Modèles statistiques pour l'image
Patch-based Image Processing

Julie Digne

LIRIS - CNRS

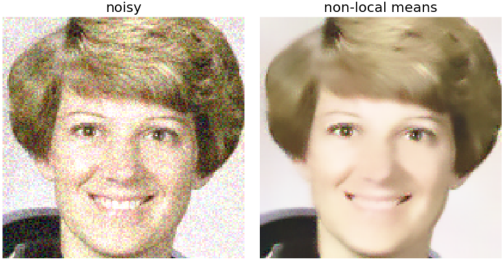
17/09/2025

Outline

Patch-based processing

- Consider patches instead of pixels

Similarity Analysis: Non Local Means [Buadès et al. 2005]



- Idea: denoise a point by comparing it to similar neighborhoods
- Compute local patch $P(p)$ around each point p
- Similarity measure between two points: $w(p, q) = \exp - \frac{dist(P(p), P(q))^2}{\sigma}$
- Update of the image :

$$I_{new}(p) = \frac{\sum_{q \in I} w(p, q) I(q)}{\sum_{q \in I} w(p, q)}$$

Example

Example

Example

Initial image

Noisy image

Gaussian filter result

Gaussian filter result

Gaussian filter result

Median result

Median result

NLmeans result

Comparison

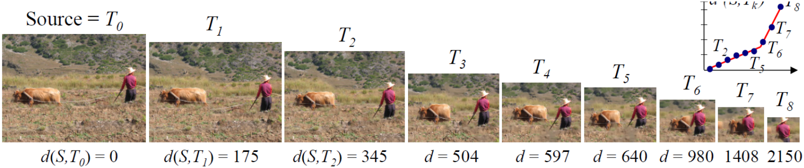



Outline

Visual Summary

Goal

Produce a smaller image that summarizes the content of the larger image



[Simakov 2008]

Comparing two images

Bidirectional Distance (BDS) [Simakov 2008]

Source image S, target image T:

$$d_{BDS}(S,T) = \frac{1}{N_S}\sum_{s\subset S}\min_{t\subset T} D(s,t) + \frac{1}{N_T}\sum_{t\subset T}\min_{s\subset S} D(t,s)$$

where s and t are patches of fixed size of S and T. D is the sum of squared difference between patches.

Comparing two images

Bidirectional Distance (BDS) [Simakov 2008]

Source image S, target image T:

$$d_{BDS}(S, T) = \frac{1}{N_S} \sum_{s \subset S} \min_{t \subset T} D(s, t) + \frac{1}{N_T} \sum_{t \subset T} \min_{s \subset S} D(t, s)$$

where s and t are patches of fixed size of S and T. D is the sum of squared difference between patches.

- Completeness term

Comparing two images

Bidirectional Distance (BDS) [Simakov 2008]

Source image S, target image T:

$$d_{BDS}(S,T) = \frac{1}{N_S}\sum_{s\subset S}\min_{t\subset T} D(s,t) + \frac{1}{N_T}\sum_{t\subset T}\min_{s\subset S} D(t,s)$$

where s and t are patches of fixed size of S and T. D is the sum of squared difference between patches.

- Completeness term
- Coherence term

Comparing two images

Bidirectional Distance (BDS) [Simakov 2008]

Source image S, target image T:

$$d_{BDS}(S,T) = \frac{1}{N_S}\sum_{s \subset S}\min_{t \subset T} D(s,t) + \frac{1}{N_T}\sum_{t \subset T}\min_{s \subset S} D(t,s)$$

where s and t are patches of fixed size of S and T. D is the sum of squared difference between patches.

