar et al. 2022a). These new diffusion models can be guided by text and mask (Xie et al. 2023), or another image (Yang et al. 2023a). RealFill (Tang et al. 2023) tries to find the true painted content using different images of the same scene.

Deep internal methods, *i.e.* methods based on neural networks trained on a single image, have been shown to work for image generation (Shaham et al. 2019) and also for image restoration (Ulyanov et al. 2018). For image inpainting, Wang et al. (2021c) have both an external and an internal network. Alkobi et al. (2023) trains a model similar to SinGAN for diverse image completion. Ruiz et al. (2023) Fine tune a large diffusion model on a few images for guided image generation.

For video inpainting, the high redundancy makes it easier to train a neural network for internal use. Zhang et al. (2019b) train a neural network on a single video, they use a reconstruction loss and a loss based on the optical flow. Ouyang et al. (2021) refine this idea but use an implicit flow that connects different frames. Ren et al. (2022) finetune a VAE with a discrete latent codebook on each video for better adequacy between training and testing.

V.2 Method

V.2.1 Diffusion models for image inpainting

We adapt the general framework of diffusion models (Ho et al. 2020; Sohl-Dickstein et al. 2015) to image inpainting. As presented in chapter II, diffusion models are generative models whose aim is to learn a distribution of natural images. They are based on a destructive *forward* process and a generative *backward* process. The forward process consists in transforming an input image, which is assumed to be a sample of the natural image distribution, into a white noise image, by adding low-amplitude noise over a number of iterations. The backward process performs the opposite steps: iterative (learned) denoisers of a sample initially drawn from Gaussian noise. If the denoiser is learned correctly, it can be used to go from a white noise to obtain a sample image. The reference image is taken to correspond to the image at time $t = 0$, and the final noisy image to that at time $t = T$. In this way, we establish a succession of images x_t with a noise level that increases with time.

Starting from the unknown distribution of natural images $q(x_0|y)$ conditioned on the observations y, we define a Markov chain $q(x_0|y), ..., q(x_T|y)$ with the forward transition kernel for $t \in [1, T]$:

$$q(x_t|x_{t-1}, y) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right), \tag{V.1}$$

with the variance schedule linearly increasing from $\beta_1 = 0.0001$ to $\beta_T = 0.02$. Recall the variables $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ which will come in handy because $q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$. In this case, the reverse process defines a conditional Gaussian distribution $p_\theta(x_t|x_{t+1}, y)$ of learnable mean and variance:

$$p_\theta\left(x_t|x_{t+1}, y\right) = \mathcal{N}\left(x_t; \mu_\theta(x_{t+1}, y, t), \sigma_t^2 \mathbf{I}\right)$$

We use a neural network to predict the conditional mean $\mu_\theta(x_{t+1}, y, t)$. We use a fixed variance schedule for the reverse process: $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$. The parameters θ are optimized to maximize a lower bound of the log-likelihood of the data in Ho et al. (2020).

In the case of inpainting, as mentioned above, we are interested in the data distribution conditioned on our observations *i.e.* the known region of the image. The observations are introduced in the network as incomplete images via concatenation in the input layer (similarly to Saharia et al. (2022b)). We also add the mask

information as input. We train our network by minimizing the reweighted ℓ_2 loss of the x_0-parametrization:

$$\mathbb{E}_{x_0,\varepsilon,t,M}\|M \odot (x_0 - f_\theta(x_t, y, t, M))\|_2^2, \tag{V.2}$$

where $M \sim \mathcal{P}_{\text{mask}}$ is a binary mask with 0 for the known region and 1 elsewhere, y is the masked image *i.e.* $y = x_0 \odot (1 - M)$, and x_t is obtained by the forward diffusion of x_0 by the following:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\varepsilon$$

where $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$.

The parameter t controls the noise level. Our network f_θ takes as input the noisy image x_t, the observation y, the mask M and the noise level at step t.

Inference. Once the denoiser f_θ has been trained, synthesis consists in drawing a random sample x_T and denoising it successively from $t = T, \dots, 0$. The mean of $p_\theta(x_{t-1}|x_t, t)$ is given by:

$$\mu_\theta(x_t, t) = \beta_t \frac{\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_t} \cdot \hat{x}_0(x_t, t) + (1 - \bar{\alpha}_{t-1}) \frac{\sqrt{\alpha_t}}{1 - \bar{\alpha}_t} \cdot x_t \tag{V.3}$$

Note that in this equation is involved the predicted clean image $\hat{x}_0(x_t, t)$. We postprocess it: we clip it to [-1,1] and reproject the observations at each diffusion step.

V.2.2 Lightweight network for image inpainting

Single image inpainting. In the case of single image inpainting, we are given a large image x_{test} and its corresponding mask M_{test}. When possible, we take sub-regions of the large image that do not intersect with the testing mask, and train on these smaller images: we generate synthetic masks, pick a random timestep, and sample noise to be added. After training, the model is then applied to the test data.

When the image is too small to take sub-regions, we still use the same training tactic but mask the input data with both the training mask and the testing mask. The loss for the pixels in the testing mask is zero.

Architecture. In practice, our conditional network takes as input the masked image y, the noisy image x_t, the inpainting binary mask M, and the temporal information t. For the network architecture, we have chosen to simplify the Ho et al. (2020) architecture as much as possible. It is therefore a UNet-type network without an attention layer (Figure V.1). The network entries are concatenated before the first convolution. In the case of single-image inpainting, it is not necessary to multiply the parameters, so we limit the number of channels to 32. In the end, our network has 160k parameters, compared to 450M for RePaint (Lugmayr et al. 2022).

Unlike Ho et al. (2020), we get better results by directly predicting the initial image x_0 rather than the noise ε. The temporal information of the t diffusion

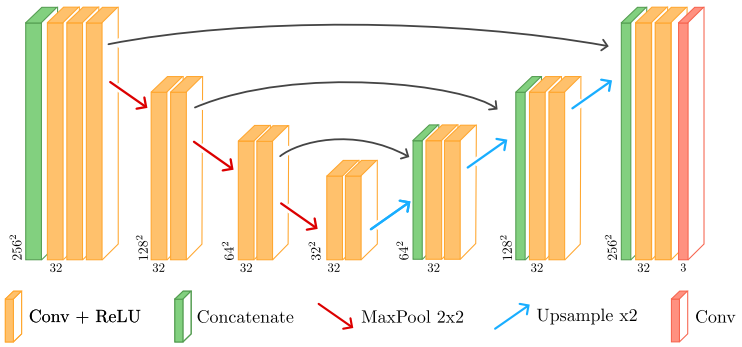
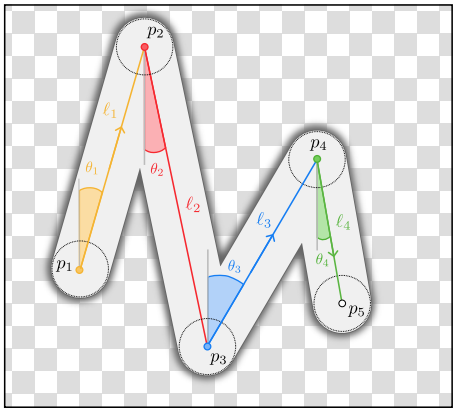


Figure V.1: Our UNet architecture is very simple, it uses multiple scales and simple operations. The number of channels is kept constant across the network. The different inputs (x_t, y, M, t) are simply concatenated before the first layer.



$p \sim \mathcal{U}(0, W) \times \mathcal{U}(0, H)$
for $i = 1, ..., N$ **do**
  $\theta \sim \mathcal{U}(0, \theta_{\max})$
  if i % 2 == 0 **then**
    $\theta \leftarrow 2\pi - \theta$ ▷ reverse mode
  end if
  $\ell \sim \mathcal{U}(0, \ell_{\max})$
  $p' \leftarrow p + \ell \cdot (\cos(\theta)\ \sin(\theta))^T$
  Draw line from p to p'
  $p \leftarrow p'$
end for

Figure V.2: Mask generation illustration and algorithm. Starting from a random position p in the image, a sequence of angles θ_i and lengths ℓ_i are used to draw a path with a given brush thickness.

step is usually introduced by a combination of sine and cosine. Encoding spatial positions using this technique facilitates the learning of high frequencies in a 3D image or scene (Tancik et al. 2020). We find that our simple formulation, similar to that used in some denoising networks, gives us equivalent results and simplifies the network. Training is performed by randomly drawing regions of size 256x256 from our reference image, which are then noised and masked. The M mask is generated synthetically according to a Yu et al. (2019) formulation. The algorithm is designed to mimick an eraser used for removing a stain with back and forth motions in a local region. Starting from a random starting point, a sequence of lengths and angles are drawn randomly and defines the brushstrokes. This is repeated multiple times. We show an illustration in Figure V.2 with the algorithm.

It is necessary to train the network to perform denoising for all time steps, so they are drawn uniformly in $[1, T]$.

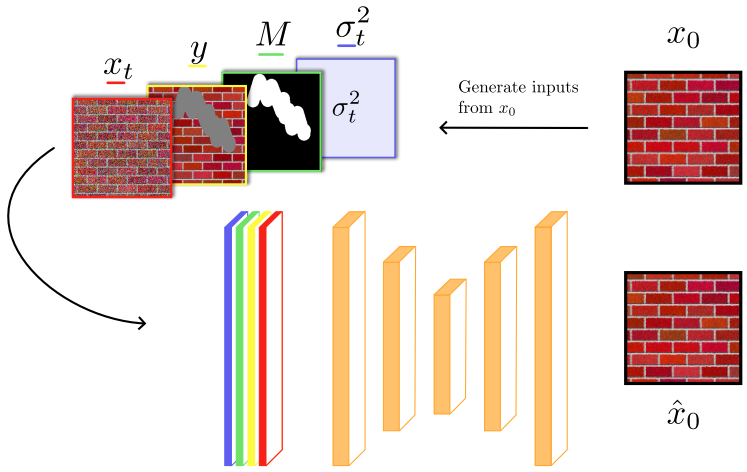


Figure V.3: Training loop for single image inpainting, starting from a sub-region x_0 of our full training image, batch samples are generated randomly (mask, timestep, noise). The network uses all those inputs for producing a denoised image $\hat{x}_0$.

The UNet architecture allows for an adaptive receptive field: depending on the task, the multi-scale nature of the network can be exploited or not. This is a clear advantage over patch-based methods, since the patch size and the number of scales in the Gaussian pyramid are two very important parameters that can be difficult to set.

V.3 Application to texture inpainting

V.3.1 Experimental details

We consider the application to texture inpainting. External information is not required for textures. They are stationary and training on a single image is sufficient. For training, we use images from the MacroTextures dataset (Lin et al. 2023). In these large images, routinely of size 1920x1080, we identify a central test region of size 256x256 that is never seen during training. We use the rest of the image for training. For training, we simply take a random crop x_0, a generated mask M, a random timestep t, and a noise ε and do a gradient step using the loss from Equation V.2. This training loop is summarized in Figure V.3.

Our neural network has only a few parameters (160k) compared to RePaint's 450M. Training takes 15 minutes on an NVidia V100. We train with the Adam optimizer with an initial learning rate of 1e-4 for 15k iterations. The batch size is 16.

We generate masks with random strokes, similar to Yu et al. (2018). These masks are of a reasonable size, neither too small nor too large, and come from the same generator for both training and testing. Inconsistencies between training and testing could introduce biases in the results.

We compare our frugal diffusion network to four other approaches. The first

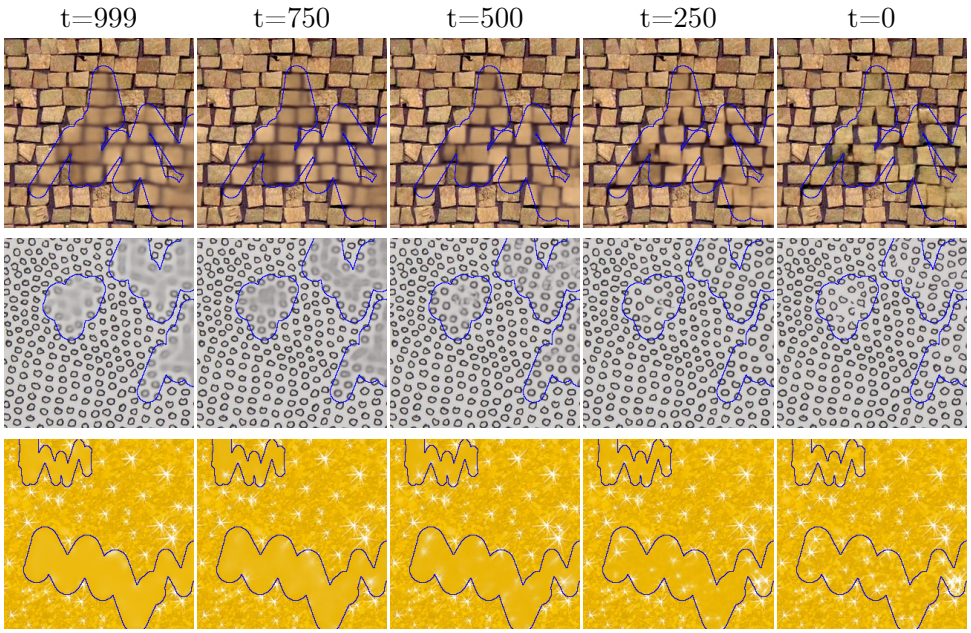


Figure V.4: Predicted $\hat{x}_0(t)$ during the inference process. The details are progressively added.

is a classical patch-based approach (Newson et al. 2017), which does not require a learning phase. The second is the inpainting formulation of RePaint (Lugmayr et al. 2022), a diffusion-based inpainting algorithm, which represents the state-of-the-art in inpainting. Their network has several hundred million parameters and requires tens of GPU days of training. The third approach is the previous state-of-the-art model from Yu et al. (2018): DeepFill, which uses an attention layer. Finally, we compare our diffusion inpainting with a standard learning approach on a **regression** problem.

Direct Comparison with Fixed Network Regression: This comparison is made *with fixed architecture*: we isolate exactly the contribution of diffusion modeling. For regression, the network is trained using only the ℓ_2 error of the image reconstruction on the masked area:

$$\mathcal{L}_\theta(x) = \|M \odot (x - f_\theta(y, M))\|^2 \tag{V.4}$$

Regression and diffusion networks are trained for the same amount of time (15 minutes). For the patch-based approach, while we test on a relatively small image, we include the full image to avoid restricting access to many different patches. This full image is the training data for our diffusion approaches.

V.3.2 Qualitative Results

Figure V.5 shows the results for the different models.

Visually, our method performs very well. It generates convincing results in all situations: for regular textures and very stochastic ones. On top of that, the stochastic nature of the diffusion process allows for sampling different results. We observe that the direct regression method leads to results that do not reproduce the stochastic variations of the images, unlike the diffusion or patch methods. For very regular images (*e.g.* with periodicities), the regression method is able to partially reconstruct the main structure, but the results remain blurred (Figure V.5, top line). This loss of detail and fuzziness is the predictable result of the averaging effect of the regression method. It's interesting to note that in this simple case and with a constant architecture, diffusion learning alone produces sharp, detailed results.

The inpainting results from the patch-based method of Newson et al. (2017) are also satisfactory. However, it has trouble handling some structures because the patches must be precisely aligned and we also see a little bit of blur in some situations.

DeepFill on the other hand is not as good as the other methods. While not blatantly wrong, the results have a few visual artifacts. It is neither trained on these textures, nor it has many parameters and attention layers, which can explain the lesser results.

It should also be noted that RePaint, although not trained on these specific textures, produces satisfactory inpainting. This ability to generalize can be explained by the size of the network, the attention layers, and the training database. The attention layers in the network also helps achieving self-similar solutions.

V.3.3 Quantitative results

We provide an in-depth analysis of the results using metrics that we also recall. It is well known that the notion of ground truth for image inpainting is controversial, but a quantitative evaluation is still needed to objectively compare the results of different methods. We chose to measure so-called *reconstruction* errors, which compute the errors between a ground truth and a prediction. They are far from perfect, but not completely irrelevant in the case of texture inpainting, where the solutions are more constrained than in the general case. For diverse methods that can produce multiple results with different scores, we only report the scores for a single sample. First, let us introduce the metrics we will use.

V.3.3.1 Metrics

PSNR is a measure of the ℓ_2 error between the predicted image and the ground truth. It is convenient to train neural networks on the ℓ_2 reconstruction error, which therefore makes PSNR a suitable metric of evaluation for these approaches. The reconstruction error has is not enough for realism which is why GANs have been a popular alternative for many tasks (Pathak et al. 2016). The optimal is the conditional expectation it is known to be of low quality perceptually (Blau and Michaeli 2018; Saharia et al. 2022a).

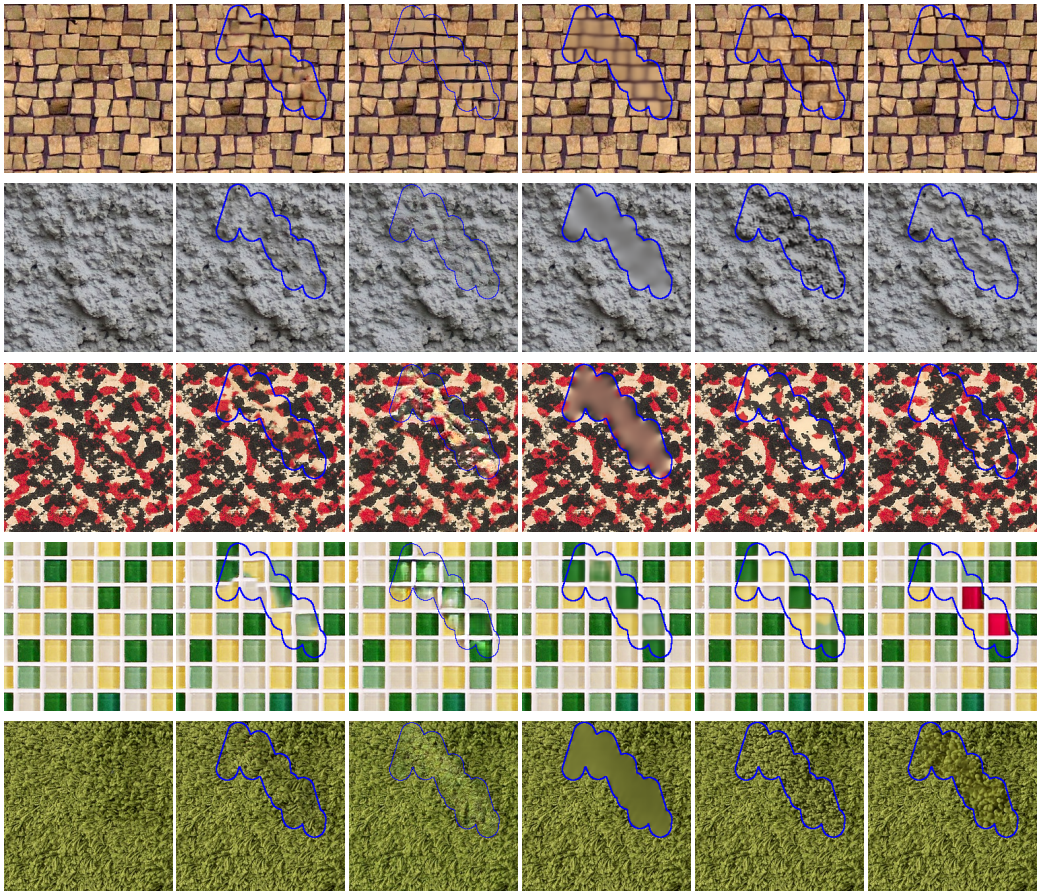


Figure V.5: The results from Regression are very blurry, and those from DeepFill have network artifacts. Result from Patch are satisfactory. Results from RePaint and our method are very good. RePaint results are impressive given that it has not been trained on these textures, nonetheless it may hallucinate (4th row).

SSIM (Wang et al. 2004) is a full reference metric that uses the local means and covariances in small windows of size 8x8. It emphasizes the changes in structure rather than uniform intensity shifts.

LPIPS (Zhang et al. 2018a) This metric is derived from deep neural network trained for other tasks, such as classification, and attempt to better reflect visual visual similarity. It is more sensitive to blurred output, handle small misalignments, and small changes in contrast.

FLIP (Andersson et al. 2020) is a hand-designed perceptual metric that aims to correctly represent the perceived changes when flipping between two images. It was designed especially for comparing image renders. It thus removes the small disalignments and contrast changes. It also includes a per-pixel error map to visualize where the dissimilarity is the highest.

DISTS (Ding et al. 2020) is designed to be similar to SSIM but uses an additional

term to handle the textures. Indeed stochastic textures are assumed to not always match and thus as long as the texture is the same, the samples can be different. There is also a structure evaluation distinct from the texture.

V.3.3.2 Results

Numerically, the results from Table V.1 are simple: the regression approach is the best in PSNR, SSIM, and FLIP. For the two other metrics, the patch method, our diffusion model and RePaint scores are close. The patch metrics are slightly better in this case. DeepFill is the very last in all metrics but LPIPS.

Comparing the visual results from Figure V.5 and the numerical results tells a different story. In fact, the regression approach is very far from the best visual results, it is excessively blurry for stochastic textures. This raises the question of PSNR as a relevant metric for inpainting, especially since there is an important gap between the regression score and the second one. It has recently been suggested that PSNR is only loosely related to the visual quality of the results (Blau and Michaeli 2018; Saharia et al. 2022a) which we confirm it in our experiments. This problem is particularly acute for inverse problems with high degrees of freedom (such as inpainting) and for which the so-called "ground truth" is already doubtful. There is a schism between the real objective of inpainting, producing good looking images, and the metrics we optimize for. It is not surprising that inpainting losses for neural networks often include a secondary objective that balances the reconstruction error and a perceptual loss. Perceptual losses include: GAN objectives (Pathak et al. 2016), or style loss (Gatys et al. 2015).

We hold the same conclusions for SSIM. In our experiments, the inpainting task is not sufficiently constrained, the inpainted structures are too different from the ground truth structures. Then the invariance of SSIM to local intensity shifts is not relevant.

