Wasserstein Generative Models for Patch-Based Texture Synthesis

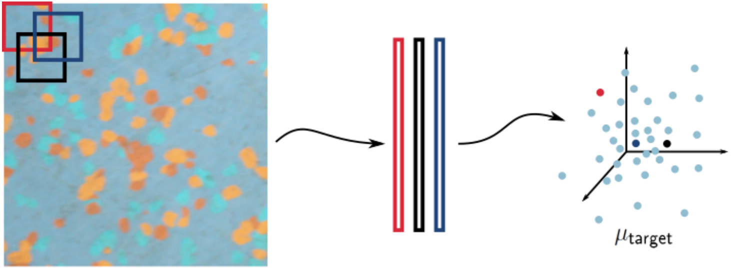
Antoine Houdard[1], Arthur Leclaire[1], Nicolas Papadakis[1], Julien Rabin[2]

SSVM online conference, May 18, 2021

1: Institut de Mathématiques de Bordeaux

2: GREYC

Patch-based Texture Synthesis

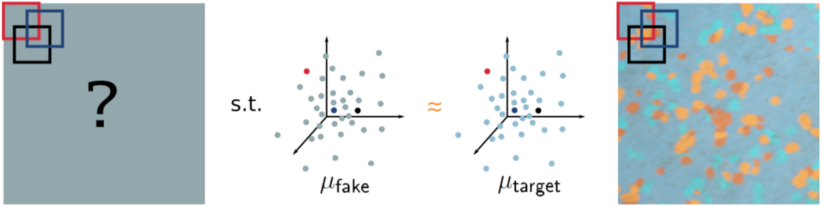


Represent a target texture u with its patch distribution

$$\mu_{\text{target}} = \frac{1}{m} \sum_{i=1}^{m} \delta_{P_i u}$$

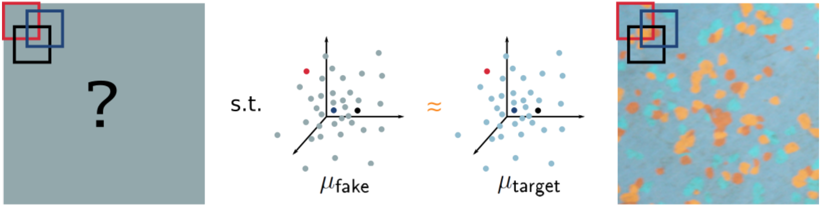
with P_i linear operator that extract the i-th patch

Patch-based Texture Synthesis



Main idea Find or generate an image s.t. μ_{fake} is *close to* μ_{target}

Patch-based Texture Synthesis

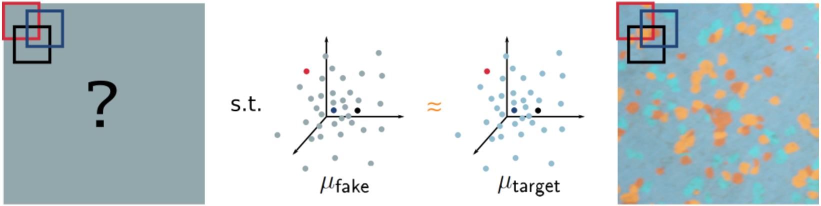


Main idea Find or generate an image s.t. μ_{fake} is *close to* μ_{target}

Two scenarios:

- Single image: $\mu_{\text{fake}} = \frac{1}{n} \sum \delta_{P_i \theta}$ discrete (part 2)
- Generative model: $\mu_{\text{fake}} = \frac{1}{n} \sum (P_i \circ g_\theta) \sharp \zeta$ continuous (part 3)

Patch-based Texture Synthesis



Main idea Find or generate an image s.t. μ_{fake} is *close to* μ_{target}

Two scenarios:

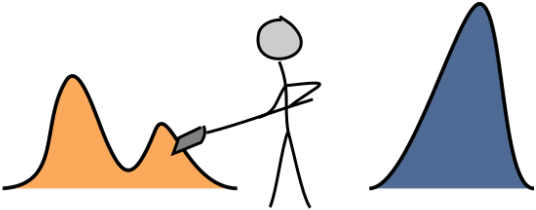
- Single image: $\mu_{\text{fake}} = \frac{1}{n} \sum \delta_{P_i \theta}$ discrete (part 2)
- Generative model: $\mu_{\text{fake}} = \frac{1}{n} \sum (P_i \circ g_\theta) \sharp \zeta$ continuous (part 3)

close to $\rightarrow$ Optimal Transport cost! (Part 1)