- Completeness term
- Coherence term

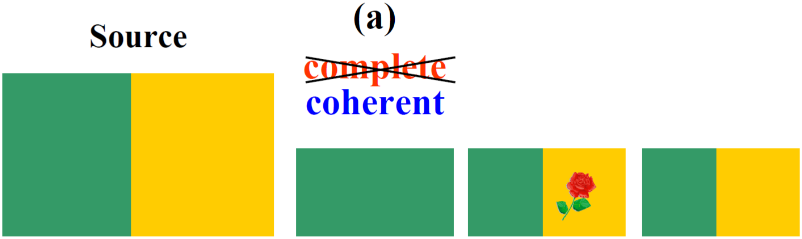
Reconstruction

Starting from an initial guess T_0 for T, build an image iteratively as the minimizer T of $d_{BDS}(S,T)$

Similarity distance

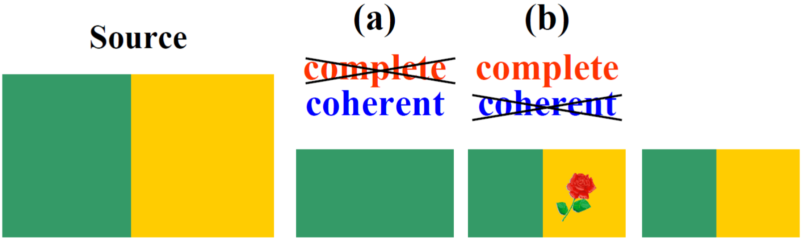
[Simakov 2008]

Similarity distance



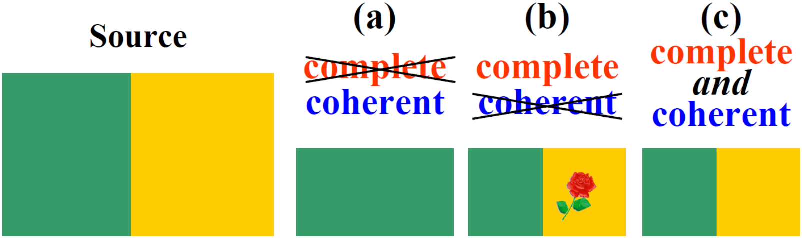
[Simakov 2008]

Similarity distance



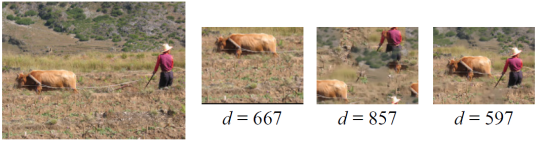
[Simakov 2008]

Similarity distance



[Simakov 2008]

Similarity distance



[Simakov 2008]

A three steps algorithm

1. For each patch of S find its closest patch on T

A three steps algorithm

1. For each patch of S find its closest patch on T
2. For each patch of T find its closest patch on S

A three steps algorithm

1. For each patch of S find its closest patch on T
2. For each patch of T find its closest patch on S
3. Aggregate color of nearest patches and update colors of T

A three steps algorithm

1. For each patch of S find its closest patch on T
2. For each patch of T find its closest patch on S
3. Aggregate color of nearest patches and update colors of T

Steps 1 - 2

The two first steps consist in applying nearest patch search. It needs to be done *efficiently*.

Aggregation Step: contribution of a pixel to the coherence measure

- Let q be a pixel of T, q lies inside m neighboring patches $Q_1, Q_2, \cdots, Q_m$

Aggregation Step: contribution of a pixel to the coherence measure

- Let q be a pixel of T, q lies inside m neighboring patches $Q_1, Q_2, \cdots, Q_m$
- These patches are matched to $P_1, P_2, \cdots, P_m$

Aggregation Step: contribution of a pixel to the coherence measure

- Let q be a pixel of T, q lies inside m neighboring patches $Q_1, Q_2, \cdots, Q_m$
- These patches are matched to $P_1, P_2, \cdots, P_m$
- The positions corresponding to q in $P_1, P_2, \cdots, P_m$ are $p_1, \cdots, p_m$

Aggregation Step: contribution of a pixel to the coherence measure

- Let q be a pixel of T, q lies inside m neighboring patches $Q_1, Q_2, \cdots, Q_m$
- These patches are matched to $P_1, P_2, \cdots, P_m$
- The positions corresponding to q in $P_1, P_2, \cdots, P_m$ are $p_1, \cdots, p_m$