According to the perceptual measure LPIPS (Zhang et al. 2018a), the results are quite different. We see that the patch-based method, our method, and RePaint perform similarly on LPIPS and DISTS, which is confirmed by similar quality visually. Compared to PSNR, LPIPS does not enforce a strict pixel-to-pixel similarity because it compares deep feature maps. LPIPS is also more sensitive to blur than PSNR. Blur is an important visual feature that significanly affects the perception of an image or texture. LPIPS is not without limitations but is currently one of the best options when a ground truth is available and makes sense.

Finally, we wanted to highlight in these experiments that PSNR scores can be deceptive for inpainting metrics. LPIPS better matches the perceptual error but still requires a ground-truth to be computed. Some authors (Saharia et al. 2022b; Suvorov et al. 2022) have adopted an evaluation score based on the Fréchet Inception Distance which measures distance between distributions of features. It does not require ground truths. One last option is to use human evaluators.

Method	PSNR (dB) ↑	SSIM ↑	LPIPS ↓	FLIP ↓	DISTS ↓	#Param.
Patch	28.12	0.902	**0.046**	0.064	**0.026**	-
DeepFill	21.55	0.870	0.105	0.156	0.149	3M
Regression	**30.31**	**0.923**	0.157	**0.045**	0.099	160k
Ours	28.02	0.908	0.057	0.054	0.031	160k
RePaint	26.77	0.899	0.053	0.078	0.031	450M

Table V.1: Reconstruction error using different metrics on our inpainting datasets.

Method	PSNR (dB) ↑	SSIM ↑	LPIPS ↓	FLIP ↓	DISTS ↓
Regression	**30.31**	**0.923**	0.157	**0.045**	0.099
Ours	28.02	0.908	**0.057**	0.054	**0.031**
Ours (averaged)	29.67	0.921	0.141	0.071	0.093

Table V.2: Reconstruction error when using the average of multiple samples for the diffusion approach. Averaging the samples leads to a better PSNR and SSIM but worse LPIPS / FLIP / DISTS.

V.3.3.3 An experiment showing the limitations of the PSNR

In this section, we will show how it is possible to artificially increase the PSNR of our approach. Diffusion models are trained to minimize the reconstruction error but are used in inference to draw samples from the distribution rather than picking the mean of the distribution. To maximize the PSNR, we make the observation that we simply need to generate samples close to the conditional mean. It is possible to approximate this using our diffusion model and many samples[1]:

$$\mathbb{E}[x|y] = \int xp(x|y) \approx \frac{1}{n}\sum_{i=1}^{n} x_i$$

Where the x_i are samples obtained using the conditional reverse process. The diffusion framework is precisely designed to approximate the true conditional distribution.

The numerical results of averaging 100 samples for evaluation are in Table V.2. We see that averaging all these samples allow for a gain of 1.5dB over our sample point estimation. SSIM is also improved. However, the other perceptual metrics are heavily impacted. We recover metrics closer to the Regression approach with a severe degradation in LPIPS, FLIP and DISTS.

Visually Figure V.6, we confirm that averaging the samples from our diffusion model approximate the output of the Regression model which is expected. There is however one difference, is that the regression model cannot give us access to a variance of map of the output while it is possible to exhib it with different samples.

[1] an alternative can also be obtained by using the estimate of x_0 at time $t = T$ *i.e.* $f_\theta(x_T, y, T)$

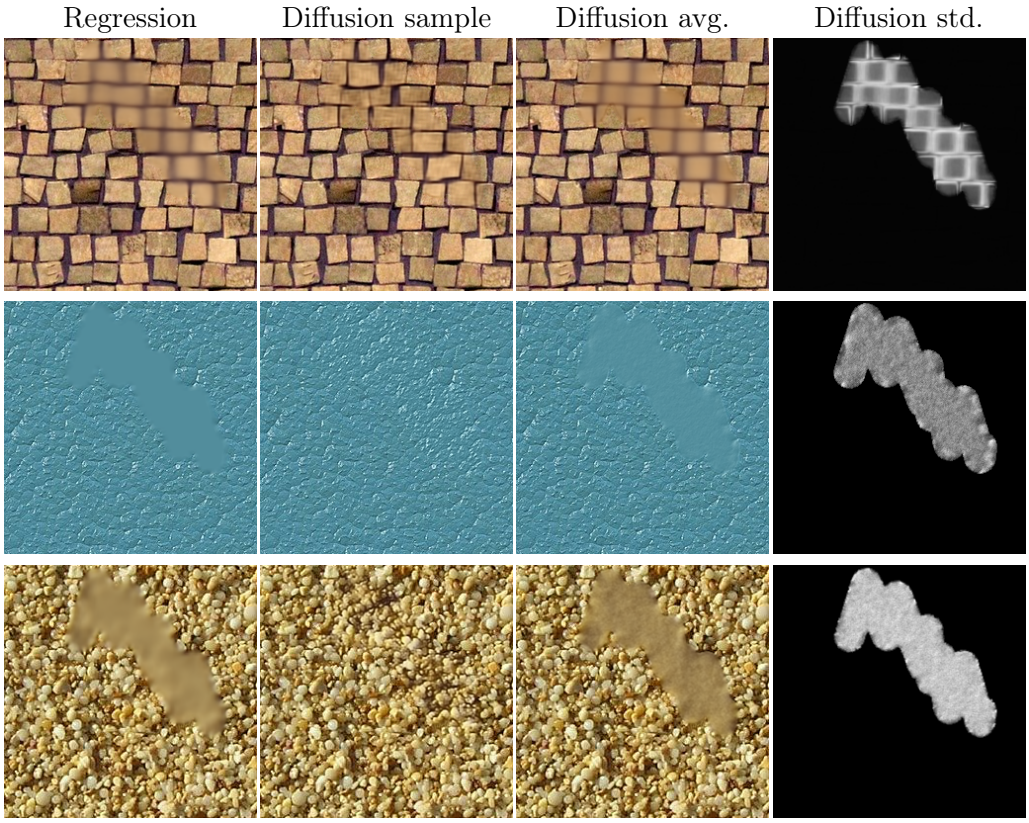


Figure V.6: It is possible to get results close the regression approach by averaging multiple diffusion samples, we also get a variance map. Variance is located along the possible edges (first row) and highlight the stochasticity of the texture.

V.3.4 Frugality

The previous sections show that RePaint (Lugmayr et al. 2022) generates high quality results and is able to generalize to very specific problems like texture inpainting. However, these capabilities have been achieved after a long learning curve and therefore at the cost of significant environmental impacts. We must note, however, that RePaint is better in this respect than Palette (Saharia et al. 2022b). In fact, RePaint is derived from an *existing* unconditional diffusion model, while Palette is specifically trained for inpainting only. This is of course an important consideration given the challenges of climate change. Table V.3 shows the estimated environmental impact for each method, including training and inference time. A major limitation of our approach is that we have to train for each new texture, which is not the case for RePaint or DeepFill. The patch-based approach of Newson et al. (2017) takes slightly longer for inference than ours, but does not require any training, resulting in the most energy-efficient method of all. If we compare the inference cost of RePaint and the total cost of our method, we see that the break-even point is reached when the number of inferences is greater than 2.

	Training		Inference (1 sample)	
	Time (h)	gCO2eq	Time (h)	gCO2eq
Patch	0	0	0.03	0.25
DeepFill	120	1707	0.001	0.01
RePaint	576	9832	0.17	2.4
Ours	0.25	3.5	0.001	0.01

Table V.3: Environmental impact of each method. For neural methods, the training time dominates the total cost, they need to be used multiple times to favorably compete with patch-based approaches.

	Inference time (s)	VRAM (MB)
Patch	18*	23
DeepFill	1	103
RePaint	610	2768
Ours	3	69

Table V.4: Inference time and VRAM requirements show that large models are not really adapted to interactive tools. While engineering efforts can optimize these values, RePaint is too long, and DeepFill is not as good as our method. *: CPU

We estimate the electricity consumption for training and inference of these models using Lacoste et al. (2019). The CO2 equivalent is provided for information purposes, for a French electricity consumption[2].

V.3.5 Interactivity

Outside of the research field, image inpainting is an important tool for artists. In this context, interactivity is an important selling point. Patch-based methods have undergone many optimizations to process images in real time, and be useful in image processing softwares such as Photoshop. In our case, previous methods deep methods like DeepFill (Yu et al. 2018) are a simple forward pass through a moderate-sized neural network. But diffusion models on the other hand, while producing very good and diverse results, have problems with inference speed, and the size of the models often makes them unusable on limited hardware. Our method combines together diffusion and a lightweight model that can be used on consumer hardware and in near real-time. We present a quick comparison of the running time and memory requirements of several inpainting methods for 256x256 images in Table V.4.

One way to add state-of-the-art inpainting algorithms to software products is to keep these applications on a remote server (*Adobe Firefly* 2023) but this requires an internet connection and merely hides the environmental impact of such methods instead of solving it. A second issue is that some of these large diffusion models are trained on copyrighted data. This can be problematic for professional use.

[2] 56.9g CO2 eq/kWh in 2021. Source: Base Empreinte® (ADEME)

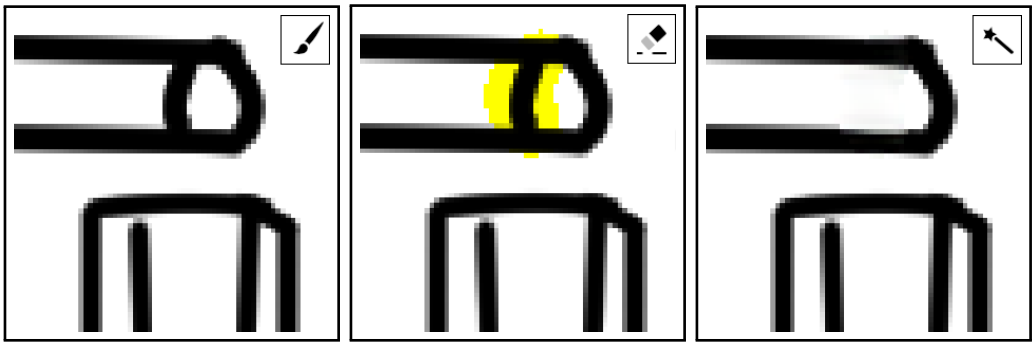


Figure V.7: Proof-of-concept that combines drawing (a), and a magic eraser / inpainting that runs interactively (b-c). Instead of removing all the pixels in a given region, this eraser is actually inpainting the region.

On the other hand, we could imagine a use case, where our method is trained locally on the artist's own data, and used locally. After training, the model can be integrated for interactive use. We show a simple proof of concept in Figure V.7 for line drawing inpainting. In this example, we trained our diffusion model on multiple images of line drawings (Sasaki et al. 2018). In the case where multiple images are available, there is no obvious way to use all the training images for the patch-based method whereas it is easy to do with a learning-based approach.

In our case, the inference time is not that far from truly real time, it could probably be achieved with some optimizations. Some fruitful approaches could be: network pruning and quantization, reducing the number of diffusion steps which is probably more than enough for the bimodal color distributions.

V.4 Limitations

A major limitation of our approach is that it requires a full training for each new texture. This puts our method at a disadvantage against patch-based methods, which are faster than a full training, even though our method is faster for inference. Large neural networks take longer to train and are more computationally expensive, but once trained they can be used in many situations. It is often easier to reduce the size of a powerful network via pruning (Blalock et al. 2020), or distillation (Hinton et al. 2015), than to train a small powerful network (Frankle and Carbin 2023).

The second limitation is that our network cannot synthesize new, unseen content and is thus limited in generalization. In Figure V.8, we show the results of inpainting a face and a natural image. On the natural image, our inpainting of the leopard and the background are satisfactory thanks to self-similarity, but are slightly blurry, this confirms that our method can be applied outside of textures. On the face, without surprise our method fails. RePaint is the best option in both cases, thanks to its large training database.

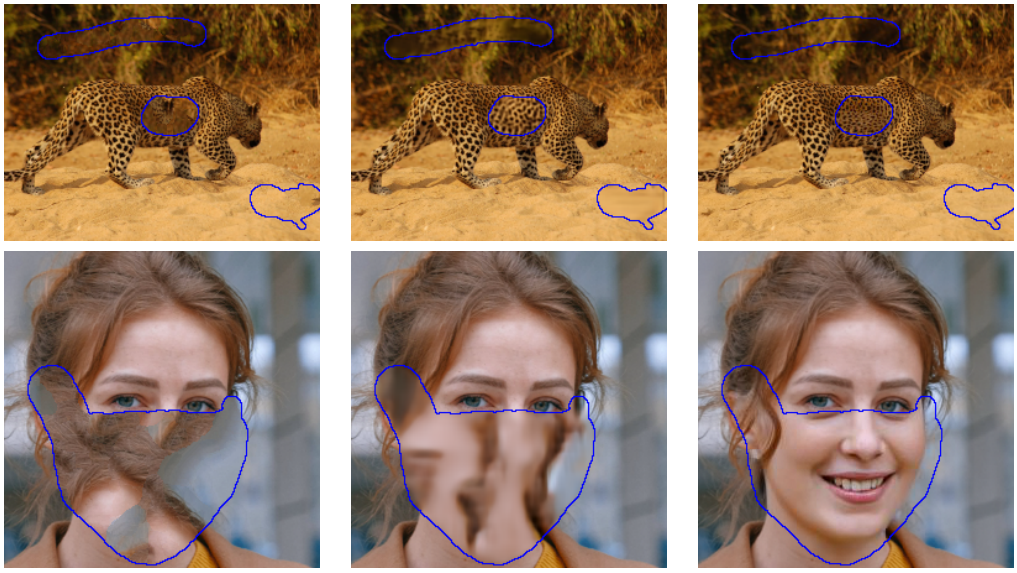


Figure V.8: Results of different methods on non-stationary images. Our method can model a multimodal distribution and be applied to natural images (top). It can't however create new content which is a limitation shared with patch-based methods (bottom).

V.5 Conclusion and perspectives

In this chapter, we have investigated the application of diffusion models to image inpainting using a single image. Diffusion methods have become the state-of-the-art for image generation and for solving many inverse problems. However, the cost of training these models is a major obstacle to their practical use, and raises questions about their durability. We have presented a network based on the principle of diffusion training but retaining only the essentials for its architecture, dividing the number of parameters by more than 1000 and the training time by 500 compared to state-of-the-art methods, without compromising the quality of results in cases of textures. Compared to patch-based approaches, our method is a slower but produce slightly better results.

We have only studied the full training of networks from scratch and not the finetuning on a small dataset, which differs from the current trend of exploiting large foundation models (Hu et al. 2022; Ruiz et al. 2023). Indeed finetuning is probably faster than a full training of a small model. While potentially giving better results, this proposition still requires to use a large network and introduces biases from the external database.

Beyond the extension to videos that we detail in the next chapter (chapter VI), we identify two interesting perspectives.

Architectures for using both patches and neural networks. We have simplified the architecture to use only convolutions. Fully convolutional neural net-

works are easy to train and often perform very well, but newer attention mechanisms could be used. In particular, attention layers have links to patch-based approaches that are well suited to the internal setting. Attention could greatly benefit internal methods to further reduce the learning phase.

Applications to other inverse problems. We have only looked at the inpainting problem which is quite close to generative problems, but other inverse problems could benefit from an internal setting. For example, in medical data, it is common to have limited data but many inverse problems to solve. These algorithms may be of interest in this context.

VI Internal diffusion for video inpainting

In this chapter, we present an extension of chapter V to video inpainting. Video inpainting is a very challenging task due to the high dimensionality of the signal and the temporal consistency required for obtaining convincing results. Despite very impressive results in the case of image inpainting (Saharia et al. 2022b; Yu et al. 2018) extension of deep learning approaches to the case of videos has proved to be difficult. Thus most methods still rely on a combination of deep learning and carefully crafted inference schemes based on optical flow.

Recently, diffusion models have shown impressive results in modeling complex data distributions, including images and videos. They are able to synthesize realistic images with unprecedented quality (Ho et al. 2020). Following their introduction for unconditional data generation of face images and more general natural images, conditional diffusion models dedicated to inverse problems have been proposed (Saharia et al. 2022b). The extensions to videos have been limited due in part to the high computational cost of training such networks, and to GPU memory constraints. Ho et al. (2022c) have presented an extension for the generation of small videos. Others, including MakeAVideo (Singer et al. 2022) and ImagenVideo (Ho et al. 2022b) attempt to circumvent the problem by combining 3D diffusion models with super-resolution and frame interpolation, producing sequences that are nevertheless not completely natural. However, these networks are extremely large, inducing long training and inference times. Several acceleration techniques have been proposed in the framework of Song et al. (2021a), or the network introduced in Rombach et al. (2022), for image diffusion, but they are still not sufficient in the case of videos.

We present the first application of diffusion models to video inpainting. We show that *internal training* is sufficient to train a lightweight network for video inpainting. Internal training means, in this context, that we only use the current information contained in the video to inpaint for the purposes of training. Indeed, this makes a lot of sense for the case of videos. Due to their high level of auto-similarity, the information outside the zone to inpaint is similar to the information inside, and thus represents a good database for inpainting. Furthermore, we present a new method to train diffusion models that is particularly efficient in the case of internal training. In internal training, the trained model is less important than the inpainting result because we expect such a network to be used only once. Therefore we prioritize the result over the model *i.e.* inference over training. Specifically, we

start the inference process and train our network only when needed. This strategy is particularly effective for diffusion models because the inference is made up of a thousand inference steps. Each step is thus solved almost individually instead of training a model to solve them all.

The video results can be found online at https://infusion.telecom-paris.fr/phd.

VI.1 Related work

VI.1.1 Diffusion Models

We complete chapter II on diffusion models, we summarize the main contributions on diffusion models. We especially focus on the application to video generation which is the close to video inpainting, since no other work has been published for video inpainting using diffusion models. Diffusion models (Ho et al. 2020; Sohl-Dickstein et al. 2015) are generative models that have shown great generative capabilities on image datasets. However, a major drawback, is the slow inference and training, which can be accelerated with various techniques (Nichol and Dhariwal 2021; Rombach et al. 2022). A solution to overcome the size of the model is to use a multiscale approach via the Gaussian pyramid (Ho et al. 2022a) or a wavelet decomposition (Guth et al. 2022).

For image inpainting, Lugmayr et al. (2022) use an unconditional diffusion model, with projections and time-travels for better coherence. Improvements to this framework have been proposed by Chung et al. (2023); Kawar et al. (2022b). Saharia et al. (2022b) train a conditional model for diverse image-to-image problems. It has also been proposed to control the inpainting via text (Xie et al. 2023).

However the extension of diffusion models to videos is not straightforward, mainly due to their huge size and training time. Ho et al. (2022c) have extended the diffusion approach to video by using a temporal attention layer for sharing the information along the temporal axis, avoiding costly 3D convolutions. Other approaches have studied the conditional generation of videos via multiple networks for frame interpolation and super-resolution (Ho et al. 2022b; Singer et al. 2022). Harvey et al. (2022) implement a diffusion network for generation based on long temporal attention and frame interpolation. To reduce the computational complexity of handling full 3D videos, Yu et al. (2023) use multiple 2D projections. Mei and Patel (2023) have two models, one for frame content and one for frame motion. Blattmann et al. (2023) finetune an image model for videos by introducing temporal attention layers. Luo et al. (2023) share a basic noise component for all frames representing the structure of the scene and a frame-wise noise for variations in each frame. Another option is to decompose the video into multiple images using neural atlases (Kasten et al. 2021) which can then be handled by image-based diffusion models (Chai et al. 2023). Ceylan et al. (2023) use noise inversion and the attention maps for consistent video editing. Liew et al. (2023) propose text-guided video editing, disentangling content, structure via the depth map, and motion. Very recently, Guo et al. (2024) add a motion module to transfer motions from a reference to a generated image.

Finally, it has recently been shown that diffusion can be applied to small datasets, for example when only a single image is available for training. It is possible to achieve single image generation with only one image (Kulikov et al. 2023; Wang et al. 2022). Similarly, Nikankin et al. (2023) show the application to both image and video, for generation or extrapolation. It is also possible to finetune a general diffusion model for a specific scene (Ruiz et al. 2023).

VI.1.2 Video Inpainting

Classical approaches to video inpainting have long relied on available information already present in the video to achieve inpainting of the occluded region using extensions of image patch-based approaches (Criminisi et al. 2004; Drori et al. 2003). The early works on video inpainting split the problem in two: the easy case of static background occluded by a moving object and the more difficult case of a moving occluded object (Patwardhan et al. 2005). To handle light camera motion, a preprocessing step is then added (Patwardhan et al. 2007).

These patch-based greedy approaches are then replaced by a global optimization problem (Wexler et al. 2007). This optimization is a significant improvement over the greedy approaches as it allows for better global coherency. One issue is the running time of this method. Newson et al. (2014) add texture features, a stabilization step, and adapt the PatchMatch algorithm (Barnes et al. 2009) to greatly improve the computation speed and quality. Patch-based methods can be long to compute but the inpainting can be propagated to new frames by estimating the camera motions (Herling and Broll 2014). Le et al. (2017) adds the optical flow to the patch description and warp patches according to their flow for better reconstructing the different moving objects. Granados et al. (2012) optimize the shift-map, they can handle non-periodic motions but solving the graph-cut problem can be very long.

Another class of approaches is based on *optical flow*. Indeed, in many situations, especially when the occluded region is observed at some point in time in the video, optical flow gives extremely useful information. The first method is by Shiratori et al. (2006), they show that inpainting the optical flow is much easier than inpainting the color values. Matsushita et al. (2006) inpaint the optical flow for full frame video stabilization. Strobel et al. (2014) first inpaint the optical flow which adds constraints in the exemplar-based color inpainting step that follows. Huang et al. (2016) jointly optimize the appearance and the optical flow, making sure that the output is temporally coherent. A joint optimal flow and appearance energy is optimized by Bokov and Vatolin (2018), their method is very fast and handle any camera motion. Xie et al. (2017) apply their method for dynamic texture synthesis in the context of inpainting.

More recently, deep learning-based methods are flourishing because they can finally learn from the large amount of data available. They are however more restricted than traditional methods because they are not memory efficient and HD video inpainting is still a challenge today. Kim et al. (2019) have proposed a direct

inpainting approach using a 3D-2D neural network with an optical flow consistency loss. Wang et al. (2019) perform the inpainting in 2 steps, the first with a 3D convolution network at low resolution and a second one with image inpainting. Gated convolution (Yu et al. 2019) is extended to 3D by Chang et al. (2019) with a 3D patch-GAN.