Part 1 – Theoretical results

Optimal Transport

Definition $\mathrm{OT}_c(\mu,\nu) = \inf_{\pi\in\Pi(\mu,\nu)} \int c(x,y)d\pi(x,y)$



Dual formulation $\mathrm{OT}_c(\mu,\nu) = \max_{\psi\in L^\infty} \mathbf{E}_{X\sim\mu}\left[\psi^c(X)\right] + \mathbf{E}_{Y\sim\nu}\left[\psi(Y)\right]$

where $\psi^c(x) = \min_y \left[c(x,y) - \psi(y)\right]$ is the c-transform of ψ

Formulation in our semi-discrete case

Goal minimize w.r.t. θ

$$W(\theta) := \mathrm{OT}_c(\mu_{\mathsf{fake}}(\theta), \mu_{\mathsf{target}}) = \max_{\psi \in \mathbf{R}^m} F(\psi, \theta)$$

where

$$F(\psi, \theta) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{E}_{Z \sim \zeta} \left[\psi^c(P_i \circ g_\theta(Z)) + \frac{1}{m} \sum_{j=1}^{m} \psi_j \right]$$

Remarks

- $F(\psi, \theta)$ is concave in ψ
- min-max formulation similar to GAN methods but the potential ψ acts as a discriminator (no need of NN)

Gradient descent algorithm

Proposition under assumptions on g_θ and c, W is differentiable for a-e θ and

$$\nabla W(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{E}_{Z\sim\zeta}\left[(\partial_\theta g(\theta, Z))^T \nabla\psi^{*c}(P_i g(\theta, Z))\right]$$

whenever both terms exists and with ψ^* an optimal potential

Sketch of algorithm

Repeat:

1. approach ψ^* with gradient ascent
2. for z sampled from ζ update θ with the stochastic gradient

Part 2 – Texture synthesis by minimization with respect to an image θ

Single scale algorithm

Let us denote $\{y_1, \dots, y_m\}$ the patches of the target image

Proposition for u, ψ^* s.t. $\sigma(i) = \mathrm{argmin}_j \|P_i\theta - y_j\|^2 - \psi_j^*$ unique

$$\nabla_u W(\theta) = \frac{1}{n}\sum_{i=1}^{n} P_i^T (P_i\theta - y_{\sigma(i)})$$

The algorithm presented before reads:

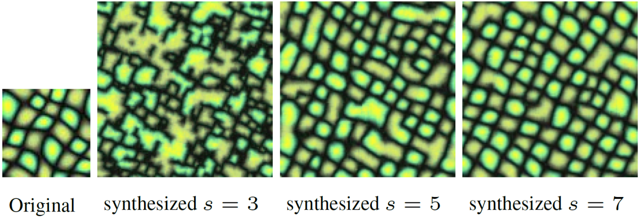
- approach ψ_k^* with gradient ascent and compute σ_k
- perform a gradient step for θ that correspond to