Contribution

$$\frac{1}{N_T} \sum_{i=1}^{m} \|S(p_i) - T(q)\|_2^2$$

Aggregation Step: contribution of a pixel to the completeness measure

- Let q be a pixel of T,

Aggregation Step: contribution of a pixel to the completeness measure

- Let q be a pixel of T,
- q lies inside n neighboring patches $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_n$ that are the nearest patch to some patches of S $\hat{P}_1, \hat{P}_2, \cdots \hat{P}_n$

Aggregation Step: contribution of a pixel to the completeness measure

- Let q be a pixel of T,
- q lies inside n neighboring patches $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_n$ that are the nearest patch to some patches of S $\hat{P}_1, \hat{P}_2, \cdots \hat{P}_n$
- The positions corresponding to q in $\hat{P}_1, \hat{P}_2, \cdots, \hat{P}_m$ are $\hat{p}_1, \cdots, \hat{p}_m$

Aggregation Step: contribution of a pixel to the completeness measure

- Let q be a pixel of T,
- q lies inside n neighboring patches $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_n$ that are the nearest patch to some patches of S $\hat{P}_1, \hat{P}_2, \cdots \hat{P}_n$
- The positions corresponding to q in $\hat{P}_1, \hat{P}_2, \cdots, \hat{P}_m$ are $\hat{p}_1, \cdots, \hat{p}_m$

Contribution

$$\frac{1}{N_S} \sum_{i=1}^{n} \|S(\hat{p}_i) - T(q)\|_2^2$$

Color update

Color Update

The best $T(q)$ should minimize:

$$\frac{1}{N_S}\sum_{i=1}^{n}\|S(\hat{p}_i) - T(q)\|_2^2 + \frac{1}{N_T}\sum_{i=1}^{m}\|S(p_i) - T(q)\|_2^2$$

Color update

Color Update

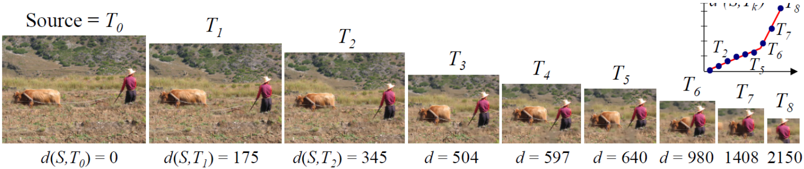
The best $T(q)$ should minimize:

$$\frac{1}{N_S}\sum_{i=1}^{n}\|S(\hat{p}_i)-T(q)\|_2^2+\frac{1}{N_T}\sum_{i=1}^{m}\|S(p_i)-T(q)\|_2^2$$

Color Update

$$T(q)=\frac{\frac{1}{N_S}\sum_{i=1}^{n}S(\hat{p}_i)+\frac{1}{N_T}\sum_{i=1}^{m}S(p_i)}{\frac{m}{N_T}+\frac{n}{N_S}}$$

Visual Summary



[Simakov 2008]

Gradual resizing

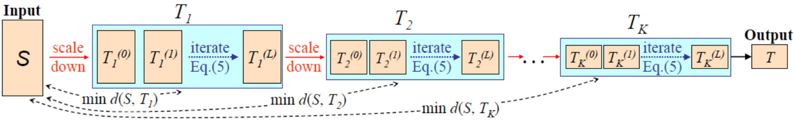
- When the target has a very different size from the source: what is a good initial guess?