Murase et al. (2019) perform the optical flow inpainting with a lightweight network and then integrate it. Methods such as (Gao et al. 2020; Xu et al. 2019) first inpaint the optical flow and then propagate pixel values. These methods have been successful in handling non-rigid motions and long range dependencies. Zhang et al. (2022c) proposed to predict the optical flow using more frames, resulting in a more accurate optical flow and better preservation of motions. Li et al. (2022b) integrate the optical flow prediction and propagation inside the training loop optimizing the network parameters for video inpainting rather than optical flow prediction. The first work, to our knowledge, on video inpainting with diffusion is by Gu et al. (2023). They use optical flow to guide the propagation of features in a latent diffusion model.

Attention-based algorithms can be seen as updated versions of patch-based algorithms. They are however severely limited the resolution of the video (recall that attention was already limited by memory constraints). Greedy onion peeling gets revisited by Oh et al. (2019). Affine transforms can align frames and copy-paste algorithms get improved with deep learning (Lee et al. 2019). Sometimes it is necessary to use both the near frames and distant frames (Li et al. 2020; Woo et al. 2020). Zeng et al. (2020) align the frames and use a Transformer to fill multiple regions at the same time using self-attention. After initialization of the inpainted region, Hu et al. (2020) combine multiple proposals together which are larger than just pixels from Transformers. Liu et al. (2021b) adapt overlapping patches which is more expensive but allows for more flexibility and better reconstruction quality. Zhang et al. (2022b) combine a Transformer with optical flow. To avoid the problem of rigid patches, Cai et al. (2022) deform the patches used in a Transformer with homographies. Lastly, Propainter (Zhou et al. 2023) improves the propagation step of optical flow methods and avoid useless computations in the attention modules using only the masked queries.

Related to internal learning, Zhang et al. (2019b) learn a model on a single video similarly to Deep Image Prior Ulyanov et al. 2018. They also model the optical flow for better temporal coherency. Ouyang et al. (2021) tackle the same problem but propose a solution to avoid explicit modeling of the optical flow. Finally, Ren et al. (2022) use VAE with a discrete codebook finetuned on the target video, similar to a masked quantized autoencoder.

VI.1.3 Dynamic Texture Synthesis

Early work in dynamic texture synthesis modeled the problem with a linear dynamical system (Doretto et al. 2003). This was later refined into a multi-scale and autoregressive model by Doretto et al. (2004). Costantini et al. (2008) factorize the video with a higher-order Singular Value Decomposition to avoid the simplistic

temporal decomposition. Zhou et al. (2009) use a non-linear dynamic system and a different sampling method. You et al. (2016) find that kernels allow non-linear modeling and improve the results without stability problems.

The greedy exemplar-based method from Wei and Levoy (2000) and its energy-based alternative (Kwatra et al. 2003) also work for dynamic texture synthesis. Spatiotemporal patches are rearranged to generate new texture samples.

Following the seminal work of Gatys et al. (2015), neural networks have been used to describe dynamic textures. Synthesis achieved by optimizing the Gram matrices (Funke et al. 2017; Yang et al. 2016). Tesfaldet et al. (2018) optimize the pixels of a video to have matching statistics of appearance and optical flow.

Xie et al. (2017) use an energy-based framework whose parameters are described by a neural network. After the analysis, the synthesis is done by Langevin dynamics. Dorkenwald et al. (2021) learn a latent representation of a video and combine it with scene dynamics through a a decoder to generate new stochastic videos. Finally, DyNCA (Pajouheshgar et al. 2023) extends the work on cellular automata (Mordvintsev et al. 2020) to dynamic textures. A tiny neural network learn the update rules of the automata.

VI.2 Method

Our method builds on the Denoising Diffusion Probabilistic Models (DDPM) (Ho et al. 2020) and chapter V, based on discrete forward and backward processes. The formulation is very similar to the previous chapter: the framework is adapted for inpainting by modeling the conditional data distribution. Our observations y are an incomplete video this time. Our vectors are from $\mathbb{R}^{L\times H\times W\times 3}$ representing RGB videos of L frames of size $H \times W$. Despite the presence of a temporal component in videos, the t variable is only used for the diffusion timesteps.

The unknown distribution of natural videos $q(x_0|y)$ conditioned on y is degraded into pure noise $q(x_T|y)$ through a series of 1000 steps. We keep a variance schedule linearly increasing from $\beta_1 = 0.0001$ to $\beta_T = 0.02$.

The reverse process is learned by a neural network predicting the means of the Gaussian variables. We keep a fixed variance schedule instead of learning the variances: $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$.

In the case of inpainting, as mentioned above, we are interested in the conditional data distribution $p(x|y)$ where y are our observations *i.e.* the known region of the image. The observations are introduced in the network as incomplete images via concatenation in the input layer (similarly to Saharia et al. (2022b)). We also add the mask information as input. We train our network by minimizing the reweighted ℓ_2 loss of the x_0-parametrization (Ho et al. 2020):

$$\mathbb{E}_{x_0,\varepsilon,t,M}\|M \odot (x_0 - f_\theta(x_t, y, t, M))\|_2^2 \tag{VI.1}$$

Where $M \sim \mathcal{P}_{\text{mask}}$ is a binary mask with 0 for the known region and 1 elsewhere, y is the masked video *i.e.* $y = x_0 \odot (1-M)$, and x_t is obtained by the forward diffusion

of x_0 by the following:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\varepsilon$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \Pi_{i=0}^{t} \alpha_t$, $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$.

This objective function derived from the Evidence Lower BOund can be expressed as a sum of loss functions (one for each timestep):

$$\mathcal{L}_t(x_0) = w(t)\|x_0 - \hat{x}_0(x_t, t)\|^2$$

with w a weighting function depending on the timestep. The choice of this weighting function is crucial for convergence and quality but is difficult to set. The original authors (Ho et al. 2020) suggest using a simple uniform weighting for easier convergence. Nichol and Dhariwal (2021) uses importance sampling to compute the weighting term to stabilize the convergence. Hang et al. (2023) propose a truncated version to avoid extreme values of this weighting term. In our case, we use a uniform weight of 1 but we use a specific training schedule (detailed in subsection VI.2.3). We get better results this way than with other tested strategies.

Unless specified we use the default parameters of DDPM (Ho et al. 2020): variance schedule, sampling, time schedule. Therefore, our method could benefit from many of the recent improvements for each of these items. We also operate in the pixel space, and not a latent space as proposed by Rombach et al. (2022).

VI.2.1 Architecture

We extend the traditional UNet architecture to video by using 3D convolution. Instead of focusing on very long-range dependencies, we focus more on local temporal features and thus use a 3D convolution approach within a UNet. Local convolutions are more important for modeling dynamic textures, which is our goal, than long-range dependencies through attention layers, or transformers.

By training on a single video, we avoid the need for very long-range dependencies. In fact these dependencies are learned during training. The spatial relationship between elements is not inferred at test time but embedded in the weights of the network. A disadvantage is that it does not enforce coherent completion for the generated content. A similar context may give a similar inpainting content but no strict rule enforces the same appearance for texture samples. This problem will be illustrated later subsection VI.4.1.

Since we are training on a single video, we can limit the number of parameters in our network. With 16 convolutional layers, 32 channels, our network has 500k parameters which are to be compared to the 16.5B parameters of ImagenVideo (Ho et al. 2022b) for text-to-video generation. Even image models are really big with 450M parameters (Lugmayr et al. 2022).

VI.2.2 Single Video (Internal) Training

Our neural network is trained to inpaint a single video, using the video itself. This type of approach is referred to as *internal learning*. Fundamentally, it is based

on the *auto-similarity* hypothesis, which has been successfully used in the case of patch-based methods (Newson et al. 2014) and neural networks via internal learning (Alkobi et al. 2023; Ouyang et al. 2021; Zhang et al. 2019b). It is quite logical that such a hypothesis is successful for video inpainting; videos contain roughly the same content in every frame, with some movement, with some or all of the occluded content being revealed at some point in time. Thus, what might be considered as overfitting in a normal machine learning situation is in fact desirable here: we want to use the content of the video for the inpainting task. This kind of situation also arises in the problem of Single-Image Generation (Shaham et al. 2019).

Thus, it is possible to efficiently inpaint a video without requiring a large external dataset. We assume that a binary test mask M_{test} is provided. It is equal to 1 in the region to be inpainted, 0 elsewhere. Training is done by sampling consecutive frames from the video, and generating training masks which are to be inpainted / denoised by our neural network. We prevent data leakage from the test region by additional masking operations for the inputs:

$$\begin{aligned} x_t &= \sqrt{\bar{\alpha}_t}(1 - M_{\text{test}}) \odot x_0 + \sqrt{1 - \bar{\alpha}_t}\varepsilon \\ y &= (1 - M_{\text{test}}) \odot (1 - M_{\text{train}}) \odot x_0 \\ M &= 1 - (1 - M_{\text{test}}) \odot (1 - M_{\text{train}}) \end{aligned}$$

M is thus the logical OR combination of M_{test} and M_{train}. Similarly, we ignore some parts of the objective function:

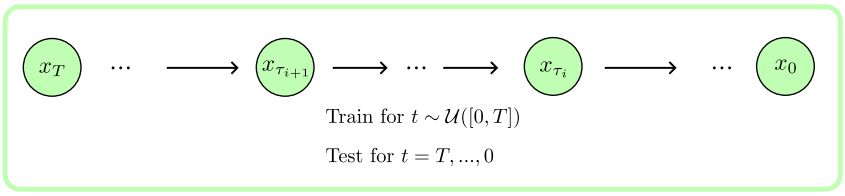
$$\mathcal{L}_\theta(x_0) = \|M_{\text{train}} \odot (1 - M_{\text{test}}) \odot (x_0 - f_\theta(x_t, y, M, t))\|^2$$

VI.2.3 Interval training

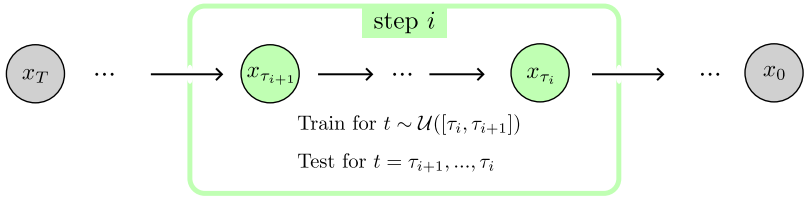
Training a diffusion network obviously involves training it for all timesteps t, from 1 to T, since all these steps are needed during inference. However, depending on the timestep, the goal of the network is quite different. If t is large (pure noise), then the network should behave more like a generative model because the input is mainly noise. If t is small, then the network should behave more like a denoiser, since the original video is only slightly noisy. This observation is crucial: indeed, in a standard diffusion model, a single network is supposed to achieve these very different goals. This is why diffusion networks tend to be extremely large.

To circumvent this problem, we propose to use one lightweight network trained on only a subset of timesteps at a time. Training starts from the end of the diffusion process, dividing the 1000 timesteps into a number of non-overlapping intervals. We train our model on a given interval. Once training on that interval is complete, we use the model to infer the start of the next time step. At this point, the model is trained only on the next interval. This is carried out until we have reached time step $t = 0$. We can, of course, save the intermediate models if we wish, but this is not really necessary for video inpainting, since the model can only be used for the current video.

In practice, we define a time schedule of N steps such that $\tau_1 = 0 < \tau_1 < ... < \tau_N = T$. Our network learns successively on each interval $[\tau_i, \tau_{i+1}]$, starting with



(a) Baseline



(b) Interval training

Figure VI.1: Baseline vs Interval training. In interval training, we train our network on a subset of timesteps of the diffusion process.

the noisiest interval. After training on one interval, the inference is done for the same interval *i.e.* for all timesteps from τ_{i+1} to τ_i (details in Alg. 5).

This strategy has several advantages. First, we can use a lighter network as it specializes only on a small subset of timesteps. Second, we can avoid complicated weighting scheme for the loss function: in one interval all timesteps have roughly the same theoretical weight and setting a uniform weight 1 is not a bad approximation. This is not the case when one network is used and must perform well on all T timesteps (Hang et al. 2023).

We note that as we were developping our interval training approach, a concurrent method by Balaji et al. (2023) was developed for image generation which shares a similar training methodology. In their work, they create an ensemble of networks specialised in each timestep. However, we note a fundamental difference in our approach: once the model has been trained for a given interval, we can let it "forget" that interval since it is no longer required for further inpainting. In practice, we simply finetune it on the next interval. In this way, the relevant weights are retained and fine-tuned while the useless bits are replaced by more approriate ones. In the approach of Balaji *et al.*, the whole ensemble is required for inference. Thus, our approach is considerably more frugal, and thus remains lightweight as we said before.

VI.3 Video inpainting experiments

VI.3.1 Evaluation

For evaluation, we use Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al. 2018a) which are full-reference quality evaluations. However these metrics are

Algorithm 5 Training / inference algorithm for interval training

Require: model f_θ, intervals $\tau_{i=1..n}$, test data y_{test}
  $x_{\text{test}} \leftarrow \mathcal{N}(0, \mathbf{I})$
  for $i = N-1, ..., 1$ **do**
    for $j = 1, .., K$ **do** ▷ Training for K iterations
      $x_j, y_j, \varepsilon_j, M_j \leftarrow \text{NextBatch}(j)$
      $t \leftarrow \mathcal{U}(\tau_i, \tau_{i+1})$
      Take gradient step with gradient:
        $\nabla_\theta \|x_j - f_\theta(\sqrt{\bar{\alpha}_t} x_j + \sqrt{1-\bar{\alpha}_t}\varepsilon_j, y_j, t, M_j)\|^2$
    end for
    for $t = \tau_{i+1}, ..., \tau_i$ **do** ▷ Inference
      $\mu_\theta \leftarrow f_\theta(x_{\text{test}}, y_{\text{test}}, t)$
      $x_{\text{test}} \sim \mathcal{N}(\mu_\theta, \sigma_t^2 \mathbf{I})$
    end for
  end for
  return x_{test}

designed for images and do not take into account the temporal component. To complete these metrics, an extension of the Fréchet Inception Distance has been proposed: Vidéo Fréchet Inception Distance (VFID) by Unterthiner et al. (2019); Wang et al. (2018a). The image classification network is replaced by a network for video classification (Carreira and Zisserman 2017). The classification network is used to extract deep spatiotemporal features and then modeled by a multivariate Gaussian is fitted. In video inpainting, the community (Li et al. 2022b; Liu et al. 2021b; Zeng et al. 2020) has adapted the following evaluation procedure: for each video, extract the deep features, averaging spatially and temporally if needed to have a single 1024-dimensional vector per video. These features are collected for the inpainted video on one side, and for the original video on the other side. The Gaussian parameters are estimated and the distance is computed. This procedure has a strong limitation however, contrary to image generation, the number of samples with respect to the dimension is very low which makes the estimation of the high-dimensional Gaussians hazardous. We keep it that way for comparison with the other methods, under the name VFID, but propose another solution.

To evaluate the quality of single video generation, Gur et al. (2020) proposed to extend the Single Image Fréchet Inception Distance (SIFID) from Shaham et al. (2019) to videos (SVFID). The idea is to extract low-level features that are descriptive of the original video appearance, rather than high-level features which are more semantic and are often used for FID or VFID. By using low level features, we get more data to estimate the Gaussian parameters. In the case of generation, the sets of generated data and reference data are disjoint, but not for inpainting.

For inpainting, we propose this new way of computing the metric because the inpainted video and the original video differ only in the occlusion, so we are comparing the features of the inpainted region with the features of the removed object

which has little sense. What we propose is to compare the features from the completed region and the features from the rest of the video. This biases the result into promoting inpainted content that is similar to its surroundings, which limits the generative aspect of inpainting but is a lesser evil compared to the alternative and is approriate for video inpainting for which very good guesses are possible compared to image inpainting. We note it SVFID°.

In practice, we use the 480-dimensional features from the layer "Mixed_3c". The reduction is still acceptable at this layer to provide enough data points for a correct estimation of the Gaussian parameters. For SVFID°, we resize the inpainting mask to the size of the features. The interior features are not that easily separated from the exterior but this approximation is common in inpainting (Yu et al. 2018).

VI.3.2 Training details

For training, we specifically use a single video and avoid introducing external data into our training set. We resize the training data and masks to 432x240. While our architecture is lightweight and can be used with many frames on a reasonable GPU, we only use 24 frames at a time for training. Unlike other video generation approaches or some video inpainting algorithms, we use consecutive frames without dropping any of them.

We generate training masks similar to the evaluation masks from Liu et al. (2021b). Their algorithm generates random shapes using Bezier curves. The shape is the same throughout the video but can move. Specifically, a mask can be fixed, relatively to the image borders, or move through a set of random points with a random acceleration component.

For better matching between the training and test sets, we generate masks of similar size and motion of the test mask. We use the Adam optimizer with an initial learning rate of 1e-4, and train for 200k steps. For interval training, the 1000 timesteps are divided into 20 intervals of length 50. Each interval has the same training budget.

VI.3.3 Video reconstruction

We present here our results on video *reconstruction i.e.* video inpainting with a ground truth. A common currently-used method of evaluating video inpainting methods is to take a video database (often the "DAVIS" database Perazzi et al. 2016), and add an artificial occlusion mask to the videos. This is done so that a ground-truth is available for evaluation, but results in somehow artificial masking conditions. Moreover, it is well-known that in the case of inpainting, the notion of ground-truth is debatable, since many different inpainting solutions can look realistic. This task and its evaluation procedure has become the new standard for comparing and benchmarking video inpainting algorithms (Li et al. 2022b; Liu et al. 2021b; Zeng et al. 2020).

We thus adopt the standard evaluation method in video inpainting (Liu et al.

2021b). We evaluate our method using 50 sequences from the DAVIS dataset (Perazzi et al. 2016; Pont-Tuset et al. 2017). We use the synthetic masks from Liu et al. (2021b). These results are interesting for evaluation but not as much as qualitative evaluation. An example is found in Figure VI.2. In contrast to the masks found in the object removal task, these masks are generated independently of the video content, they are completely artificial and have no semantic meaning. As a result, they often cover simple cases of video inpainting: static backgrounds and small partial occlusions. We argue that the real challenge lies in complex motion and dynamic backgrounds, which is not reflected in these evaluations. We stabilize, when possible, the videos which is an important pre-processing step for our method (see subsection VI.3.7) and train our network for 200k iterations.

We report the results using a single sample and do not use the fact that our method can generate multiple solutions. We measure the frame-wise PSNR and SSIM, as well as Video Fréchet Inception Distance (VFID). The first two metrics rely on a ground-truth, which, as explained before, prevent their use in the case of real object removal applications. These metrics are moreover computed frame-by-frame and therefore invariant to motion discontinuities. We also report the VFID (Wang et al. 2018a), which uses spatio-temporal features and does not require a pixelwise ground truth: the statistics of all the inpainted videos are compared to the statistics of all the original videos. Quantitative results can be found in Table VI.1, the results for other methods are taken from Zhou et al. (2023). Our method performs worse than several other methods. ProPainter gets the best results overall, in PSNR, SSIM, and VFID. It turns out that video reconstruction is a restoration problem for which solutions very close to the ground truth can be achieved, thanks to camera motions and moving occlusions, and thus very high PSNR values can be reached. In these cases, optical flow-based approaches can provide a very accurate completion, while any invention, no matter how good, is penalized. Our method is designed to hallucinate dynamic and stochastic completions, which are still a challenge in video inpainting today. These cases are of secondary importance in the DAVIS benchmark, which focuses on the reconstruction of static backgrounds. These simpler cases are very well handled by optical flow or affine motion estimation (Bokov and Vatolin 2018; Herling and Broll 2014), and have been further refined using deep neural networks (Zhou et al. 2023).

A major limitation of our approach becomes appearant: after training on a synthetic mask, our model cannot be used for object removal, which is the interesting part of video inpainting. In fact, our model is specific to a video/mask pair, and reusing the same model for a different mask is equivalent to introducing test data into the training set. This is a methodological problem, but it can still be done. There is also a risk that a model will overfit to a video and not be able to remove an object whose appearance and position are hardwired into the weights. Ideally, a new round of training and inference round must be performed. Our method is not suitable for large-scale experiments, as it requires training a new network for each video. We refrain from evaluating our method on 508 videos and synthetic masks from the Youtube-VOS test set (Xu et al. 2018a).

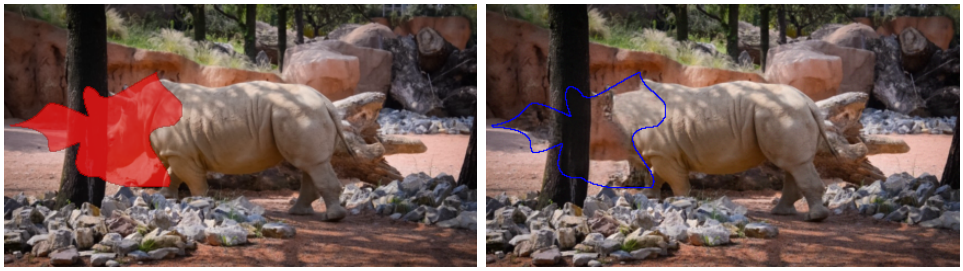


Figure VI.2: Measuring the reconstruction error on the DAVIS dataset using synthetic masks. Right: result using Infusion. Link to videos

VI.3.4 Object removal

Object removal is probably the most exciting application of video inpainting, but it has the major drawback of being difficult to evaluate quantitatively. Therefore, in this section, we propose to *look* mainly at the results of video inpainting results and not to judge the results by numbers from a table. Furthermore, we are interested in complex motion and dynamic backgrounds, which is the real challenge of video inpainting today. The easy cases (static backgrounds) are now really well handled now using optical flow, affine motion estimation, or deep networks (Bokov and Vatolin 2018; Gao et al. 2020; Herling and Broll 2014). We use the 4 videos from Granados et al. (2012), 3 videos from Newson et al. (2014) and 1 video from Le et al. (2017). All of them provide accurate segmentation masks and present challenging situations for video inpainting.