$$\theta_{k+1} = (1-\lambda)\theta_k + \lambda v_k$$

where

$$v_k = \frac{1}{p}\sum_{i=1}^{N} P_i^T y_{\sigma^k(i)},$$

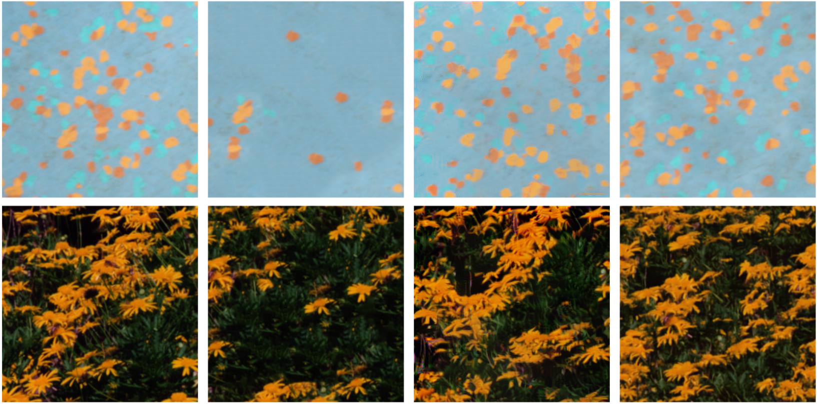
Multiscale algorithm



Take into account larger scales

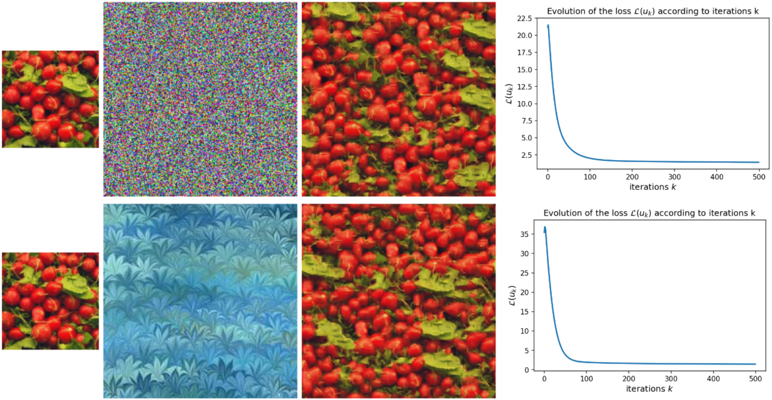
- create downsampled pyramid of images $\theta_l = S_l(\theta)$
- minimize $\sum_l OT({\mu_{\text{fake}}}^l, {\mu_{\text{target}}}^l)$

Some results



Original Kwatra Gatys Ours

Stability results



a. Sample b. Initialization c. Alg. 1 k=500 d. Loss

Other applications

Texture inpainting

Other applications

Texture barycenter

Summary

Advantages

- texture synthesis enforcing only patch distributions
- numerically stable method
- framework can actually be used with any feature distributions

Summary

Advantages

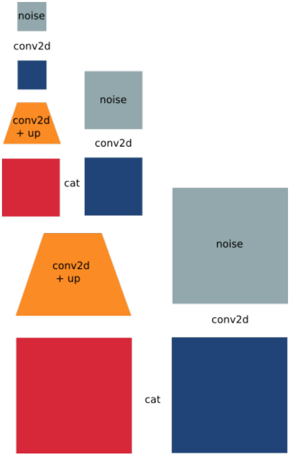
- texture synthesis enforcing only patch distributions
- numerically stable method
- framework can actually be used with any feature distributions

Downsides

- full optimization required for each synthesis
- gradient ascent algorithm for ψ may be long

Part 3 - Learning a generative model

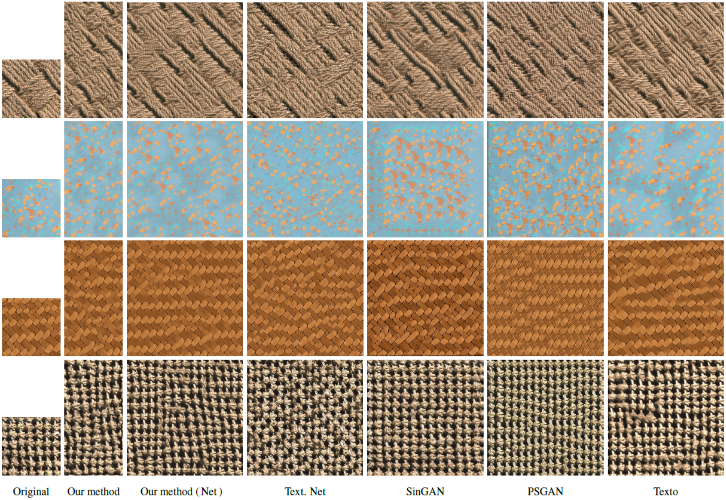
Simply plug a generative model into the framework!



Texture Network proposed by Ulyanov is a feed-forward convolutional neural network designed for texture generation

We use this architecture here!

CNN texture synthesis based on patch distributions



Conclusion

Conclusion

- produce similar results than state-of-the-art
- generative network learned only using patch distributions

Ideas of improvement

- designing a lighter network for generating textures
- find a better multiscale representation
- mix patches with other features

Thank you for your attention!