Gradual resizing

- When the target has a very different size from the source: what is a good initial guess?
- Iterative process: downsample the image and apply the reconstruction

Gradual resizing

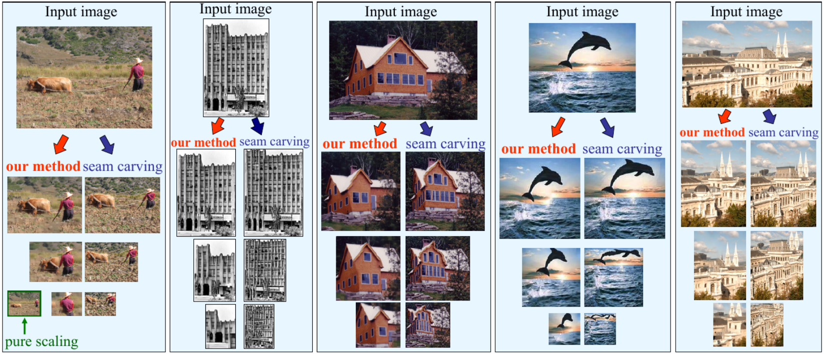
- When the target has a very different size from the source: what is a good initial guess?
- Iterative process: downsample the image and apply the reconstruction



[Simakov 2008]

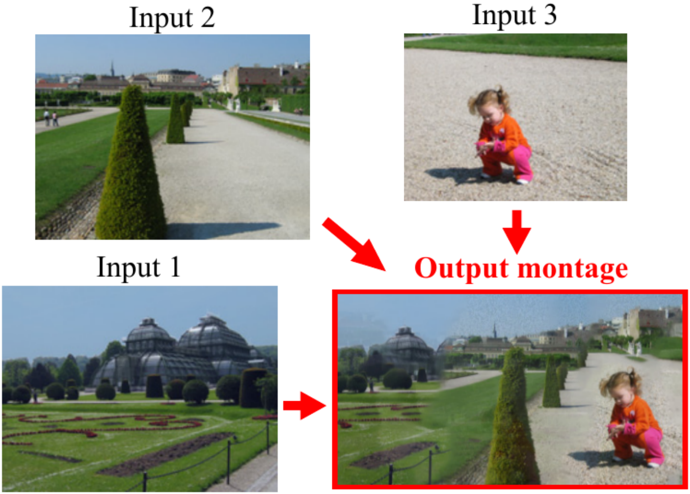
video

Visual Summary



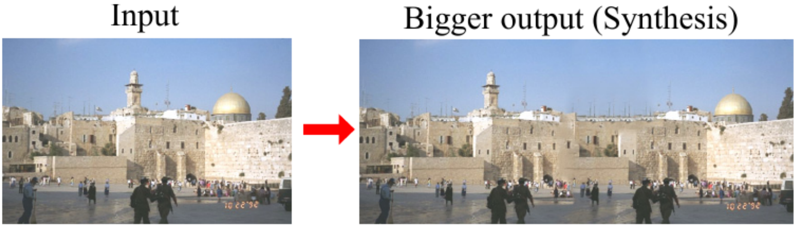
[Simakov 2008]

Montage



[Simakov 2008]

Synthesis



[Simakov 2008]

Key ingredient for all these methods

Requirement

A fast method to find similar patches

- Naive way: traverse the whole image at each query

Key ingredient for all these methods

Requirement

A fast method to find similar patches

- Naive way: traverse the whole image at each query
- Better: put all patches in a search structure