We compare ourselves to a classical patch-based approach (Newson et al. 2014), and more recent deep learning approaches based on optical flow: E2FGVI (Li et al. 2022b) and ISVI (Zhang et al. 2022c).

For object removal, we obtain very good results on different videos using the provided test masks, comparable to the state-of-the-art flow-based approaches for static background and significantly better for dynamic backgrounds such as dynamic textures (Figure VI.3) or occluded moving objects (Figure VI.4). We wan to emphasize that static background inpainting is not really our focus as it is already solved in many situations. Our method shines when generation is needed because the masked region is never seen, or the optical flow is of little help. Our framework is well suited for dynamic texture inpainting which requires stochasticity both temporally and spatially. No other deep learning method can currently handle that. Examples of dynamic textures with comparisons with other methods can be found in the videos *fontaine-chatelet* and *jumping-girl-fire* (https://infusion.telecom-paris.fr/phd).

For complex motion, optical flow is difficult to inpaint because it requires a multi-frame understanding of the motion. Long temporal occlusions such as those found in *loulous* (Figure VI.3, bottom) or *museum* (Figure VI.4) are challenging examples. We see that in the museum sequence, our method better preserves the appearance of people crossing the occlusion while the other methods show people appearing and disappearing. In the *loulous* sequence, our method is able to reconstruct the

Model	PSNR (dB) ↑	SSIM ↑	VFID ↓
VINET (2019)	28.96	0.941	0.199
DFVI (2019)	28.81	0.940	0.187
CAP (2019)	30.28	0.952	0.182
FGVC (2020)	30.80	0.950	0.165
STTN (2020)	30.67	0.956	0.149
TSAM (2021)	30.67	0.955	0.146
FuseFormer (2021b)	32.54	0.970	0.138
ISVI (2022c)	32.54	0.970	0.138
FGT (2022b)	32.86	0.965	0.129
E2FGVI (2022b)	33.01	0.972	0.116
ProPainter (2023)	**34.47**	**0.978**	**0.098**
Infusion (ours)	31.55	0.970	0.183

Table VI.1: Inpainting metrics on the DAVIS dataset using the evaluation masks of Liu et al. (2021b)

Method	SVFID° ↓
ISVI Zhang et al. 2022c	29.81
E2FGVI Li et al. 2022b	29.88
Patch Newson et al. 2014	25.59
Infusion	31.29

Table VI.2: Inpainting metrics for object removal. Our method is the worse under this metric.

occluded person without distortion and also generates a convincing moving body.

We also try to evaluate our method using the SVFID°, which is not ideal, but a less bad solution in these cases. In fact, SVFID° measures the distance between the feature distributions inside the occlusion and outside the occlusion, so it does not directly measure the quality of the inpainting, but rather the self-similarity of the inpainted video. The results can be found in Table VI.2 and place our method in the last position among the ones that we benchmarked.

VI.3.5 Dynamic texture inpainting

We also evaluate our method on dynamic textures. The dynamic textures are also mostly absent from the usual training set for video inpainting. It is therefore expected that methods trained on such datasets will underperform. Indeed, dynamic textures have been a blind spot of recent methods, especially deep learning methods, in spite of the great importance and presence of this components of videos. In fact, they are key for obtaining realistic and vivid results.

We use the texture dataset from Tesfaldet et al. (2018), which is a collection of 59 textures. Since the dataset is designed for texture synthesis, no mask is provided,

Figure VI.3: Results on object removal in the context of dynamic textures. Link to videos

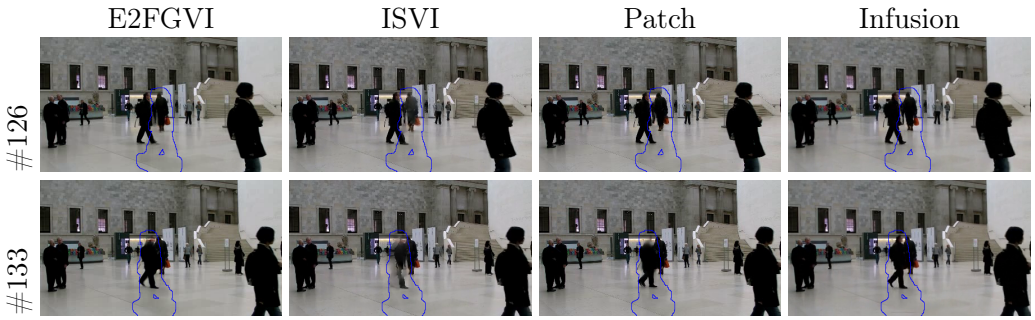


Figure VI.4: On the *museum* sequence, our method can faithfully reconstruct a moving object hidden behind the occlusion. The other methods struggle with this complex occlusion which lasts for several frames. Link to videos

Method	PSNR (dB) ↑	SSIM ↑	LPIPS ↓	SVFID° ↓
ISVI Zhang et al. 2022c	27.41	0.945	0.062	1.359
E2FGVI Li et al. 2022b	**31.58**	**0.961**	0.046	1.562
Patch Newson et al. 2014	29.47	0.954	0.051	1.035
Infusion	30.14	0.958	**0.039**	**0.517**

Table VI.3: Inpainting metrics for dynamic textures. E2FGVI has better PSNR and SSIM, but our method performs better on the perceptual metrics.

and we define synthetic masks using the algorithm from Kim et al. (2019). In the case of the DAVIS dataset, the ground truth is not unique and many solutions are possible with very different appearances. For texture inpainting, there is no unique solution either, but the completion should look like the rest of the texture due to stationarity. Thus, the pixelwise error metric (PSNR) is not completely accurate, but a perceputal metric such as LPIPS is reasonable for evaluation. For our method, we train for fewer iterations than before due to the small size of the texture samples (12 frames of size 256x256).

We report the quantitative results in Table VI.3, which shows that the evaluated deep learning approaches perform poorly on these textures. E2FGVI still reports high PSNR, but we have seen that this does not alway correlate with better visual results Saharia et al. (2022a). In fact, for textures, the PSNR can be misleading due to the stochasticity in both space and time. Therefore, it is possible to achieve high PSNR by ignoring these variations. However, this leads to very unrealistic results such as a static and blurred inpainting. Infusion has better perceptual scores, followed by the patch-based method of Newson et al. (2014). We let the reader judge by looking at Figure VI.5. We can see that the results of the competing methods are either blurry or still while the patch-based method performs well on these examples. Our method is able to produce complex and visually satisfying dynamic textures. As before, these results are better appreciated when played as videos. It turns out that our method is perfectly adapted to this setting where the optical flow is unusable and the content must be hallucinated. Unlike other deep methods which are deterministic, our method is inherently diverse and better reflects the stochastic nature of dynamic textures.

VI.3.6 Ablation study

In this section, we show the contribution of interval training through several experiments. We train the same neural network for the same total number of training steps for both the baseline *i.e.* training on 1000 timesteps, and the interval training *i.e.* training sequentially for small interval of timesteps.

We first evaluate this in image texture inpainting, using the same dataset and experimental setting from chapter V). The quantitative results are reported in Table VI.4. We see that training with interval training improves the PSNR and lowers the LPIPS, the SSIM values are very similar. We also show visual results in Figure VI.6, we observe that the baseline inpainting do not correctly match the color of the surrounding area, this is not the case for interval training. More details are inpainted because the loss is no longer dominated by the most noisy steps.

We then confirm these results on the dynamic texture inpainting task. Using the dataset from Tesfaldet et al. (2018), we measure the inpainting error in frame-wise PSNR, SSIM, LPIPS, and SFVID introduced earlier. The quantitative results can be found in Table VI.5 and the visual results can be found in Figure VI.7. Once again, we see an undeniable improvement over the baseline.

Finally, we note that the weighting scheme of the diffusion loss is important

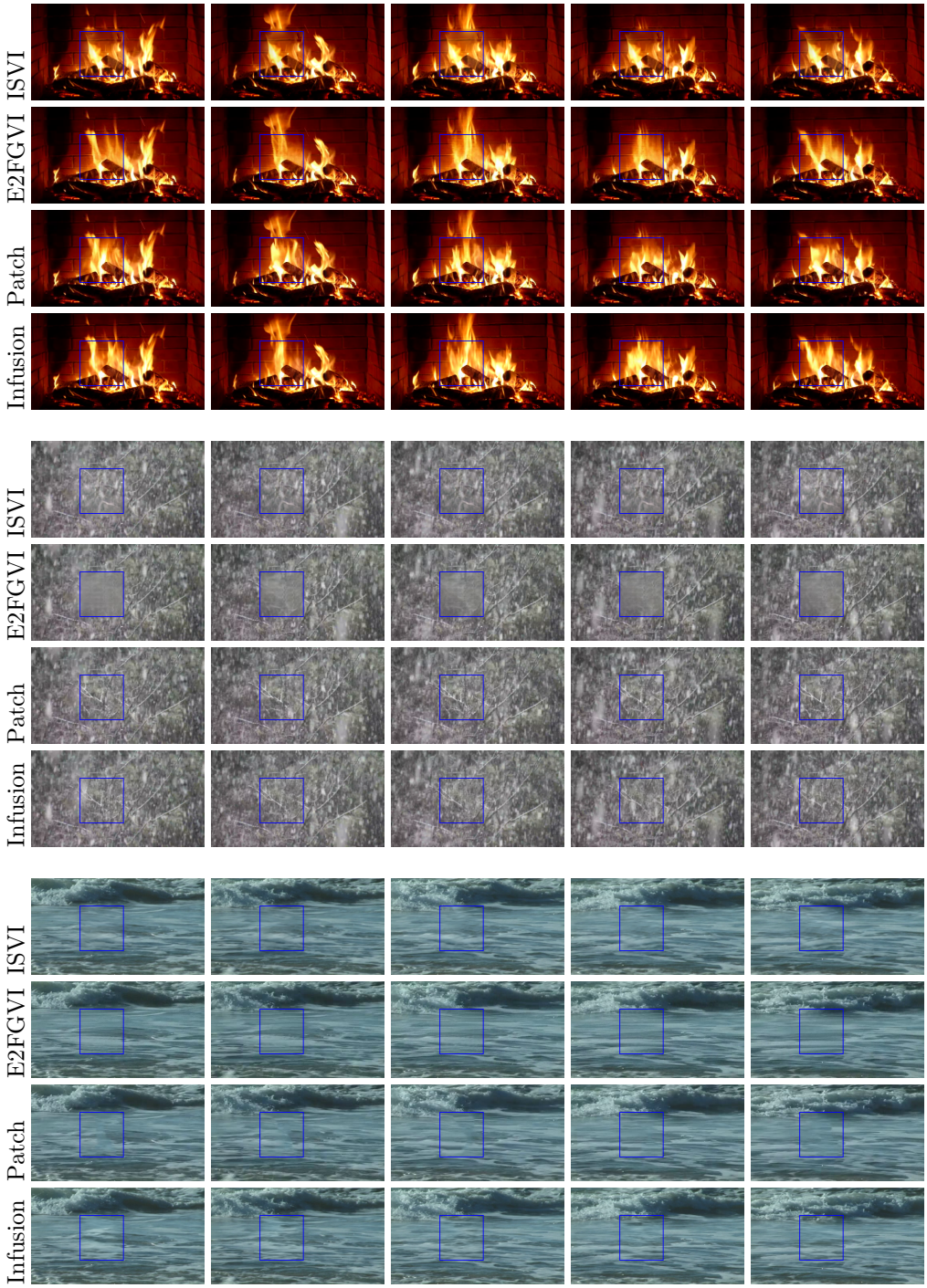


Figure VI.5: On these challenging textures, Infusion is able to inpaint all of theme satisfactorily. Link to videos

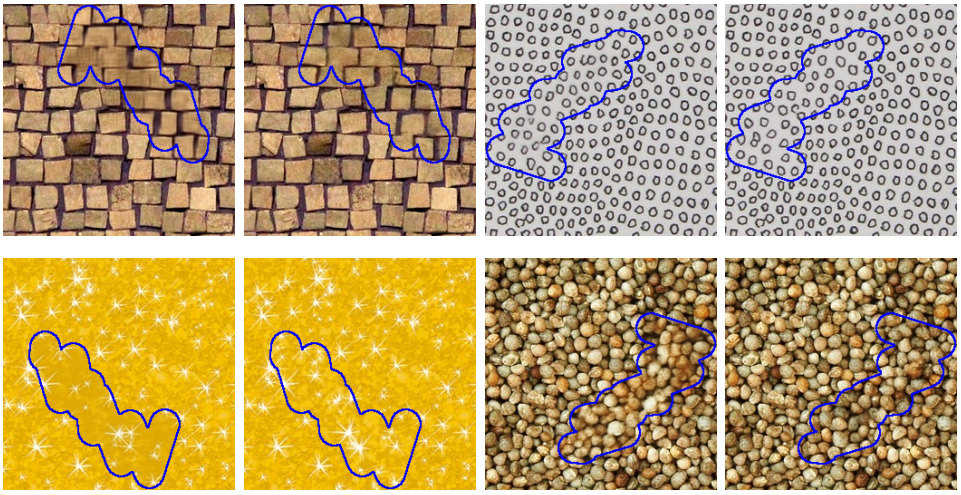


Figure VI.6: Texture inpainting, with and without interval training. Left is the baseline. Right is with interval training. The results using interval training are better visually. Link to videos

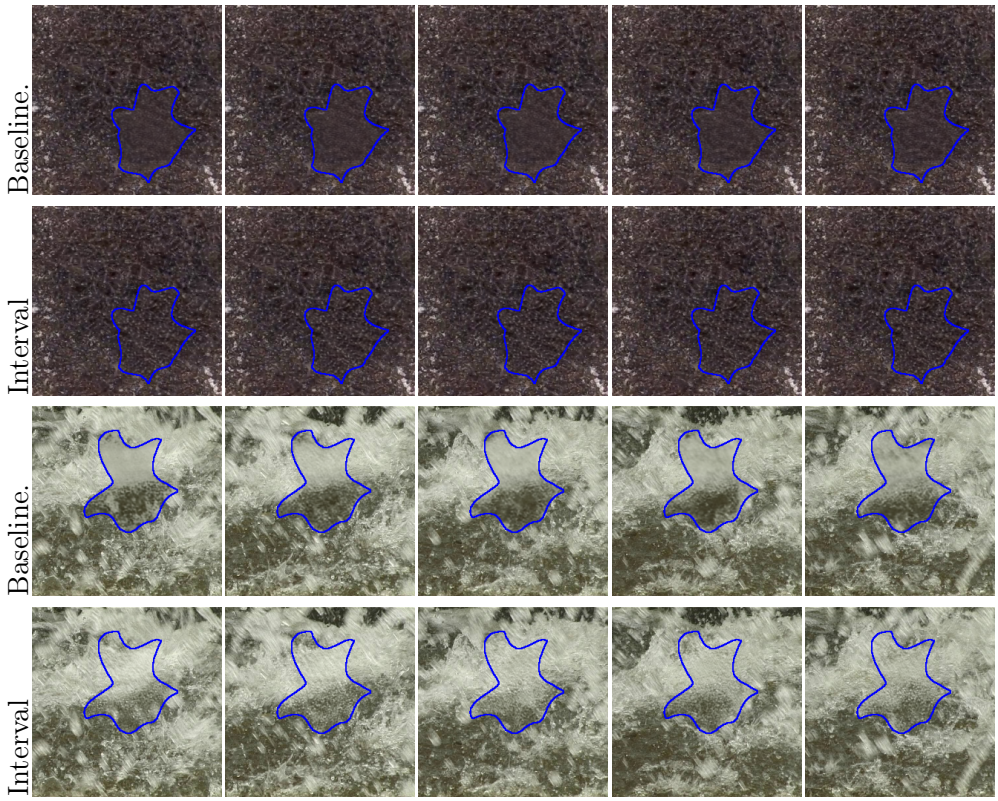


Figure VI.7: Baseline vs interval training on dynamic textures inpainting. Interval training improves the visual results. Best seen with zoom-in. Link to videos.

Method	PSNR (dB) ↑	SSIM ↑	LPIPS ↓
Baseline	28.02	**0.908**	0.057
Interval	**28.33**	0.907	**0.042**

Table VI.4: Inpainting metrics on image textures when using the interval approach vs baseline. Interval training improves the performances of the diffusion model at no cost (same training time).

Method	PSNR (dB) ↑	SSIM ↑	LPIPS ↓	SVFID ↓
Baseline	29.54	0.960	0.042	0.720
Interval	**30.27**	**0.964**	**0.035**	**0.517**

Table VI.5: Inpainting metrics on dynamic textures when using interval training vs baseline. All metrics are improved with interval training.

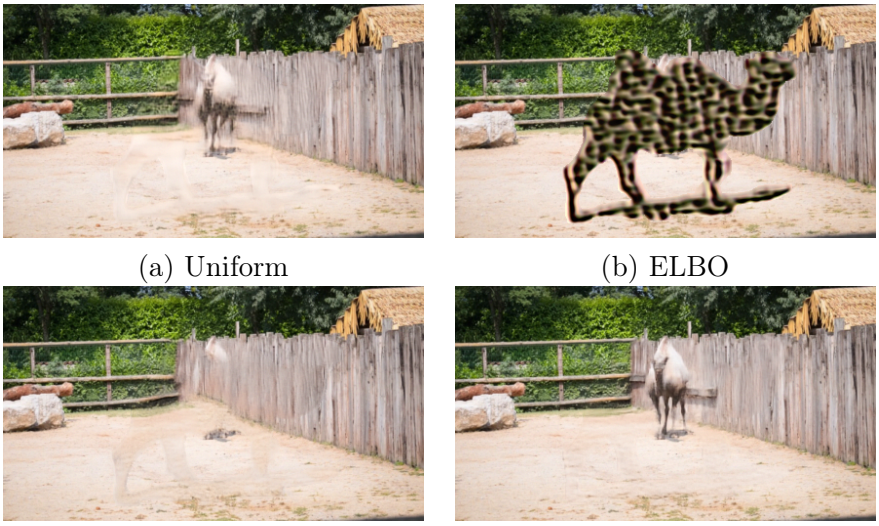
for baseline training. Indeed it is somewhat expected that when training on all timesteps, the ℓ_2 error, and thus the gradients, will be larger for noisy inputs than for less noisy ones. Thus, uniform weighting in the x_0-parametrization implicitly focuses on large timesteps, which explains the lack of detail. Our method simplifies the training of diffusion models for which the weighting term for the different timesteps can be difficult to set in order to achieve both a low Fréchet Inception Distance and a good likelihood (Ho et al. 2020; Nichol and Dhariwal 2021). Specifically, we compare the results when training with a uniform weight for all timesteps (baseline), the theoretical weight derived from the Evidence Lower Bound (ELBO) loss (details in Appendix A), the MIN-SNR weight (Hang et al. 2023), and our proposed approach which uses uniform weight in an interval.

- Uniform: $\forall t \in [1, T], w(t) = 1$
- ELBO: $\forall t \in [1, T], w(t) = \frac{1}{2\sigma_t^2} \frac{\bar{\alpha}_{t-1}}{(1-\bar{\alpha}_t)^2} \beta_t^2$
- Min-SNR: $\forall t \in [1, T], w(t) = \min\left(\frac{\bar{\alpha}_t}{1-\bar{\alpha}_{t-1}}, 5\right)$
- Interval: $\forall i \in [1, N-1], \forall t \in [\tau_i, \tau_{i+1}], w(t) = 1$

The results can be seen in Figure VI.8. Optimizing the ELBO bound does not converge to a correct result. In our case, Min-SNR does not perform much better than the baseline. Among all propositions, the interval approach performs the best.

VI.3.7 Stabilization

Similar to patch-based methods (Newson et al. 2014), we observe improved results when using a preprocessing stabilization step to remove the camera motion from the equation. The intuition is that the stabilization reduces the complexity of the data distribution learned by our model. While a deep architecture does not use



(a) Uniform (b) ELBO

(c) Min-SNR (d) Interval

Figure VI.8: Among the possible weighting schemes commonly used in diffusion, our approach is very simple and perform extremely well for video inpainting. The inpainting mask is not highlighted to better illustrate the problems corrected by interval training at its boundaries. The contour is easily guessed from the ELBO result. Link to video

rigid spatiotemporal cubes, and could handle small misalignments between frames, stabilizing the video is a simple but effective way to reduce temporal inconsistencies.

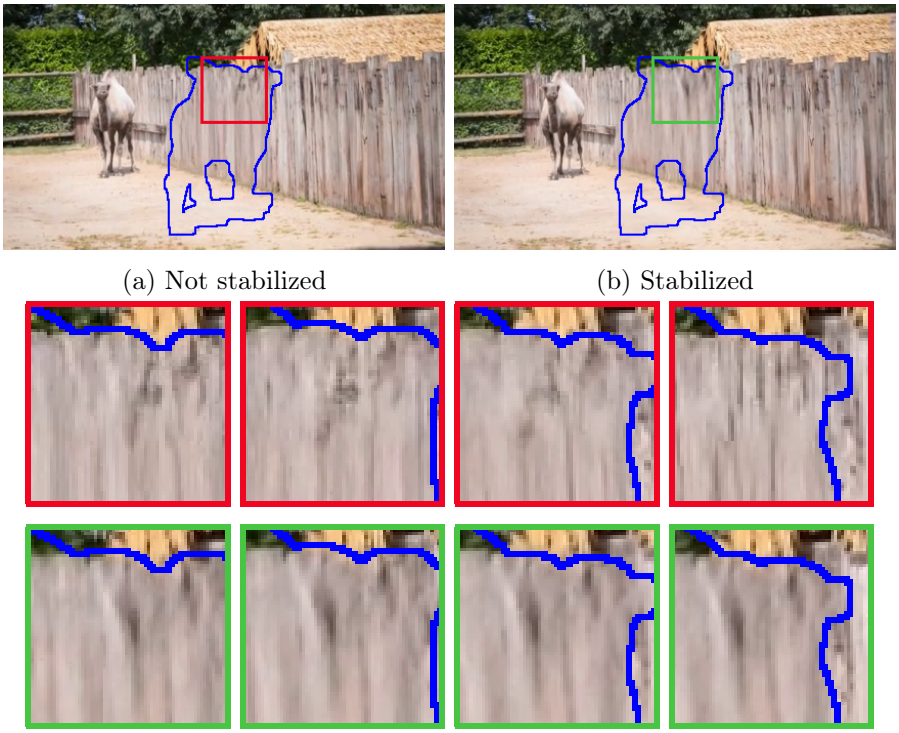
We use the method of Sánchez (2017) to estimate the transforms between frames. During training, we locally stabilize our image batch. During inference, we process the full stabilized video with padding. In Figure VI.9, we show the results with and without stabilization. We can see that the results are more stable, which is especially important for static backgrounds.