Paper and code available on my website and github

houdard.wp.imt.fr

github.com/ahoudard/wgenpatex

twitter.com/AntoineHou

Appendix – Evaluation of texture synthesis

SIFID (first max pooling Inception features)

	Alg. 1	Alg. 2	TexNet	SINGAN	PSGAN	TexTo
	0.43	1.13	**0.11**	0.93	0.27	1.22
	**0.02**	0.06	0.08	0.10	0.91	0.07
	**0.08**	0.18	0.18	0.17	1.14	0.18
	0.71	1.82	**0.17**	0.37	0.49	1.67
	0.31	0.80	**0.14**	0.39	0.70	0.79

VGG score (cross-correlation of VGG features)

$\times 10^3$	Alg. 1	Alg. 2	TexNet	SINGAN	PSGAN	TexTo
	**122**	233	218	299	224	260
	**6**	19	9	8	512	24
	141	151	**54**	207	753	152
	865	922	**190**	394	1366	1030
	283	331	**118**	227	714	367

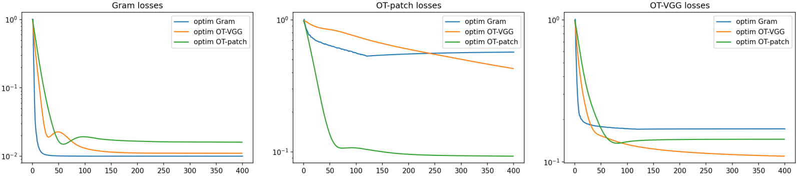
Proposed Multiscale OT distance

	Alg. 1	Alg. 2	TexNet	SINGAN	PSGAN	TexTo
	**0.45**	0.48	0.65	0.54	0.68	0.49
	**0.15**	0.16	0.24	0.24	0.43	0.16
	**0.09**	0.10	0.17	0.26	0.34	0.11
	**0.69**	0.78	1.22	0.79	1.19	0.75
	0.35	0.38	0.57	0.46	0.66	0.38

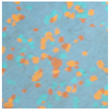
Appendix – Link with Gatys' algorithm

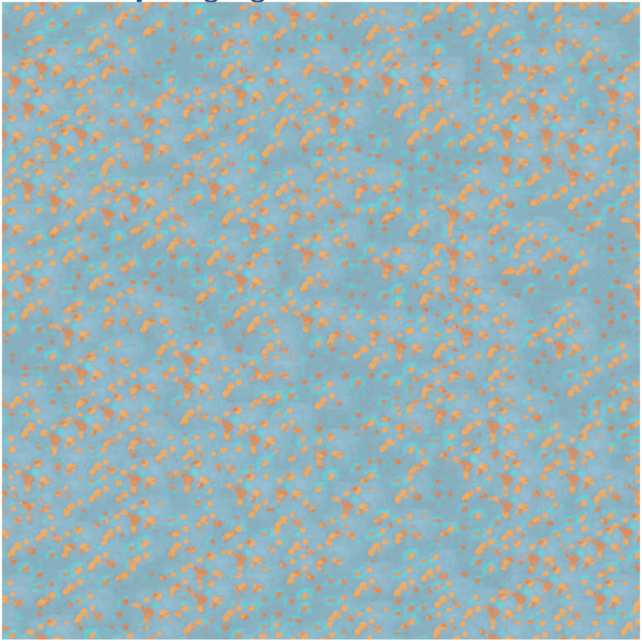
Gatys's texture synthesis: minimize w.r.t. image u in order to have VGG features at different scales $F_l(u)$ close to $F_l(v)$ in Gram loss.

- patch distribution and Gram loss $\rightarrow$ does not work
- patch distribution and OT loss $\rightarrow$ our algorithm
- VGG feature distribution and Gram loss $\rightarrow$ Gatys
- VGG feature distribution and OT loss $\rightarrow$ extension of our idea



Appendix – Arbitrarily large generation





Appendix – Link with WGAN

- in WGAN duality of W_1 distance yields the formulation

$$\min_{\theta} \max_{\psi \in Lip_1} \mathbf{E}_{\nu} [\psi(Y)] - \mathbf{E}_{\zeta} [\psi(g_{\theta}(Z))]$$

here $c(x, y) = \|x - y\|$, $c - conv(Y) = Lip_1(Y)$ and $\psi^c = -\psi$

Question how to compute ψ^*?

WGAN approach the potential with a deep neural network d_{η}

Issues

- enforcing Lip_1 is hard
- the neural network may fail to approach ψ^*
- a lot of parameters

Our approach allows to use more general OT cost and get rid of the neural network for ψ!