Key ingredient for all these methods

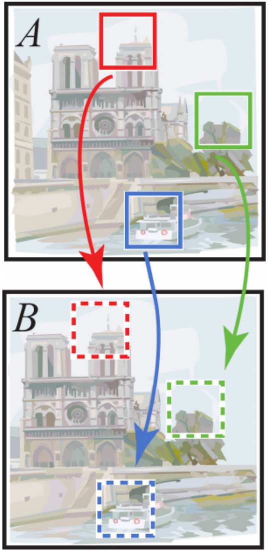
Requirement

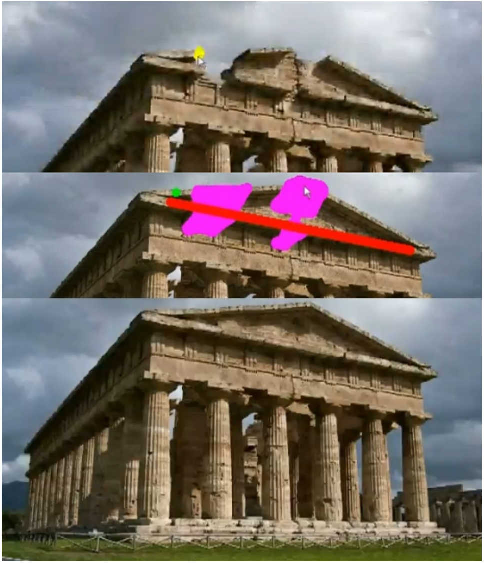
A fast method to find similar patches

- Naive way: traverse the whole image at each query
- Better: put all patches in a search structure
- Even better: the patch match algorithm

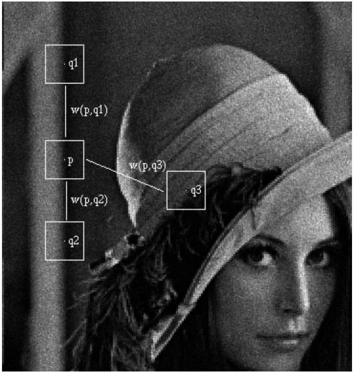
Outline

Patch Match [Barnes 2009]





Similar patches



[Buadès 2005]

Similarity distance

The similarity distance between two patches p_A, p_B of size $n \times n$ is computed as $\sum_{1 \le i,j \le n} \|p_A(i,j) - p_B(i,j)\|_2^2$.

Similarity

Two patches are considered as similar is their similarity distance is small.

Patch Match

Goal

Given an image A and an image B find *efficiently* for all patches of image A an approximate nearest patch of image B.

Patch Match

Goal

Given an image A and an image B find *efficiently* for all patches of image A an approximate nearest patch of image B.

Patch Match Principle

Assume we have found a patch p_B of B corresponding to a given patch p_A of A, assume we have a patch p'_A located close to p_A in image A, then its corresponding patch p'_B has a high probability to lie close to p_B

- Look for p'_B close to p_B.

What if?

- If we have an initial corresponding pairs (p_A, p_B) then the search is made easier

What if?

- If we have an initial corresponding pairs (p_A, p_B) then the search is made easier
- However: How can we find an initial pair?

What if?

- If we have an initial corresponding pairs (p_A, p_B) then the search is made easier
- However: How can we find an initial pair?
- However: Should we start with a single initial pair? Why not many?

What if?

- If we have an initial corresponding pairs (p_A, p_B) then the search is made easier
- However: How can we find an initial pair?
- However: Should we start with a single initial pair? Why not many?

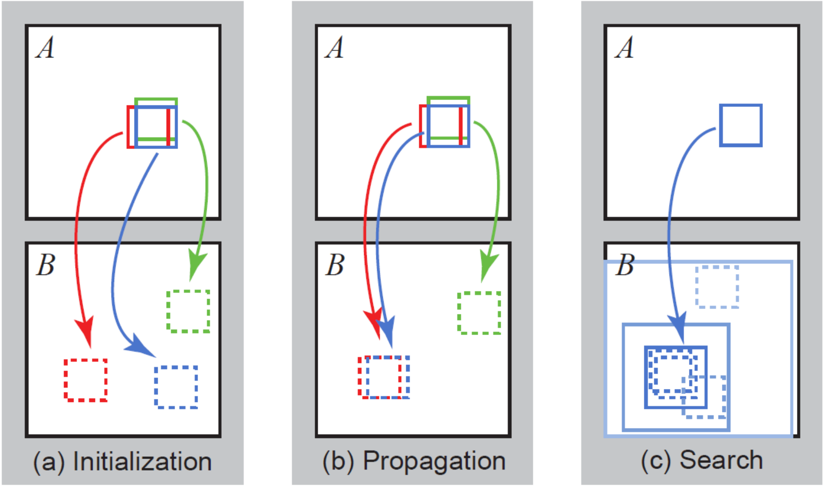
Notation

Let p_A be a patch centered at a in image A and p_B a patch centered at b in image B. We define an offset vector $f(a)$ as $f(a) = b - a$. The set of all offset vectors is called the Nearest Neighbor Field (NNF).