One drawback, however, is that the stabilization using homographies can't handle all types of motion and in these cases, optical flow is superior because it can handle any type of motion.

VI.4 Dynamic texture synthesis

Given our architecture, we also evaluate its ability to synthesize dynamic textures. We train an unconditional network for each texture using interval training. We enforce stationarity by using only sub-regions of the original texture instead of using the whole texture during training. The reason is to remove the spatial bias due to zero padding in neural networks which can be useful for structured images but not in our case (Shaham et al. 2019; Xu et al. 2021).

We use the texture dataset from Tesfaldet et al. (2018), which consists of 59 textures of moderate size: 12 frames of size 256x256. We compare our results to



(a) Not stabilized (b) Stabilized

Figure VI.9: Inpainting results, with and without stabilization. The stabilization removes small temporal inconsistencies. Better appreciated when played as video

the three recent papers: Xie et al. (2017), Tesfaldet et al. (2018), and DyNCA (Pajouheshgar et al. 2023).

We evaluate the synthesis using SVFID. We simply compute the features on the synthesized output and the reference using a single sample. We do not measure the diversity of outputs. We report these results in Table VI.6. Quantitatively, we perform worse than the other methods designed for dynamic texture synthesis. We hypothesize that this is due to two differences: first, we assume a strict stationarity which is sometimes not verified (Figure VI.10), and we do not enforce a similar distribution during inference while SVFID measures the distances between the distributions of the features. We also include the PSNR using the reference as ground truth, and it tells a different story. The method of Xie et al. (2017) produces results much closer to the reference than the other methods, which makes one wonder how diverse their method really is.

Qualitatively, we observe similar results for most textures (Figure VI.10). Our synthesis may not respect the original structure nor the original distribution of elements in the image.

VI.4.1 Limitations and failure cases

Our method has two inherent limitations. The first one, is that our method relies on a convolutional architecture, which has a limited receptive field, especially in the

Method	SVFID ↓	PSNR (dB)
Xie et al. (2017)	**4.47**	22.22
Tesfaldet et al. (2018)	6.70	14.41
DyNCA (2023)	8.54	14.02
Infusion	9.95	14.37

Table VI.6: Evaluation of dynamic texture synthesis. The SVFID measures the unstructured distance to the ground truth and the PSNR measures the structured distance to the ground truth. Low PSNR is not the goal, but high PSNR is suspicious.

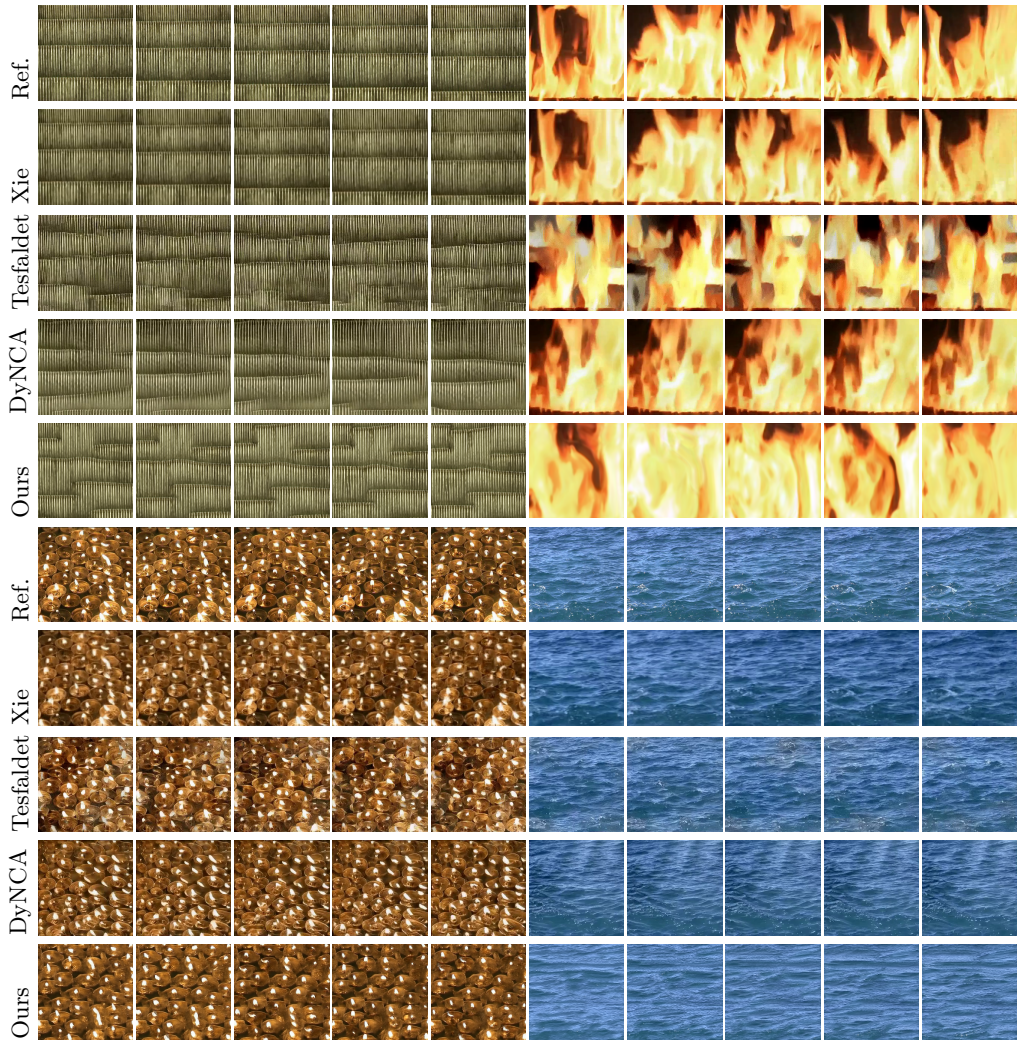


Figure VI.10: Results of dynamic texture synthesis. Our approach produces locally coherent content: the motion and visual appearance are good, but do not respect the global geometry. The results from Xie et al. (2017) are very close to the reference video. Link to videos

(a) Frame #1 (b) Frame #55

Figure VI.11: (a-b) Our method does not guarantee long-term temporal consistency when generating unseen content.

temporal dimension, because we avoid subsampling in this direction. The long-term visible coherence along the temporal axis is mostly established during the learning phase by the network's weights, not by its architecture. This failure mode is most noticeable when a purely generative guess has to be made, a region is never seen in the whole video, then it is very unlikely that our method will generate a still textured background. In fact, the visual appearance may be the same but it is unlikely to be the same texture sample for all frames. We show an example in Figure VI.11 where the inpainted region changes between the beginning and the end of the video which are more than 50 frames apart. This is not the case with many optical flow-based approaches, which are designed to promote stability.

The second limitation is due to the nature of our approach: we have to train a new neural network for each mask / video pair. We mitigated this aspect by designing a training algorithm - interval training - with this in mind. Knowing that each model will be used only once, we have maximized its usefulness. Our execution time for completing a video is therefore longer than other deep learning based approaches such as E2FGVI (Li et al. 2022b) or ISVI (Zhang et al. 2022c).

VI.5 Conclusion

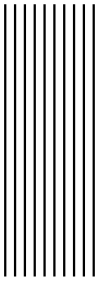
In this chapter, we have presented a method for video inpainting that uses the diffusion framework. We overcome the lengthy training and large model size associated with diffusion models by training a lightweight model on a single video. This allows for diffusion that is possible on consumer hardware and still produces very satisfying results both quantitatively and qualitatively. For dynamic textures, we show a dramatic improvement over current methods. We also propose an adapted training scheme that is particularly suitable for single video inpainting, and produces better results than the baseline training. Our method has some shortcomings: it must be trained from scratch for each new video, and the results are less accurate than optical flow-based methods for still background inpainting. Nevertheless, we see several directions of improvement for future work.

Designing a video-specific model. In this work, we have applied a diffusion model to the problem of video inpainting without any special architecture or

knowledge of video specifics. Different solutions are possible: it could be to add optical flow to the equation, or to combine two different models: a first one suitable for background inpainting and a second one for dynamic textures. Splitting the problem may not be easy but it is probably a fruitful direction for very high quality results.

Imbalanced learning. We train our diffusion model on all the visible regions of the video. In theory, it is then capable of inpainting any region. In practice, the test region is spatially limited, and its inpainting does not require all this capacity. In the case of neural networks, training on too much data with a limited network will degrade the results. While it is impossible to know beforehand which regions are important for the final inpainting, after a few steps of reverse diffusion, the occluded region already has a coarse completion. It is then possible to adjust the learning goal and focus learning on regions similar to the completion.

In addition to these two main directions, other challenges remain, such as handling high-resolution videos, or accurately evaluating the results without human intervention. Immediate improvements in diffusion could also be addressed by better architectures, faster diffusion processes, multi-scale approaches, etc.



Conclusion

In this thesis, we presented several works on internal methods for the generation and inpainting of images and videos. In the internal setting, data is limited and the main goal is to exploit the self-similarity of the image / video. This is explicit in patch-based methods, and indirect in learning-based internal approaches. The attention mechanism combines both patch-based and learning-based approaches. We summarize here our contributions to these three aspects of internal methods.

In chapter III, we considered the problem of single-image generation. This problem has been introduced by Shaham et al. (2019), who extend the texture synthesis problem to non-stationary images. In their approach, they train a multi-scale Generative Adversarial Network and generate new variations of a reference image. It turns out that in this setting, a patch-based method is competitive with this deep learning approach. We show that minimizing an energy similar to that of Kwatra et al. (2005) leads to good results without loss of detail or network artifacts. To enforce similar patch distributions, *i.e.* fidelity, between the generated samples and the reference image, we turn to optimal transport for the first coarsest scales. Once correctly initialized, the main structure of the image is fixed, and our global energy term is quickly minimized thanks to an approximate nearest neighbor search. We have thus shown that our method can produce convincing samples without using a neural network.

In chapter IV, we have developed a very efficient attention layer. The complexity of the default attention layer (Vaswani et al. 2017) scales quadratically with the number of input elements. This is incompatible with high-resolution images and videos. This problem applies to computation time, but more importantly to memory, which is a hard limit on GPUs. Exploiting the link between attention computation and nearest neighbor search, we have proposed a new attention layer called "Patch-based Stochastic Attention Layer". We approximate the attention by focusing on the nearest neighbors, which are those with the largest weights in the attention layer. Our layer is based on the PatchMatch algorithm (Barnes et al. 2009), parallelized and modified for deep learning. A key modification is to ensure end-to-end differentiability, which is of paramount importance for deep learning applications. We show that our layer has negligible memory impact compared to the standard attention layer, as the memory complexity is linear compared to quadratic. While not universally applicable, our layer is a good approximation in many image editing problems such

as image inpainting, image colorization, or single-image super-resolution. We obtain competitive results compared to other efficient attention layers.

In chapter V, we have investigated the use of diffusion models for image inpainting. The diffusion framework is particularly interesting for modeling the diversity that is inherent in image inpainting. We approach the task by training a small neural network on the test image only. Thanks to the lightweight architecture, the learning phase is relatively fast although not negligible. Despite limited data, we obtain high-quality and diverse results on texture images. These results are competitive with patch-based methods, recent inpainting networks, and extremely large diffusion models. We also discuss some of the common metrics for image inpainting, and show their limitations. In particular, PSNR is not a reliable metric when stochasticity is involved. On the other hand, LPIPS correlates better with the perceptual quality. A final advantage of our tiny neural network is that its size allows interactive use.

Finally, in chapter VI, we extend our work on diffusion models to video inpainting. As of our knowledge, it is the first work on video inpainting using diffusion models. Internal learning is particularly well suited to videos because they are highly redundant. This explains why very good results are obtained using internal data exploited by patches or optical flow. In our method, we train a diffusion model for each video, which is therefore perfectly adapted and specialized for the task. Aware that most of our networks will be used only a few times, we propose an adapted learning strategy, called "interval training". In interval training, the inference and learning are intertwined. Instead of training a model for all timesteps and then performing the inference for all those timesteps, we work iteratively on subsets of timesteps. On each interval, we train and then perform inference. These small intervals make the learning task easier. This strategy proves to be very effective in improving the quality of the results while keeping the network size constant. We then demonstrate the performance of our method on dynamic texture inpainting and complex video inpainting. Compared to other methods, our approach is very effective in dealing with these difficult situations.

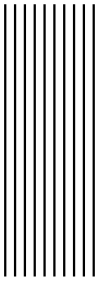
To complete the perspectives proposed at the end of each chapter, we present here three ideas of interest:

Lightweight diffusion models with attention. We have proposed in chapter V a tiny fully convolutional diffusion model. However, the images we tested on were highly self-similar. In this situation, patch-based methods (Newson et al. 2017) are well adapted, and attention layers are also a great fit. It would be very interesting to try to incorporate attention layers in this simple setting. It could probably reduce even more the number of parameters required to solve this inpainting problem; patch-based methods are efficient with no parameters at all. It is also a good case study for understanding the role of attention layers; the network is small enough to try to interpret it. Furthermore, the diffusion process covers a wide range of noise levels with probably different interpretations.

Fast and efficient video inpainting. In our experiments, we have used a diffusion model to handle all aspects of the inpainting task, but this is suboptimal.

In fact, static background inpainting can be solved very efficiently with optical flow. For simple camera motions and reasonable occlusions, following the optical flow is very accurate and runs in real time (Bokov and Vatolin 2018). The remaining part consists of dynamic backgrounds, complex motions, and regions that are never revealed during the whole sequence. For these cases, the method developed in chapter VI is well adapted. Using simple models whenever possible and resorting to complex networks only when necessary is probably a safe strategy to achieve fast and high-quality video inpainting. The major challenge of this approach is to design an algorithm that correctly splits the tasks and combines the results. A second challenge for our diffusion model is that its training time does not depend on the occlusion size. Right now, we train a neural network on a single video almost independently of the masked region, the only difference is that no loss is computed in the occlusion. Compared to patch-based methods that iterate only over the missing pixels, our training scheme does not benefit from an "almost" complete video. Small occlusions are probably easier to inpaint and might require less training, but this is not reflected in our current method. This aspect is therefore of interest for a two-step approach.

Unconditional training for inpainting. In the last two chapters on diffusion, we trained models from scratch. We decided to use a conditional network for inpainting following Saharia et al. (2022b). The conditional information is probably helpful for better end-results and faster convergence. The alternative is to use an unconditional model and adapt the inference process for inpainting (Kawar et al. 2022a; Lugmayr et al. 2022). Some of our early experiments suggest that unconditional training is possible and sometimes faster to train too. However, we have to be careful because in the case of internal learning, we are training on degraded and incomplete data. This approach is similar to the setting of Deep Image Prior (Ulyanov et al. 2018) where a neural network learns on degraded data only, in a self-supervised manner.



Bibliography

Adobe Firefly (2023). https://www.adobe.com/fr/sensei/generative-ai/firefly.html (cit. on pp. 4, 81).

Alkobi, N., T. Rott Shaham, and T. Michaeli (2023). "Internal Diverse Image Completion". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Vancouver, BC, Canada: IEEE, pp. 648–658. ISBN: 9798350302493. DOI: 10.1109/CVPRW59228.2023.00072 (cit. on pp. 25, 70, 91).

Andersson, P. et al. (2020). "FLIP: A Difference Evaluator for Alternating Images." In: *Proc. ACM Comput. Graph. Interact. Tech.* 3.2, pp. 15–1 (cit. on p. 77).

Arias, P., V. Caselles, and G. Facciolo (2012). "Analysis of a Variational Framework for Exemplar-Based Image Inpainting". en. In: *Multiscale Modeling & Simulation* 10.2, pp. 473–514. ISSN: 1540-3459 (cit. on p. 12).

Arjovsky, M., S. Chintala, and L. Bottou (2017). "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, pp. 214–223 (cit. on p. 14).

Arnab, A. et al. (2021). "ViViT: A Video Vision Transformer". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Montreal, QC, Canada: IEEE, pp. 6816–6826. ISBN: 978-1-66542-812-5. DOI: 10.1109/ICCV48922.2021.00676 (cit. on p. 42).

Bahdanau, D., K. Cho, and Y. Bengio (2015). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun (cit. on p. 18).

Balaji, Y. et al. (2023). "eDiff-I: Text-to-Image Diffusion Models with an Ensemble of Expert Denoisers". In: arXiv:2211.01324. arXiv: 2211.01324 [cs] (cit. on p. 92).

Barnes, C. and E. Shechtman (2010). "The generalized patchmatch correspondence algorithm". In: *(ECCV) European Conference on Computer Vision*, pp. 29–43 (cit. on pp. 11, 36, 43, 47).

Barnes, C. and F.-L. Zhang (2017). "A Survey of the State-of-the-Art in Patch-Based Synthesis". In: *Computational Visual Media* 3.1, pp. 3–20. ISSN: 2096-0433, 2096-0662. DOI: 10.1007/s41095-016-0064-2 (cit. on p. 11).

Barnes, C. et al. (2009). "PatchMatch: a randomized correspondence algorithm for structural image editing". In: *SIGGRAPH 2009*. DOI: 10.1145/1531326.1531330 (cit. on pp. vi, 6, 24, 26, 28, 39, 40, 43, 45, 87, 109).

Batson, J. and L. Royer (n.d.). "Noise2Self: Blind Denoising by Self-Supervision". In: () (cit. on pp. 13, 25).

Bello, I. (2021). "LambdaNetworks: Modeling long-range Interactions without Attention". In: *International Conference on Learning Representations* (cit. on p. 42).

Beltagy, I., M. E. Peters, and A. Cohan (2020). *Longformer: The Long-Document Transformer*. arXiv: 2004.05150 [cs] (cit. on p. 42).

Bergmann, U., N. Jetchev, and R. Vollgraf (2017). "Learning Texture Manifolds with the Periodic Spatial GAN". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by D. Precup and Y. W. Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 469–477 (cit. on p. 16).

Bertalmio, M. et al. (2000). "Image Inpainting". In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '00. USA: ACM Press/Addison-Wesley Publishing Co., pp. 417–424. ISBN: 978-1-58113-208-3. DOI: 10/dcvpvb (cit. on pp. 5, 70).

Beyer, L. et al. (2023). "FlexiViT: One Model for All Patch Sizes". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vancouver, BC, Canada: IEEE, pp. 14496–14506. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.01393 (cit. on p. 66).

Blalock, D. W. et al. (2020). "What is the State of Neural Network Pruning?" In: *MLSys*. Ed. by I. S. Dhillon, D. S. Papailiopoulos, and V. Sze. mlsys.org (cit. on p. 82).

Blattmann, A. et al. (2023). "Align Your Latents: High-Resolution Video Synthesis With Latent Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575 (cit. on p. 86).

Blau, Y. and T. Michaeli (2018). "The Perception-Distortion Tradeoff". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA: IEEE, pp. 6228–6237. ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00652 (cit. on pp. 76, 78).

Bokov, A. and D. Vatolin (2018). "100+ Times Faster Video Completion by Optical-Flow-Guided Variational Refinement". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 2122–2126. DOI: 10/gg2ntf (cit. on pp. 87, 95, 96, 111).

Bornard, R. et al. (2002). "Missing Data Correction in Still Images and Image Sequences". In: *Proceedings of the Tenth ACM International Conference on Multimedia*. MULTIMEDIA '02. New York, NY, USA: Association for Computing Machinery, pp. 355–361. ISBN: 978-1-58113-620-3. DOI: 10.1145/641007.641084 (cit. on pp. 11, 70).

Buades, A., B. Coll, and J.-M. Morel (2005). "A Non-Local Algorithm for Image Denoising". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2, 60–65 vol. 2. DOI: 10.1109/CVPR.2005.38 (cit. on pp. 12, 19).

Bugeau, A. et al. (2010). "A Comprehensive Framework for Image Inpainting". In: *IEEE Transactions on Image Processing* 19.10, pp. 2634–2645. DOI: 10.1109/TIP.2010.2049240 (cit. on p. 11).

Cai, J. et al. (2022). "DeViT: Deformed Vision Transformers in Video Inpainting". In: *Proceedings of the 30th ACM International Conference on Multimedia*. MM '22. New York, NY, USA: Association for Computing Machinery, pp. 779–789. ISBN: 978-1-4503-9203-7. DOI: 10.1145/3503161.3548395 (cit. on p. 88).

Calian, D. A. et al. (2019). "SCRAM: Spatially Coherent Randomized Attention Maps". en. In: *arXiv:1905.10308 [cs, stat]*. arXiv: 1905.10308 (cit. on p. 41).

Cao, F. et al. (2011). "Geometrically Guided Exemplar-Based Inpainting". In: *SIAM Journal on Imaging Sciences* 4.4, pp. 1143–1179. ISSN: 1936-4954. DOI: 10/cvpdfk (cit. on p. 11).

Carreira, J. and A. Zisserman (2017). "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4724–4733. DOI: 10.1109/CVPR.2017.502 (cit. on p. 93).

Ceylan, D., C.-H. P. Huang, and N. J. Mitra (2023). "Pix2Video: Video Editing Using Image Diffusion". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23206–23217 (cit. on p. 86).

Chai, W. et al. (2023). "StableVideo: Text-driven Consistency-aware Diffusion Video Editing". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23040–23050 (cit. on p. 86).

Chang, H. et al. (2022). "MaskGIT: Masked Generative Image Transformer". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 21).

Chang, Y.-L. et al. (2019). "Free-Form Video Inpainting With 3D Gated Convolution and Temporal PatchGAN". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9066–9075 (cit. on p. 88).

Chatillon, P., Y. Gousseau, and S. Lefebvre (2022). "A Statistically Constrained Internal Method for Single Image Super-Resolution". In: *2022 26th International Conference on Pattern Recognition (ICPR)*. Montreal, QC, Canada: IEEE, pp. 1322–1328. ISBN: 978-1-66549-062-7. DOI: 10.1109/ICPR56361.2022.9956498 (cit. on p. 25).

— (2023). "A Geometrically Aware Auto-Encoder for Multi-texture Synthesis". In: *Scale Space and Variational Methods in Computer Vision*. Ed. by L. Calatroni et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 263–275. ISBN: 978-3-031-31975-4. DOI: 10.1007/978-3-031-31975-4_20 (cit. on p. 16).

Chefer, H. et al. (2023). "Attend-and-Excite: Attention-Based Semantic Guidance for Text-to-Image Diffusion Models". In: *ACM Trans. Graph.* 42.4. ISSN: 0730-0301. DOI: 10.1145/3592116 (cit. on p. 21).

Chen, C.-F., R. Panda, and Q. Fan (2022). "RegionViT: Regional-to-Local Attention for Vision Transformers". In: *International Conference on Learning Representations* (cit. on p. 41).

Chen, M. et al. (2020). "Generative Pretraining From Pixels". In: *Proceedings of the 37th International Conference on Machine Learning*. PMLR, pp. 1691–1703 (cit. on p. 20).

Cherel, N. et al. (2022a). "A Patch-Based Algorithm for Diverse and High Fidelity Single Image Generation". In: *2022 IEEE International Conference on Image Processing (ICIP)*. Oral presentation, pp. 3221–3225. DOI: 10.1109/ICIP46576.2022.9897913 (cit. on p. 7).

Cherel, N. et al. (2022b). "Attention stochastique basée patchs pour l'édition d'images". In: *28° Colloque sur le traitement du signal et des images*. 001-0302. Nancy: GRETSI - Groupe de Recherche en Traitement du Signal et des Images, p. 1209–1212 (cit. on p. 7).

Cherel, N. et al. (2023a). *Infusion: Internal Diffusion for Video Inpainting*. In preparation. arXiv: 2311.01090 [cs] (cit. on p. 7).

— (2023b). "Modèle de Diffusion Frugal Pour l'inpainting d'images". In: *GRETSI 2023 :XXIXème Colloque Francophone de Traitement Du Signal et Des Images*. Grenoble, France (cit. on p. 7).

— (2024). "Patch-Based Stochastic Attention for Image Editing". In: *Computer Vision and Image Understanding* 238, p. 103866. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2023.103866 (cit. on p. 7).

Child, R. et al. (2019). "Generating Long Sequences with Sparse Transformers". In: *ArXiv* (cit. on pp. 39, 41).

Choromanski, K. M. et al. (2020). "Rethinking Attention with Performers". In: *International Conference on Learning Representations* (cit. on pp. 39, 42, 48, 52).

Chung, H. et al. (2022). "Diffusion Posterior Sampling for General Noisy Inverse Problems". In: *The Eleventh International Conference on Learning Representations* (cit. on pp. 18, 70).

Chung, H. et al. (2023). "Diffusion Posterior Sampling for General Noisy Inverse Problems". In: *(ICLR) International Conference on Learning Representations*, pp. 1–28. arXiv: 2209.14687 (cit. on p. 86).

Correia, G. M., V. Niculae, and A. F. T. Martins (2019). "Adaptively Sparse Transformers". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by K. Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 2174–2184. DOI: 10.18653/v1/D19-1223 (cit. on p. 42).

Costantini, R., L. Sbaiz, and S. Susstrunk (2008). "Higher Order SVD Analysis for Dynamic Texture Synthesis". In: *IEEE Transactions on Image Processing* 17.1, pp. 42–52. DOI: 10.1109/TIP.2007.910956 (cit. on p. 88).

Criminisi, A., P. Perez, and K. Toyama (2004). "Region Filling and Object Removal by Exemplar-Based Image Inpainting". In: *IEEE Transactions on Image Processing* 13.9, pp. 1200–1212. ISSN: 1057-7149. DOI: 10/fbk8ft (cit. on pp. 5, 11, 12, 21, 24, 40, 56, 70, 87).

Dai, T. et al. (2019a). "Second-Order Attention Network for Single Image Super-Resolution". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern*

Recognition (CVPR). Long Beach, CA, USA: IEEE, pp. 11057–11066. ISBN: 978-1-72813-293-8. DOI: 10.1109/CVPR.2019.01132 (cit. on p. 40).

Dai, Z. et al. (2019b). "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context". In: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Ed. by A. Korhonen, D. R. Traum, and L. Marquez. Association for Computational Linguistics, pp. 2978–2988. DOI: 10.18653/V1/P19-1285 (cit. on p. 42).

Darabi, S. et al. (2012). "Image Melding". In: *ACM Trans. Graph.* DOI: 10.1145/2185520.2185578 (cit. on pp. 12, 36, 37, 40).

Dhariwal, P. and A. Nichol (2021). "Diffusion Models Beat GANs on Image Synthesis". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., pp. 8780–8794 (cit. on p. 15).

Ding, K. et al. (2020). "Image Quality Assessment: Unifying Structure and Texture Similarity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2020.3045810. arXiv: 2004.07728 [cs] (cit. on p. 77).

Doretto, G., E. Jones, and S. Soatto (2004). "Spatially homogeneous dynamic textures". In: *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part II 8*. Springer, pp. 591–602 (cit. on p. 88).

Doretto, G. et al. (2003). "Dynamic textures". In: *International journal of computer vision* 51, pp. 91–109 (cit. on p. 88).

Dorkenwald, M. et al. (2021). "Stochastic Image-to-Video Synthesis Using cINNs". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3742–3753 (cit. on p. 89).

Dosovitskiy, A. et al. (2020). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR 2021* (cit. on pp. 20, 62, 66).

Drori, I., D. Cohen-Or, and H. Yeshurun (2003). "Fragment-based image completion". In: *ACM Trans. Graph.* 22.3, pp. 303–312. ISSN: 0730-0301. DOI: 10.1145/882262.882267 (cit. on pp. 11, 87).

Duggal, S. et al. (2019). "DeepPruner: Learning Efficient Stereo Matching via Differentiable PatchMatch". en. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 4383–4392. ISBN: 978-1-72814-803-8 (cit. on p. 64).

Efros, A. and T. Leung (1999). "Texture Synthesis by Non-Parametric Sampling". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1033–1038 vol.2. ISBN: 978-0-7695-0164-2. DOI: 10.1109/ICCV.1999.790383 (cit. on pp. 2, 10, 11, 23, 24, 70).

Ehret, T. and P. Arias (2018). "On the Convergence of PatchMatch and Its Variants". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, pp. 1121–1129. ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00123 (cit. on p. 66).

Esser, P., R. Rombach, and B. Ommer (2021). "Taming Transformers for High-Resolution Image Synthesis". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, pp. 12873–12883. DOI: 10.1109/CVPR46437.2021.01268 (cit. on p. 20).

Fineman, M. (2012). *Faking It: Manipulated Photography Before Photoshop*. Yale University Press (cit. on p. 4).

Frankle, J. and M. Carbin (2023). "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". In: *International Conference on Learning Representations* (cit. on p. 82).

Freeman, W., T. Jones, and E. Pasztor (2002). "Example-Based Super-Resolution". In: *IEEE Computer Graphics and Applications* 22.2, pp. 56–65. ISSN: 1558-1756. DOI: 10.1109/38.988747 (cit. on p. 40).

Frigo, O. et al. (2016). "Split and Match: Example-Based Adaptive Patch Sampling for Unsupervised Style Transfer". In: *(CVPR) Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 553–561. ISBN: 978-1-4673-8851-1 (cit. on p. 40).

Funke, C. M. et al. (2017). "Synthesising Dynamic Textures using Convolutional Neural Networks". In: *CoRR* abs/1702.07006. arXiv: 1702.07006 (cit. on p. 89).

Galerne, B., A. Leclaire, and J. Rabin (2018). "A Texture Synthesis Model Based on Semi-Discrete Optimal Transport in Patch Space". In: *SIAM Journal on Imaging Sciences* 11.4, pp. 2456–2493. DOI: 10.1137/18M1175781 (cit. on pp. 10, 24).

Gao, C. et al. (2020). "Flow-Edge Guided Video Completion". In: *Computer Vision – ECCV 2020*. Ed. by A. Vedaldi et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 713–729. ISBN: 978-3-030-58610-2. DOI: 10.1007/978-3-030-58610-2_42 (cit. on pp. 88, 96, 97).

Gatys, L., A. S. Ecker, and M. Bethge (2015). "Texture Synthesis Using Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc. (cit. on pp. 16, 19, 36, 78, 89).

Glasner, D., S. Bagon, and M. Irani (2009). "Super-Resolution from a Single Image". In: *2009 IEEE 12th International Conference on Computer Vision*, pp. 349–356. DOI: 10/dqph3d (cit. on pp. 12, 35, 40, 58).

Goodfellow, I. et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc. (cit. on pp. 13, 24, 69, 70).

Granados, M. et al. (2012). "How Not to Be Seen — Object Removal from Videos of Crowded Scenes". In: *Computer Graphics Forum* 31.2pt1, pp. 219–228. ISSN: 0167-7055. DOI: 10/f996wc (cit. on pp. 87, 96).

Granot, N. et al. (2022). "Drop the GAN: In Defense of Patches Nearest Neighbors As Single Image Generative Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13460–13469 (cit. on pp. 11, 24, 28, 31).

Gu, B. et al. (2023). *Flow-Guided Diffusion for Video Inpainting*. arXiv: 2311.15368 [cs] (cit. on p. 88).

Guo, S. et al. (2022). "U-Attention to Textures: Hierarchical Hourglass Vision Transformer for Universal Texture Synthesis". In: *Proceedings of the 19th ACM SIGGRAPH European Conference on Visual Media Production*. CVMP '22. London, United Kingdom: Association for Computing Machinery. ISBN: 9781450399395. DOI: 10.1145/3565516.3565525 (cit. on p. 19).

Guo, Y. et al. (2024). "AnimateDiff: Animate Your Personalized Text-to-Image Diffusion Models without Specific Tuning". In: *The Twelfth International Conference on Learning Representations* (cit. on p. 86).

Gur, S., S. Benaim, and L. Wolf (2020). "Hierarchical Patch VAE-GAN: Generating Diverse Videos from a Single Sample". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 16761–16772 (cit. on pp. 25, 93).

Guth, F. et al. (2022). "Wavelet Score-Based Generative Modeling". In: *Advances in Neural Information Processing Systems* (cit. on p. 86).

Gutierrez, J. et al. (2017). "Optimal Patch Assignment for Statistically Constrained Texture Synthesis". In: *Scale Space and Variational Methods in Computer Vision*. Ed. by F. Lauze, Y. Dong, and A. B. Dahl. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 172–183. ISBN: 978-3-319-58771-4. DOI: 10.1007/978-3-319-58771-4_14 (cit. on pp. 10, 24).

HaCohen, Y. et al. (2011). "Non-Rigid Dense Correspondence with Applications for Image Enhancement". In: *ACM Transactions on Graphics* 30.4, 70:1–70:10. ISSN: 0730-0301. DOI: 10.1145/2010324.1964965 (cit. on p. 37).

Haim, N. et al. (2022). "Diverse Generation from a Single Video Made Possible". In: *Computer Vision – ECCV 2022*. Ed. by S. Avidan et al. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, pp. 491–509. ISBN: 978-3-031-19790-1. DOI: 10.1007/978-3-031-19790-1_30 (cit. on pp. 11, 24).

Hamazaspyan, M. and S. Navasardyan (2023). "Diffusion-Enhanced PatchMatch: A Framework for Arbitrary Style Transfer With Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 797–805 (cit. on p. 64).

Hang, T. et al. (2023). "Efficient Diffusion Training via Min-SNR Weighting Strategy". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7441–7451 (cit. on pp. 90, 92, 102).

Hao, G. et al. (2023). "Diffusion-Based Holistic Texture Rectification and Synthesis". In: *SIGGRAPH Asia 2023 Conference Papers*. SA '23. Sydney, Australia: Association for Computing Machinery. ISBN: 9798400703157. DOI: 10.1145/3610548.3618233 (cit. on p. 70).

Harvey, W. et al. (2022). "Flexible Diffusion Modeling of Long Videos". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., pp. 27953–27965 (cit. on p. 86).

Hays, J. and A. A. Efros (2007). "Scene Completion Using Millions of Photographs". In: *ACM Transactions on Graphics* 26.3, 4–es. ISSN: 0730-0301. DOI: 10/cm8hqq (cit. on pp. 24, 70).

He, K. and J. Sun (2012). "Statistics of Patch Offsets for Image Completion". In: *Computer Vision – ECCV 2012*. Ed. by A. Fitzgibbon et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 16–29. ISBN: 978-3-642-33709-3. DOI: 10/gfvgqz (cit. on p. 12).

He, K. et al. (2022). "Masked Autoencoders Are Scalable Vision Learners". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009 (cit. on p. 21).

Herling, J. and W. Broll (2014). "High-Quality Real-Time Video Inpaintingwith PixMix". In: *IEEE Transactions on Visualization and Computer Graphics* 20.6, pp. 866–879. ISSN: 1941-0506. DOI: 10/f58vvq (cit. on pp. 87, 95, 96).

Hertz, A. et al. (2023). "Prompt-to-Prompt Image Editing with Cross-Attention Control". In: *The Eleventh International Conference on Learning Representations* (cit. on p. 21).

Hertzmann, A. et al. (2001). "Image Analogies". In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, pp. 327–340. ISBN: 978-1-58113-374-5. DOI: 10/dbjqnf (cit. on p. 35).

Heusel, M. et al. (2017). "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *Advances in neural information processing systems* 30 (cit. on p. 30).

Hinton, G., O. Vinyals, and J. Dean (2015). "Distilling the Knowledge in a Neural Network". In: *NIPS Deep Learning and Representation Learning Workshop* (cit. on p. 82).

Hinz, T. et al. (2021). "Improved Techniques for Training Single-Image GANs". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1300–1309 (cit. on p. 25).

Ho, J., A. Jain, and P. Abbeel (2020). "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 6840–6851 (cit. on pp. v, vi, 6, 14, 15, 18, 21, 69, 71, 72, 85, 86, 89, 90, 102, 137).

Ho, J. and T. Salimans (2021). "Classifier-Free Diffusion Guidance". In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications* (cit. on p. 18).

Ho, J. et al. (2022a). "Cascaded Diffusion Models for High Fidelity Image Generation". In: *The Journal of Machine Learning Research* 23.1, 47:2249–47:2281. ISSN: 1532-4435 (cit. on pp. 15, 86).

Ho, J. et al. (2022b). "Imagen Video: High Definition Video Generation with Diffusion Models". In: arXiv:2210.02303. DOI: 10.48550/arXiv.2210.02303. arXiv: 2210.02303 [cs] (cit. on pp. 85, 86, 90).

Ho, J. et al. (2022c). "Video Diffusion Models". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., pp. 8633–8646 (cit. on pp. 15, 85, 86).

Houdard, A. et al. (2021). "Wasserstein Generative Models for Patch-based Texture Synthesis". In: *Scale Space and Variational Methods in Computer Vision*. Vol. LNCS 12679. Cabourg, France, pp. 269–280 (cit. on pp. 10, 24, 28, 30).

Hu, E. J. et al. (2022). "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations* (cit. on p. 83).

Hu, Y.-T. et al. (2020). "Proposal-Based Video Completion". In: *Computer Vision – ECCV 2020*. Ed. by A. Vedaldi et al. Cham: Springer International Publishing, pp. 38–54. ISBN: 978-3-030-58583-9 (cit. on p. 88).

Huang, J.-B. et al. (2016). "Temporally Coherent Completion of Dynamic Video". In: *ACM Transactions on Graphics* 35.6, pp. 1–11. ISSN: 0730-0301, 1557-7368. DOI: 10.1145/2980179.2982398 (cit. on p. 87).

Huang, T. et al. (2021). "Neighbor2Neighbor: Self-Supervised Denoising from Single Noisy Images". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, pp. 14776–14785. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.01454 (cit. on p. 25).

Iizuka, S., E. Simo-Serra, and H. Ishikawa (2017). "Globally and locally consistent image completion". In: *ACM Transactions on Graphics* 36.4, 107:1–107:14. ISSN: 0730-0301 (cit. on pp. 17, 70).

Isola, P. et al. (2017). "Image-to-Image Translation with Conditional Adversarial Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976. DOI: 10/gfrfv9 (cit. on p. 25).

Jaegle, A. et al. (2021). "Perceiver: General Perception with Iterative Attention". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 4651–4664 (cit. on p. 42).

Jetchev, N., U. Bergmann, and R. Vollgraf (2016). *Texture Synthesis with Spatial Generative Adversarial Networks* (cit. on p. 16).

Jiang, Y., S. Chang, and Z. Wang (2021). "Transgan: Two pure transformers can make one strong gan, and that can scale up". In: *Advances in Neural Information Processing Systems* 34 (cit. on p. 20).

Karras, T., S. Laine, and T. Aila (2019). "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410 (cit. on pp. 3, 14, 16, 37).

Karras, T. et al. (2020). "Analyzing and improving the image quality of stylegan". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119 (cit. on p. 37).

Kaspar, A. et al. (2015). "Self Tuning Texture Optimization". In: *Computer Graphics Forum* 34.2, pp. 349–359. ISSN: 1467-8659. DOI: 10.1111/cgf.12565 (cit. on p. 10).

Kasten, Y. et al. (2021). "Layered Neural Atlases for Consistent Video Editing". In: *ACM Transactions on Graphics* 40.6, 210:1–210:12. ISSN: 0730-0301. DOI: 10.1145/3478513.3480546 (cit. on p. 86).

Katharopoulos, A. et al. (2020). "Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention". In: *Proceedings of the International Conference on Machine Learning (ICML)* (cit. on pp. 39, 42, 48–50, 52, 54).

Kawar, B. et al. (2022a). "Denoising Diffusion Restoration Models". In: *Advances in Neural Information Processing Systems* 35, pp. 23593–23606 (cit. on pp. 18, 69, 70, 111).

— (2022b). "Denoising Diffusion Restoration Models". In: *ICLR Workshop on Deep Generative Models for Highly Structured Data*. Vol. 2020-Decem. arXiv: 2201.11793 (cit. on p. 86).

Kim, D. et al. (2019). "Deep Video Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5792–5801 (cit. on pp. 87, 97, 99).

Kim, S. Y. et al. (2022). "Zoom-to-Inpaint: Image Inpainting With High-Frequency Details". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 477–487 (cit. on p. 18).

King, D. (1997). *The Commissar Vanishes: The Falsification of Photographs and Art in Stalin's Russia*. Metropolitan Books (cit. on p. 4).

Kingma, D. P. and M. Welling (2014). "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun (cit. on p. 14).

Kitaev, N., L. Kaiser, and A. Levskaya (2020). "Reformer: The Efficient Transformer". In: *ICLR* (cit. on pp. 41, 44, 49, 52, 54).

Ko, K. and C.-S. Kim (2023). "Continuously Masked Transformer for Image Inpainting". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13169–13178 (cit. on p. 22).

Kong, Z. et al. (2020). "DiffWave: A Versatile Diffusion Model for Audio Synthesis". In: *International Conference on Learning Representations* (cit. on p. 15).

Korman, S. and S. Avidan (2011). "Coherency Sensitive Hashing". In: *2011 International Conference on Computer Vision*, pp. 1607–1614. DOI: 10.1109/ICCV.2011.6126421 (cit. on p. 36).

Krull, A., T.-O. Buchholz, and F. Jug (2019). "Noise2Void - Learning Denoising From Single Noisy Images". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, pp. 2124–2132. ISBN: 978-1-72813-293-8. DOI: 10.1109/CVPR.2019.00223 (cit. on p. 25).

Kulikov, V. et al. (2023). "SinDDM: A Single Image Denoising Diffusion Model". In: *Proceedings of the 40th International Conference on Machine Learning*. PMLR, pp. 17920–17930 (cit. on pp. 25, 70, 87).

Kwatra, V. et al. (2003). "Graphcut textures: Image and video synthesis using graph cuts". In: *Acm transactions on graphics (tog)* 22.3, pp. 277–286 (cit. on pp. 11, 89).

Kwatra, V. et al. (2005). "Texture optimization for example-based synthesis". In: *ACM SIGGRAPH 2005 Papers*, pp. 795–802 (cit. on pp. 10, 11, 23, 24, 26, 109).

Lacoste, A. et al. (2019). *Quantifying the Carbon Emissions of Machine Learning*. DOI: 10.48550/arXiv.1910.09700. arXiv: 1910.09700 [cs] (cit. on p. 81).

Lample, G. et al. (2019). "Large Memory Layers with Product Keys". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc. (cit. on p. 41).

Le, T. T. et al. (2017). "Motion-consistent video inpainting". In: *ICIP 2017: IEEE international conference on image processing*. ICIP 2017: IEEE international conference on image processing. Beijing, China. DOI: 10/gg2ntp (cit. on pp. 87, 96).

Leclaire, A. and J. Rabin (2019). "A Fast Multi-layer Approximation to Semi-discrete Optimal Transport". In: *Scale Space and Variational Methods in Computer Vision*. Ed. by J. Lellmann, M. Burger, and J. Modersitzki. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 341–353. ISBN: 978-3-030-22368-7. DOI: 10.1007/978-3-030-22368-7_27 (cit. on pp. 10, 24).

Lee, J. H., I. Choi, and M. H. Kim (2016). "Laplacian Patch-Based Image Synthesis". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 12).

Lee, K. et al. (2022). "ViTGAN: Training GANs with Vision Transformers". In: *International Conference on Learning Representations* (cit. on p. 20).

Lee, S. et al. (2019). "Copy-and-Paste Networks for Deep Video Inpainting". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4413–4421 (cit. on pp. 88, 97).

Lefebvre, S. and H. Hoppe (2006). "Appearance-Space Texture Synthesis". In: *ACM Transactions on Graphics* 25.3, pp. 541–548. ISSN: 0730-0301. DOI: 10.1145/1141911.1141921 (cit. on p. 10).

Lehtinen, J. et al. (2018). "Noise2Noise: Learning Image Restoration without Clean Data". In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, pp. 2965–2974 (cit. on pp. 13, 25).

Lei, C., Y. Xing, and Q. Chen (2020). "Blind Video Temporal Consistency via Deep Video Prior". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., pp. 1083–1093 (cit. on p. 25).

Lei, C. et al. (2023). "Deep Video Prior for Video Consistency and Propagation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1, pp. 356–371. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2022.3142071 (cit. on p. 25).

Li, A. et al. (2020). "Short-Term and Long-Term Context Aggregation Network for Video Inpainting". In: *Computer Vision – ECCV 2020*. Ed. by A. Vedaldi et al. Cham: Springer International Publishing, pp. 728–743. ISBN: 978-3-030-58548-8 (cit. on p. 88).

Li, C. and M. Wand (2016). "Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis". In: *2016 IEEE Conference on Computer*

Vision and Pattern Recognition (CVPR). ISSN: 1063-6919, pp. 2479–2486 (cit. on p. 43).

Li, W. et al. (2022a). "MAT: Mask-Aware Transformer for Large Hole Image Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10758–10768 (cit. on p. 22).

Li, Y. et al. (2017). "Universal style transfer via feature transforms". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., pp. 385–395. ISBN: 9781510860964 (cit. on p. 16).

Li, Z. et al. (2022b). "Towards An End-to-End Framework for Flow-Guided Video Inpainting". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, pp. 17541–17550. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01704 (cit. on pp. 88, 93, 94, 96–98, 106).

Liang, J. et al. (2022). *VRT: A Video Restoration Transformer*. arXiv: 2201.12288 [cs, eess] (cit. on p. 42).

Liang, L. et al. (2001). "Real-time texture synthesis by patch-based sampling". In: *ACM Transactions on Graphics (ToG)* 20.3, pp. 127–150 (cit. on p. 10).

Liao, L. et al. (2018). "Edge-Aware Context Encoder for Image Inpainting". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3156–3160. DOI: 10.1109/ICASSP.2018.8462549 (cit. on p. 17).

Liew, J. H. et al. (2023). *MagicEdit: High-Fidelity and Temporally Coherent Video Editing*. arXiv: 2308.14749 [cs] (cit. on p. 86).

Lin, J., G. Sharma, and T. N. Pappas (2023). "Toward Universal Texture Synthesis by Combining Texton Broadcasting with Noise Injection in StyleGAN-2". In: *e-Prime - Advances in Electrical Engineering, Electronics and Energy* 3, p. 100092. ISSN: 2772-6711. DOI: 10.1016/j.prime.2022.100092 (cit. on p. 74).

Liu, D. et al. (2018a). "Non-Local Recurrent Network for Image Restoration". en. In: *Advances in Neural Information Processing Systems* 31 (cit. on pp. 40, 43, 58).

Liu, G. et al. (2018b). "Image Inpainting for Irregular Holes Using Partial Convolutions". In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari et al. Vol. 11215. Cham: Springer International Publishing, pp. 89–105. ISBN: 978-3-030-01251-9 978-3-030-01252-6. DOI: 10.1007/978-3-030-01252-6_6 (cit. on pp. 17, 70).

Liu, G. et al. (2020a). "Transposer: Universal Texture Synthesis Using Feature Maps as Transposed Convolution Filter". In: *ArXiv Preprint*. arXiv: 2007.07243 (cit. on p. 19).

Liu, H. et al. (2019). "Coherent Semantic Attention for Image Inpainting". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 4169–4178. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00427 (cit. on p. 22).

Liu, R. et al. (2021a). "Decoupled Spatial-Temporal Transformer for Video Inpainting". In: *arXiv:2104.06637 [cs]*. arXiv: 2104.06637 [cs] (cit. on p. 66).

— (2021b). "FuseFormer: Fusing Fine-Grained Information in Transformers for Video Inpainting". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14040–14049 (cit. on pp. 88, 93–95, 97).

Liu, R. et al. (2020b). "Temporal Adaptive Alignment Network for Deep Video Inpainting". In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. Yokohama, Japan: International Joint Conferences on Artificial Intelligence Organization, pp. 927–933. ISBN: 978-0-9992411-6-5. DOI: 10/gg7zr3 (cit. on p. 67).

Liu, Y. and V. Caselles (2013). "Exemplar-Based Image Inpainting Using Multiscale Graph Cuts". In: *IEEE Transactions on Image Processing* 22.5, pp. 1699–1711. ISSN: 1941-0042. DOI: 10.1109/TIP.2012.2218828 (cit. on pp. 12, 30).

Liu, Z. et al. (2021c). "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *(ICCV) International Conference on Computer Vision*, p. 11. arXiv: 2103.14030 (cit. on pp. 20, 41).

Liu, Z. et al. (2022). "Video Swin Transformer". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, pp. 3192–3201. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.00320 (cit. on p. 42).

Lu, J. (2022). "Transformer-Based Neural Texture Synthesis and Style Transfer". In: *Proceedings of the 2022 4th Asia Pacific Information Technology Conference*. APIT '22. Virtual Event, Thailand: Association for Computing Machinery, pp. 88–95. ISBN: 9781450395571. DOI: 10.1145/3512353.3512366 (cit. on p. 19).

Lugmayr, A. et al. (2022). "RePaint: Inpainting Using Denoising Diffusion Probabilistic Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11461–11471 (cit. on pp. 18, 69, 70, 72, 75, 80, 86, 90, 111).

Luo, Z. et al. (2023). "VideoFusion: Decomposed Diffusion Models for High-Quality Video Generation". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vancouver, BC, Canada: IEEE, pp. 10209–10218. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.00984 (cit. on p. 86).

Masnou, S. and J. Morel (1998). "Level Lines Based Disocclusion". In: *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*, 259–263 vol.3. DOI: 10/cs7zx5 (cit. on p. 70).

Matsushita, Y. et al. (2006). "Full-frame video stabilization with motion inpainting". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.7, pp. 1150–1163. DOI: 10.1109/TPAMI.2006.141 (cit. on p. 87).

Mei, K. and V. Patel (2023). "VIDM: Video Implicit Diffusion Models". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 37.8, pp. 9117–9125. ISSN: 2374-3468. DOI: 10.1609/aaai.v37i8.26094 (cit. on p. 86).

Mei, Y. et al. (2020). "Image Super-Resolution With Cross-Scale Non-Local Attention and Exhaustive Self-Exemplars Mining". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, pp. 5689–5698. ISBN: 978-1-72817-168-5. DOI: 10.1109/CVPR42600.2020.00573 (cit. on pp. 40, 58).

Michaeli, T. and M. Irani (2014). "Blind Deblurring Using Internal Patch Recurrence". In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 783–798. ISBN: 978-3-319-10578-9. DOI: 10/gg87x9 (cit. on pp. 40, 58).

Monga, V., Y. Li, and Y. C. Eldar (2021). "Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing". In: *IEEE Signal Processing Magazine* 38.2, pp. 18–44. ISSN: 1558-0792. DOI: 10.1109/MSP.2020.3016905 (cit. on p. 36).

Mordvintsev, A. et al. (2020). "Growing Neural Cellular Automata". In: *Distill*. https://distill.pub/2020/growing-ca. DOI: 10.23915/distill.00023 (cit. on p. 89).

Murase, R., Y. Zhang, and T. Okatani (2019). "Video-Rate Video Inpainting". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1553–1561. DOI: 10.1109/WACV.2019.00170 (cit. on p. 88).

Nazeri, K. et al. (2019). "EdgeConnect: Structure Guided Image Inpainting using Edge Prediction". en. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. Seoul, Korea (South): IEEE, pp. 3265–3274. ISBN: 978-1-72815-023-9 (cit. on p. 17).

Newson, A. et al. (2014). "Video Inpainting of Complex Scenes". In: *SIAM Journal on Imaging Sciences* 7.4, pp. 1993–2019. DOI: 10/gf395r (cit. on pp. 30, 66, 87, 91, 96–99, 102).

Newson, A. et al. (2017). "Non-Local Patch-Based Image Inpainting". In: *IPOL* (cit. on pp. 67, 70, 75, 76, 80, 110).

Nichol, A. Q. and P. Dhariwal (2021). "Improved Denoising Diffusion Probabilistic Models". In: *Proceedings of the 38th International Conference on Machine Learning*. PMLR, pp. 8162–8171 (cit. on pp. 15, 86, 90, 102).

Nikankin, Y., N. Haim, and M. Irani (2023). "SinFusion: training diffusion models on a single image or video". In: *Proceedings of the 40th International Conference on Machine Learning*. ICML'23. Honolulu, Hawaii, USA: JMLR.org (cit. on pp. 25, 70, 87).

Oh, S. W. et al. (2019). "Onion-Peel Networks for Deep Video Completion". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. DOI: 10/gg2ntk (cit. on p. 88).

Ouyang, H., T. Wang, and Q. Chen (2021). "Internal Video Inpainting by Implicit Long-range Propagation". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Montreal, QC, Canada: IEEE, pp. 14559–14568. ISBN: 978-1-66542-812-5. DOI: 10.1109/ICCV48922.2021.01431 (cit. on pp. 25, 71, 88, 91).

Pajouheshgar, E. et al. (2023). "DyNCA: Real-Time Dynamic Texture Synthesis Using Neural Cellular Automata". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vancouver, BC, Canada: IEEE, pp. 20742–20751. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.01987 (cit. on pp. 89, 104, 105).

Pan, X. et al. (2023). "Slide-Transformer: Hierarchical Vision Transformer with Local Self-Attention". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vancouver, BC, Canada: IEEE, pp. 2082–2091. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.00207 (cit. on p. 41).

Parmar, N. et al. (2018). "Image Transformer". In: *ICML* (cit. on pp. 20, 39, 41, 48, 51–54).

Pathak, D. et al. (2016). "Context Encoders: Feature Learning by Inpainting". In: *(CVPR) Conference on Computer Vision and Pattern Recognition*. Vol. 2016-Decem. IEEE Computer Society, pp. 2536–2544. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.278. arXiv: 1604.07379 (cit. on pp. 16, 21, 70, 76, 78).

Patwardhan, K., G. Sapiro, and M. Bertalmio (2005). "Video Inpainting of Occluding and Occluded Objects". In: *IEEE International Conference on Image Processing 2005*. Vol. 2, pp. II–69. DOI: 10.1109/ICIP.2005.1529993 (cit. on p. 87).

Patwardhan, K. A., G. Sapiro, and M. Bertalmio (2007). "Video Inpainting Under Constrained Camera Motion". In: *IEEE Transactions on Image Processing* 16.2, pp. 545–553. DOI: 10.1109/TIP.2006.888343 (cit. on p. 87).

Peebles, W. and S. Xie (2023). "Scalable Diffusion Models with Transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205 (cit. on p. 21).

Peng, H. et al. (2021a). "Random Feature Attention". In: *International Conference on Learning Representations* (cit. on p. 42).

Peng, J. et al. (2021b). "Generating Diverse Structure for Image Inpainting With Hierarchical VQ-VAE". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10775–10784 (cit. on p. 18).

Perazzi, F. et al. (2016). "A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 724–732. DOI: 10/ggdmmw (cit. on pp. 54, 94, 95).

Pérez, P., M. Gangnet, and A. Blake (2003). "Poisson Image Editing". In: *ACM SIGGRAPH 2003 Papers*. SIGGRAPH '03. New York, NY, USA: Association for Computing Machinery, pp. 313–318. ISBN: 978-1-58113-709-5. DOI: 10.1145/1201775.882269 (cit. on p. 12).

Plötz, T. and S. Roth (2018). "Neural nearest neighbors networks". In: *(NeurIPS) Advances in Neural Information Processing Systems*, pp. 1087–1098. arXiv: 1810.12575 (cit. on pp. 41, 46).

Pont-Tuset, J. et al. (2017). "The 2017 DAVIS Challenge on Video Object Segmentation". In: *arXiv:1704.00675* (cit. on p. 95).

Portilla, J. and E. P. Simoncelli (2000). "A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients". In: *International Journal of Computer Vision* 40.1, pp. 49–70. ISSN: 1573-1405. DOI: 10.1023/A:1026553619983 (cit. on pp. 9, 23).

Pritch, Y., E. Kav-Venaki, and S. Peleg (2009). "Shift-Map Image Editing". In: *2009 IEEE 12th International Conference on Computer Vision*, pp. 151–158. DOI: 10/bphb6x (cit. on p. 11).

Raad, L., A. Desolneux, and J.-M. Morel (2014). "Locally Gaussian Exemplar Based Texture Synthesis". In: *2014 IEEE International Conference on Image Processing (ICIP)*. Paris, France: IEEE, pp. 4667–4671. ISBN: 978-1-4799-5751-4. DOI: 10.1109/ICIP.2014.7025946 (cit. on p. 10).

Rae, J. W. et al. (2020). "Compressive Transformers for Long-Range Sequence Modelling". In: *International Conference on Learning Representations* (cit. on p. 42).

Ren, J. et al. (2022). "DLFormer: Discrete Latent Transformer for Video Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3511–3520 (cit. on pp. 71, 88).

Ren, Y. et al. (2019). "StructureFlow: Image Inpainting via Structure-Aware Appearance Flow". en. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 181–190. ISBN: 978-1-72814-803-8 (cit. on pp. 17, 22).

Rezende, D. and S. Mohamed (2015). "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, pp. 1530–1538 (cit. on p. 14).

Rombach, R. et al. (2022). "High-Resolution Image Synthesis With Latent Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695 (cit. on pp. 15, 21, 63, 69, 85, 86, 90).

Rong, G. and T.-S. Tan (2006). "Jump flooding in GPU with applications to Voronoi diagram and distance transform". en. In: *Proceedings of the 2006 symposium on Interactive 3D graphics and games - SI3D '06*. Redwood City, California: ACM Press, p. 109. ISBN: 978-1-59593-295-2 (cit. on p. 45).

Roy, A. et al. (2021). "Efficient Content-Based Sparse Attention with Routing Transformers". In: *Transactions of the Association for Computational Linguistics* 9, pp. 53–68. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00353 (cit. on p. 41).

Ruiz, N. et al. (2023). "DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510 (cit. on pp. 70, 83, 87).

Saharia, C. et al. (2022a). "Image Super-Resolution Via Iterative Refinement". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–14. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2022.3204461 (cit. on pp. 70, 76, 78, 99).

Saharia, C. et al. (2022b). "Palette: Image-to-Image Diffusion Models". In: *SIGGRAPH* (cit. on pp. v, 18, 69–71, 78, 80, 85, 86, 89, 111).

Saharia, C. et al. (2022c). *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. arXiv: 2205.11487 [cs.CV] (cit. on p. 21).

Salimans, T. and J. Ho (2022). "Progressive Distillation for Fast Sampling of Diffusion Models". In: *International Conference on Learning Representations* (cit. on p. 137).

Samuth, B., D. Tschumperlé, and J. Rabin (2022). "A Patch-Based Approach for Artistic Style Transfer Via Constrained Multi-Scale Image Matching". In: *2022 IEEE International Conference on Image Processing (ICIP)*, pp. 3490–3494. DOI: 10.1109/ICIP46576.2022.9897334 (cit. on pp. 64, 67).

Samuth, B. et al. (2023). "LatentPatch: A Non-Parametric Approach for Face Generation and Editing". In: *2023 IEEE International Conference on Image Processing (ICIP)*, pp. 1790–1794. DOI: 10.1109/ICIP49359.2023.10222005 (cit. on p. 12).

Sánchez, J. (2017). "Comparison of Motion Smoothing Strategies for Video Stabilization Using Parametric Models". In: *Image Processing On Line* 7, pp. 309–346. ISSN: 2105-1232. DOI: 10/gg4wbs (cit. on p. 103).

Sasaki, K. et al. (2018). "Learning to Restore Deteriorated Line Drawing". In: *IJCG* (cit. on p. 82).

Shaham, T. R., T. Dekel, and T. Michaeli (2019). "SinGAN: Learning a Generative Model From a Single Natural Image". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 4569–4579. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00467 (cit. on pp. v, 5, 11, 23–25, 28, 30, 31, 70, 91, 93, 103, 109).

Shamsolmoali, P., M. Zareapoor, and E. Granger (2023). "TransInpaint: Transformer-Based Image Inpainting with Context Adaptation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 849–858 (cit. on p. 22).

Shiratori, T. et al. (2006). "Video Completion by Motion Field Transfer". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 1, pp. 411–418. DOI: 10.1109/CVPR.2006.330 (cit. on p. 87).

Shocher, A. et al. (2019). "InGAN: Capturing and Retargeting the "DNA" of a Natural Image". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4492–4501 (cit. on p. 25).

Simakov, D. et al. (2008). "Summarizing Visual Data Using Bidirectional Similarity". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. DOI: 10/dwwxrz (cit. on p. 11).

Simard, P., D. Steinkraus, and J. Platt (2003). "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis". In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* Vol. 1. Edinburgh, UK: IEEE Comput. Soc, pp. 958–963. ISBN: 978-0-7695-1960-9. DOI: 10.1109/ICDAR.2003.1227801 (cit. on p. 37).

Simonyan, K. and A. Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations* (cit. on p. 16).

Singer, U. et al. (2022). "Make-A-Video: Text-to-Video Generation without Text-Video Data". In: arXiv:2209.14792. DOI: 10.48550/arXiv.2209.14792. arXiv: 2209.14792 [cs] (cit. on pp. 85, 86).

Sohl-Dickstein, J. et al. (2015). "Deep Unsupervised Learning Using Nonequilibrium Thermodynamics". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, pp. 2256–2265 (cit. on pp. vi, 14, 69, 71, 86).

Song, J., C. Meng, and S. Ermon (2021a). "Denoising Diffusion Implicit Models". In: *International Conference on Learning Representations* (cit. on pp. 15, 85).

Song, J. et al. (2023). "Pseudoinverse-Guided Diffusion Models for Inverse Problems". In: *(ICLR) International Conference on Learning Representations* (cit. on p. 18).

Song, Y. and S. Ermon (2019). "Generative Modeling by Estimating Gradients of the Data Distribution". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc. (cit. on p. 15).

Song, Y. et al. (2021b). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations* (cit. on p. 15).

Song, Y. et al. (2018). "Contextual-Based Image Inpainting: Infer, Match, and Translate". In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari et al. Cham: Springer International Publishing, pp. 3–18. ISBN: 978-3-030-01216-8 (cit. on p. 21).

Strobel, M., J. Diebold, and D. Cremers (2014). "Flow and Color Inpainting for Video Completion". In: *Pattern Recognition*. Ed. by X. Jiang, J. Hornegger, and R. Koch. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 293–304. ISBN: 978-3-319-11752-2. DOI: 10.1007/978-3-319-11752-2_23 (cit. on p. 87).

Sun, J. et al. (2005). "Image completion with structure propagation". In: *ACM Trans. Graph.* 24.3, pp. 861–868. ISSN: 0730-0301. DOI: 10.1145/1073204.1073274 (cit. on p. 11).

Sushko, V., J. Gall, and A. Khoreva (2021). "One-Shot GAN: Learning To Generate Samples From Single Images and Videos". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 2596–2600 (cit. on p. 25).

Suvorov, R. et al. (2022). "Resolution-Robust Large Mask Inpainting with Fourier Convolutions". In: *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, pp. 3172–3182. ISBN: 978-1-66540-915-5. DOI: 10.1109/WACV51458.2022.00323 (cit. on pp. 18, 70, 78).

Tancik, M. et al. (2020). "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains". In: *NeurIPS* (cit. on pp. 19, 73).

Tang, L. et al. (2023). *RealFill: Reference-Driven Generation for Authentic Image Completion*. DOI: 10.48550/arXiv.2309.16668. arXiv: 2309.16668 [cs] (cit. on pp. 18, 70).

Tay, Y. et al. (2020a). "Efficient Transformers: A Survey". In: *ArXiv* (cit. on p. 42).

Tay, Y. et al. (2020b). "Sparse sinkhorn attention". In: ICML'20. JMLR.org (cit. on p. 41).

Tesfaldet, M., M. A. Brubaker, and K. G. Derpanis (2018). "Two-Stream Convolutional Networks for Dynamic Texture Synthesis". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6703–6712 (cit. on pp. 89, 97, 99, 103–105).

Tong, Z. et al. (2022). "VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training". In: *Advances in Neural Information Processing Systems* (cit. on p. 21).

Trippe, B. L. et al. (2023). "Diffusion Probabilistic Modeling of Protein Backbones in 3D for the Motif-Scaffolding Problem". In: *The Eleventh International Conference on Learning Representations* (cit. on p. 18).

Turk, G. (2001). "Texture Synthesis on Surfaces". In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, pp. 347–354. ISBN: 978-1-58113-374-5. DOI: 10.1145/383259.383297 (cit. on p. 10).

Ulyanov, D., A. Vedaldi, and V. Lempitsky (2018). "Deep Image Prior". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454 (cit. on pp. 25, 70, 88, 111).

Ulyanov, D. et al. (2016). "Texture networks: Feed-forward synthesis of textures and stylized images." In: *ICML*. Vol. 1, p. 4 (cit. on p. 16).

Unterthiner, T. et al. (2019). *FVD: A new Metric for Video Generation* (cit. on p. 93).

van den Oord, A. et al. (2016). "Conditional Image Generation with PixelCNN Decoders". In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc. (cit. on p. 14).

Van Den Oord, A., N. Kalchbrenner, and K. Kavukcuoglu (2016). "Pixel Recurrent Neural Networks". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, pp. 1747–1756 (cit. on p. 14).

Vaswani, A. et al. (2017). "Attention is All you Need". In: *(NeurIPS) Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., pp. 5998–6008 (cit. on pp. vi, 5, 18–20, 36, 39, 48, 52, 109).

Vaswani, A. et al. (2021). "Scaling Local Self-Attention for Parameter Efficient Visual Backbones". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, pp. 12889–12899. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.01270 (cit. on p. 41).

Vyas, A., A. Katharopoulos, and F. Fleuret (2020). "Fast Transformers with Clustered Attention". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., pp. 21665–21674 (cit. on pp. 39, 41).

Wan, Z. et al. (2021). "High-Fidelity Pluralistic Image Completion with Transformers". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Montreal, QC, Canada: IEEE, pp. 4672–4681. ISBN: 978-1-66542-812-5. DOI: 10.1109/ICCV48922.2021.00465 (cit. on p. 22).

Wang, C. et al. (2019). "Video Inpainting by Jointly Learning Temporal Structure and Spatial Details". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01, pp. 5232–5239. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v33i01.33015232 (cit. on p. 88).

Wang, F. et al. (2021a). "Patchmatchnet: Learned multi-view patchmatch stereo". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14194–14203 (cit. on p. 64).

Wang, S. et al. (2021b). "Cluster-Former: Clustering-based Sparse Transformer for Long-Range Dependency Encoding". In: *ACL-IJCNLP 2021* (cit. on p. 41).

Wang, S. et al. (2020). "Linformer: Self-attention with linear complexity". In: *arXiv preprint arXiv:2006.04768* (cit. on pp. 39, 42).

Wang, T., H. Ouyang, and Q. Chen (2021c). "Image Inpainting with External-internal Learning and Monochromic Bottleneck". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, pp. 5116–5125. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.00508 (cit. on p. 70).

Wang, T.-C. et al. (2018a). "Video-to-Video Synthesis". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., pp. 1144–1156 (cit. on pp. 93, 95).

Wang, W. et al. (2022). *SinDiffusion: Learning a Diffusion Model from a Single Natural Image*. DOI: 10.48550/arXiv.2211.12445. arXiv: 2211.12445 [cs] (cit. on pp. 25, 70, 87).

Wang, W. et al. (2021d). "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 548–558. DOI: 10.1109/ICCV48922.2021.00061 (cit. on p. 41).

Wang, X. et al. (2018b). "Non-Local Neural Networks". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. DOI: 10/gftc9c (cit. on p. 39).

Wang, Y. et al. (2018c). "Image Inpainting via Generative Multi-column Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. (cit. on p. 17).

Wang, Z. et al. (2004). "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4, pp. 600–612. DOI: 10.1109/TIP.2003.819861 (cit. on p. 77).

Wei, L.-Y. and M. Levoy (2000). "Fast Texture Synthesis Using Tree-Structured Vector Quantization". In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '00*. Not Known: ACM Press, pp. 479–488. ISBN: 978-1-58113-208-3. DOI: 10.1145/344779.345009 (cit. on pp. 10, 89).

Wei, L.-Y. et al. (2009). "State of the Art in Example-based Texture Synthesis". In: *Eurographics 2009, State of the Art Report, EG-STAR* (cit. on p. 11).

Wexler, Y., E. Shechtman, and M. Irani (2007). "Space-Time Completion of Video". en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.3, pp. 463–476. ISSN: 0162-8828, 2160-9292. DOI: 10/ccnqgr (cit. on pp. 11, 21, 24, 40, 56, 70, 87).

Woo, S. et al. (2020). "Align-and-Attend Network for Globally and Locally Coherent Video Inpainting". In: *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*. BMVA Press (cit. on p. 88).

Wu, R. and C. Zheng (2022). "Learning to Generate 3D Shapes from a Single Example". In: *ACM Transactions on Graphics* 41.6, 224:1–224:19. ISSN: 0730-0301. DOI: 10.1145/3550454.3555480 (cit. on p. 25).

Xie, C. et al. (2019). "Image Inpainting With Learnable Bidirectional Attention Maps". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 8857–8866. ISBN: 978-1-72814-803-8. DOI: 10/ghfg93 (cit. on pp. 17, 22).

Xie, J., S.-C. Zhu, and Y. N. Wu (2017). "Synthesizing Dynamic Patterns by Spatial-Temporal Generative ConvNet". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, pp. 1061–1069. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.119 (cit. on pp. 87, 89, 104, 105).

Xie, J., L. Xu, and E. Chen (2012). "Image Denoising and Inpainting with Deep Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc. (cit. on pp. 16, 70).

Xie, S. et al. (2023). "SmartBrush: Text and Shape Guided Object Inpainting With Diffusion Model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22428–22437 (cit. on pp. 15, 18, 70, 86).

Xiong, W. et al. (2019). "Foreground-Aware Image Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5840–5848 (cit. on p. 17).

Xiong, Y. et al. (2021). "Nyströmformer: A Nyström-based Algorithm for Approximating Self-Attention". In: (cit. on p. 42).

Xu, N. et al. (2018a). "YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark". In: arXiv:1809.03327. DOI: 10.48550/arXiv.1809.03327. arXiv: 1809.03327 [cs] (cit. on p. 95).

Xu, R. et al. (2019). "Deep Flow-Guided Video Inpainting". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3718–3727. DOI: 10/ggr9sh (cit. on pp. 88, 97).

Xu, R. et al. (2021). "Positional Encoding as Spatial Inductive Bias in GANs". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, pp. 13564–13573. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.01336 (cit. on pp. 25, 103).

Xu, T. et al. (2018b). "AttnGAN: Fine-Grained Text to Image Generation With Attentional Generative Adversarial Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1316–1324 (cit. on pp. 19, 21).

Yan, Z. et al. (2018). "Shift-Net: Image Inpainting via Deep Feature Rearrangement". en. In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari et al. Vol. 11218. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 3–19. ISBN: 978-3-030-01263-2 978-3-030-01264-9 (cit. on pp. 17, 22).

Yang, B. et al. (2023a). "Paint by Example: Exemplar-based Image Editing with Diffusion Models". In: *2023 IEEE/CVF Conference on Computer Vision and*

Pattern Recognition (CVPR). Vancouver, BC, Canada: IEEE, pp. 18381–18391. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.01763 (cit. on p. 70).

Yang, C. et al. (2017). "High-Resolution Image Inpainting Using Multi-Scale Neural Patch Synthesis". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6721–6729 (cit. on pp. 12, 17, 21).

Yang, F. et al. (2016). "Stationary dynamic texture synthesis using convolutional neural networks". In: *2016 IEEE 13th International Conference on Signal Processing (ICSP)*, pp. 1135–1139. DOI: 10.1109/ICSP.2016.7878005 (cit. on p. 89).

Yang, L. et al. (2023b). "Diffusion Models: A Comprehensive Survey of Methods and Applications". In: *ACM Computing Surveys* 56.4, 105:1–105:39. ISSN: 0360-0300. DOI: 10.1145/3626235 (cit. on p. 15).

Yao, X. et al. (2022). "A Style-Based GAN Encoder for High Fidelity Reconstruction of Images and Videos". In: *Computer Vision – ECCV 2022*. Ed. by S. Avidan et al. Vol. 13675. Cham: Springer Nature Switzerland, pp. 581–597. ISBN: 978-3-031-19783-3 978-3-031-19784-0. DOI: 10.1007/978-3-031-19784-0_34 (cit. on p. 14).

Yeh, R. A. et al. (2017). "Semantic Image Inpainting with Deep Generative Models". In: *arXiv:1607.07539 [cs]*. arXiv: 1607.07539 [cs] (cit. on p. 17).

Yi, Z. et al. (2020). "Contextual Residual Aggregation for Ultra High-Resolution Image Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7508–7517 (cit. on pp. 18, 22).

You, X. et al. (2016). "Kernel Learning for Dynamic Texture Synthesis". In: *IEEE Transactions on Image Processing* 25.10, pp. 4782–4795. DOI: 10.1109/TIP.2016.2598653 (cit. on p. 89).

Yu, J. et al. (2018). "Generative Image Inpainting with Contextual Attention". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. DOI: 10/ggv983 (cit. on pp. 17, 21, 39, 40, 43, 48, 56, 57, 61, 62, 70, 74, 75, 81, 85, 94).

Yu, J. et al. (2019). "Free-Form Image Inpainting With Gated Convolution". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4470–4479. DOI: 10/gg2nth (cit. on pp. 17, 70, 73, 88).

Yu, S. et al. (2023). "Video Probabilistic Diffusion Models in Projected Latent Space". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18456–18466 (cit. on p. 86).

Zaheer, M. et al. (2020). "Big bird: Transformers for longer sequences". In: *Advances in Neural Information Processing Systems* 33 (cit. on p. 42).

Zamir, S. W. et al. (2022). "Restormer: Efficient Transformer for High-Resolution Image Restoration". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, pp. 5718–5729. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.00564 (cit. on p. 66).

Zeng, Y., J. Fu, and H. Chao (2020). "Learning Joint Spatial-Temporal Transformations for Video Inpainting". In: *Computer Vision – ECCV 2020*. Ed. by A.

Vedaldi et al. Cham: Springer International Publishing, pp. 528–543. ISBN: 978-3-030-58517-4 (cit. on pp. 88, 93, 94, 97).

Zeng, Y. et al. (2021). "CR-Fill: Generative Image Inpainting with Auxiliary Contextual Reconstruction". In: *Proceedings of the IEEE International Conference on Computer Vision* (cit. on p. 22).

Zhai, X. et al. (2022). "Scaling Vision Transformers". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, pp. 1204–1213. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01179 (cit. on p. 21).

Zhang, B. et al. (2022a). "StyleSwin: Transformer-based GAN for High-resolution Image Generation". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, pp. 11294–11304. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01102 (cit. on p. 20).

Zhang, H. et al. (2019a). "Self-Attention Generative Adversarial Networks". In: *ICML* (cit. on pp. 20, 39).

Zhang, H. et al. (2019b). "An Internal Learning Approach to Video Inpainting". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, pp. 2720–2729. ISBN: 978-1-72814-803-8. DOI: 10/gg2ntg (cit. on pp. 25, 71, 88, 91).

Zhang, K., J. Fu, and D. Liu (2022b). "Flow-Guided Transformer for Video Inpainting". In: *European Conference on Computer Vision*. Springer, pp. 74–90 (cit. on pp. 88, 97).

— (2022c). "Inertia-Guided Flow Completion and Style Fusion for Video Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5982–5991 (cit. on pp. 88, 96–98, 106).

Zhang, L. et al. (2022d). "Inpainting at Modern Camera Resolution by Guided PatchMatch with Auto-curation". In: *Computer Vision – ECCV 2022*. Ed. by S. Avidan et al. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, pp. 51–67. ISBN: 978-3-031-19790-1. DOI: 10.1007/978-3-031-19790-1_4 (cit. on pp. 12, 18).

Zhang, L., A. Rao, and M. Agrawala (2023). "Adding Conditional Control to Text-to-Image Diffusion Models". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847 (cit. on p. 15).

Zhang, R. et al. (2018a). "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, pp. 586–595. ISBN: 978-1-5386-6420-9. DOI: 10/gfz33w (cit. on pp. 36, 77, 78, 92).

Zhang, Y. et al. (2018b). "Image Super-Resolution Using Very Deep Residual Channel Attention Networks". In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari et al. Vol. 11211. Cham: Springer International Publishing, pp. 294–310. ISBN: 978-3-030-01233-5 978-3-030-01234-2. DOI: 10.1007/978-3-030-01234-2_18 (cit. on p. 40).

Zhang, Z., C. Han, and T. Guo (2021). "ExSinGAN: Learning an Explainable Generative Model from a Single Image". In: *32nd British Machine Vision Conference*

2021, BMVC 2021, Online, November 22-25, 2021. BMVA Press, p. 251 (cit. on p. 25).

Zhao, L. et al. (2021). "Improved Transformer for High-Resolution GANs". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. (cit. on p. 20).

Zhao, S. et al. (2023). "Large Scale Image Completion via Co-Modulated Generative Adversarial Networks". In: *International Conference on Learning Representations* (cit. on p. 18).

Zheng, C., T.-J. Cham, and J. Cai (2019). "Pluralistic Image Completion". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, pp. 1438–1447. ISBN: 978-1-72813-293-8. DOI: 10/gg9hw8 (cit. on pp. 18, 22, 70).

Zheng, C. et al. (2022a). "Bridging Global Context Interactions for High-Fidelity Image Completion". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11502–11512. DOI: 10.1109/CVPR52688.2022.01122 (cit. on p. 22).

Zheng, Z. et al. (2022b). "DIP: Deep Inverse Patchmatch for High-Resolution Optical Flow". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8925–8934 (cit. on p. 64).

Zhou, B. et al. (2017). "Places: A 10 million image database for scene recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 40.6, pp. 1452–1464 (cit. on p. 31).

Zhou, G., N. Dong, and Y. Wang (2009). "Non-Linear Dynamic Texture Analysis and Synthesis Using Constrained Gaussian Process Latent Variable Model". In: *2009 Pacific-Asia Conference on Circuits, Communications and Systems*, pp. 27–30. DOI: 10.1109/PACCS.2009.30 (cit. on p. 89).

Zhou, S. et al. (2023). "ProPainter: Improving Propagation and Transformer for Video Inpainting". In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Paris, France: IEEE, pp. 10443–10452. ISBN: 9798350307184. DOI: 10.1109/ICCV51070.2023.00961 (cit. on pp. 88, 95, 97).

Zhou, Y. et al. (2021). "TransFill: Reference-Guided Image Inpainting by Merging Multiple Color and Spatial Transformations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2266–2276 (cit. on p. 17).

Zou, X. et al. (2021). "Progressive Temporal Feature Alignment Network for Video Inpainting". In: *CVPR* (cit. on p. 97).

A Appendix

A.1 Parametrizations in diffusion models

We briefly review the few lines of computation needed to establish the loss function formulas for different parametrizations in diffusion models. We recall the parameters of the diffusion process for a given transition kernel:

$$q(x_{t+1}|x_t) \sim \mathcal{N}(\sqrt{1-\beta_t}x_t|\beta_t\mathbf{I})$$

We have $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Ho et al. (2020) derive the loss function for a given timestep:

$$L_{t-1} = D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))$$

For Gaussian variables, then we have $q(x_{t-1}|x_t, x_0) \sim \mathcal{N}(\tilde{\mu}_t(x_0, x_t), \tilde{\beta}_t I)$ and $p_\theta(x_{t-1}|x_t) \sim \mathcal{N}(\mu_\theta(x_t, t), \Sigma_t)$.

For q, the parameters $\left(\tilde{\mu}_t(x_0, x_t), \tilde{\beta}_t\right)$ are known. For p, Ho *et al.* chose to use fixed variances for the reverse process *i.e.* $\Sigma_t = \sigma_t^2\mathbf{I}$. Then the L_{t-1} loss depends only on the differences between the means of the Gaussian variables since we do not learn the other variables:

$$\begin{aligned}
L_{t-1} &= D_{KL}(\mathcal{N}(\tilde{\mu}_t(x_0, x_t), \tilde{\beta}_t\mathbf{I})||\mathcal{N}(\mu_\theta(x_t, t), \sigma_t^2\mathbf{I}) \\
&= \frac{1}{2}\left[\log\frac{|\sigma_t^2 I|}{|\beta_t I|} - d + \mathrm{tr}\{(\sigma_t^2 I)^{-1}(\tilde{\beta}_t I)\} + (\tilde{\mu}_t - \mu_\theta)^T(\sigma_t^2 I)^{-1}(\tilde{\mu}_t - \mu_\theta)\right] \\
&= \frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(x_0, x_t) - \mu_\theta\|^2 + C
\end{aligned}$$

Now, we have to consider that we will use a neural network f_θ to predict this mean. Here we have several options, we present 3 natural parametrizations but there are alternatives such as the v-parametrization (Salimans and Ho 2022).

A.1.1 Parametrization #1

The first and obvious parametrization is to train the network to predict the mean, *i.e.* $f_\theta(x_t, t) = \mu_\theta(x_t, t)$. Then the loss is directly:

$$L_{t-1} = \frac{1}{2\sigma_t^2} \left\| \tilde{\mu}_t(x_t, x_0) - f_\theta(x_t, t) \right\|^2$$

In practice, we sample a clean image x_0 from the dataset, and a noisy version x_t, not directly $\tilde{\mu}_t(x_0, x_t) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$. This parametrization is not very popular.

A.1.2 Parametrization #2

We write $\tilde{\mu}_t(x_0, x_t)$ as a function of x_0 and x_t :

$$\tilde{\mu}_t(x_0, x_t) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

In this expression, x_0 is the important term to predict from the the point of view of the neural network because x_t is an input. We can thus write:

$$\mu_\theta(x_t, t) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} \hat{x}_{0,\theta}(x_t, t) + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

With the neural network f_θ predicting the clean original image: $f_\theta(x_t, t) = \hat{x}_{0,\theta}(x_t, t)$. Then the loss function is:

$$\begin{aligned}
L_{t-1} &= \frac{1}{2\sigma_t^2} \left\| \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \cancel{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t} - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} \hat{x}_{0,\theta}(x_t, t) - \cancel{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t} \right\|^2 \\
&= \frac{1}{2\sigma_t^2} \left\| \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} \left(x_0 - \hat{x}_{0,\theta}(x_t, t) \right) \right\|^2 \\
&= \frac{1}{2\sigma_t^2} \frac{\bar{\alpha}_{t-1}\beta_t^2}{(1-\bar{\alpha}_t)^2} \left\| x_0 - \hat{x}_{0,\theta}(x_t, t) \right\|^2
\end{aligned}$$

The link with denoising is obvious in this formulation (with the minor nitpick that x_t is a noisy *and* scaled-down version of x_0). This is a very intuitive formulation.

A.1.3 Parametrization #3

Finally, we can rewrite x_0 as a function of ε and x_t:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon \leftrightarrow x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon}{\sqrt{\bar{\alpha}_t}}$$

Then we can write $\tilde{\mu}_t(x_0, x_t)$ as a sum of ε and x_t:

$$\begin{aligned}
\tilde{\mu}_t(x_0, x_t) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\
&= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\left(\frac{1}{\sqrt{\bar{\alpha}}}x_t - \frac{\sqrt{1-\bar{\alpha}}}{\sqrt{\bar{\alpha}}}\varepsilon\right) + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\
&= \left(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{\sqrt{\bar{\alpha}}(1-\bar{\alpha}_t)} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\right)x_t - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\varepsilon
\end{aligned}$$

Same as previously, the unknown and important variable is ε because x_t is known during inference. So we set $f_\theta(x_t, t) = \hat{\varepsilon}(x_t, t)$.

Plugging this into our main equation, we have:

$$\begin{aligned}
L_{t-1} &= \frac{1}{2\sigma_t^2}\|\cancel{\left(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{\sqrt{\bar{\alpha}}(1-\bar{\alpha}_t)} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\right)x_t} - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\varepsilon \\
&\quad - \cancel{\left(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{\sqrt{\bar{\alpha}}(1-\bar{\alpha}_t)} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\right)x_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\hat{\varepsilon}_\theta(x_t, t)\|^2 \\
&= \frac{1}{2\sigma_t^2}\left\|\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\left(\varepsilon - \hat{\varepsilon}_\theta(x_t, t)\right)\right\|^2 \\
&= \frac{1}{2\sigma_t^2}\frac{\beta_t^2}{(1-\bar{\alpha}_t)}\frac{\bar{\alpha}_{t-1}}{\bar{\alpha}_t}\|\varepsilon - \hat{\varepsilon}_\theta(x_t, t)\|^2 \\
&= \frac{1}{2\sigma_t^2}\frac{\beta_t^2}{\alpha_t(1-\bar{\alpha}_t)^2}\|\varepsilon - \hat{\varepsilon}_\theta(x_t, t)\|^2
\end{aligned}$$

This formulation can be seen as a prediction of the residual noise, similar to some denoising approaches. Ho *et al.* prefer this formulation, which is considered more stable. In their original paper, the authors also propose a simplified loss:

$$L_{\text{simple},t-1} = ||\varepsilon - \hat{\varepsilon}_\theta(x_t, t)||^2$$

ECOLE
DOCTORALE

Titre : Méthodes internes pour la génération et l'inpainting d'images et de vidéos

Mots clés : interne, inpainting, vidéo, apprentissage profond

Résumé : L'édition et la génération d'images sont des problèmes complexes dans le domaine du traitement d'images. Récemment, nous avons vu un grand bond en avant dans le développement en utilisant des approches basées sur l'apprentissage qui tirent parti de grandes bases de données d'images. Dans cette thèse, nous étudions les méthodes internes, c'est-à-dire les méthodes basées sur une seule image comme source de données. Cela inclut les méthodes par patchs, les approches d'apprentissage interne qui entraînent un réseau neuronal sur une seule image, et les mécanismes d'attention qui combinent les patchs et les réseaux profonds.
Tout d'abord, nous présentons une contribution à la génération mono-image avec une méthode par patchs. Cette approche classique est compétitive par rapport aux approches récentes par réseau mais évite la phase d'apprentissage.
Deuxièmement, les mécanismes d'attention sont importants pour modéliser les dépendances à longue distance et sont plus flexibles que les convolutions, mais souffrent d'une terrible complexité calculatoire. En fait, la complexité croît de façon quadratique avec le nombre d'éléments d'entrée, ce qui rend ces couches inutilisables pour les images haute-résolution ou les vidéos. Nous proposons une approximation efficace basée sur la recherche du plus proche voisin.
Enfin, nous examinons les modèles de diffusion récents pour l'inpainting d'images et de vidéos. Dans les cas mono-images, nous montrons comment des architectures très légères sont compétitives par rapport à l'état de l'art. Nos modèles s'exécutent et s'entraînent pour une fraction du coût de calcul des modèles les plus courants. Nous proposons également une application à l'inpainting vidéo avec une stratégie d'entraînement spécifique qui améliore significativement les résultats par rapport à la méthode de base. Cette stratégie est particulièrement adaptée à ces modèles à usage unique.

Title : Internal methods for the generation and inpainting of images and videos

Keywords : internal, inpainting, video, deep learning

Abstract : Image editing and generation are complex problems in image processing. Recently, we have seen a great leap in development by using learning-based approaches that take advantage of large image databases. In this thesis, we study internal methods i.e. methods based on a single image as a data source. This includes patch-based methods, internal learning approaches that train a neural network on a single image, and attention mechanisms that combine patches and deep networks.
First, we present a contribution to single image generation with a patch-based method. This classical approach is competitive with recent network-based approaches but does not require a learning phase.
Second, attention mechanisms are important for modeling long-range dependencies and are more flexible than convolutions but suffer from poor computational complexity. In fact, the complexity grows quadratically with the number of input elements making such layers unusable for high-resolution images or videos. We propose an efficient approximation based on nearest neighbor search.
Finally, we look at the recent diffusion models for image and video inpainting. In the single image setting, we show how very lightweight architectures are competitive with the state-of-the-art. Our models run and train at a fraction of the computational cost of popular models. We also propose an application to video inpainting with a specific training strategy that significantly improves the results over the baseline. This strategy is particularly adapted to these one-shot models.

Institut Polytechnique de Paris
91120 Palaiseau, France