Algorithm

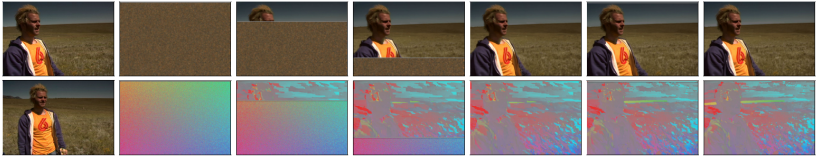
1. Initialize the NNF with random vectors
2. **Propagation:** for $i = 1 \cdots M$, for $j = 1 \cdots M$
    1. Evaluate the offset $f(i-1,j)$, $f(i-1,j-1)$, $f(i-1,j+1)$ and $f(i,j-1)$
    2. If one of them is better than $f(i,j)$ replace $f(i,j)$ with it.
3. **Randomization:** For all (i,j), draw a random offset w, if w is better than $f(i,j)$ set $f(i,j) = w$

Algorithm



[Barnes 2009]

Algorithm



(a) originals (b) random (c) $\frac{1}{4}$ iteration (d) $\frac{3}{4}$ iteration (e) 1 iteration (f) 2 iterations (g) 5 iterations

[Barnes 2009]

Probabilistic analysis

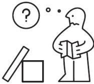
Exercise

Assume we have two images of size M and assign randomly patches of image A to patches of image B (+ unicity of the correspondence)

Probabilistic analysis

Exercise

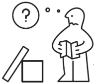
Assume we have two images of size M and assign randomly patches of image A to patches of image B (+ unicity of the correspondence)

- What is the probability that at least one patch is indeed paired to its corresponding patch?

Probabilistic analysis

Exercise

Assume we have two images of size M and assign randomly patches of image A to patches of image B (+ unicity of the correspondence)

- What is the probability that at least one patch is indeed paired to its corresponding patch?

Simplification

Assume that a pair is correct if a patch is assigned to a patch that is spatially close (in a neighborhood of size C) to its true correspondence

Probabilistic analysis

Exercise

Assume we have two images of size M and assign randomly patches of image A to patches of image B (+ unicity of the correspondence)

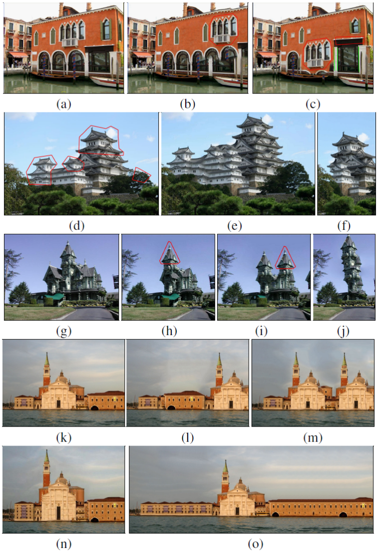
- What is the probability that at least one patch is indeed paired to its corresponding patch?

Simplification

Assume that a pair is correct if a patch is assigned to a patch that is spatially close (in a neighborhood of size C) to its true correspondence

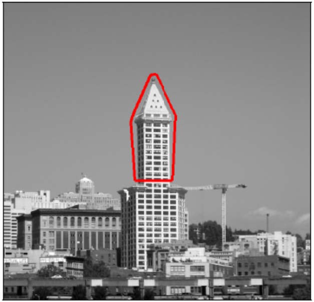
- What is the probability that at least one patch is paired to an approximate corresponding patch?

Reshuffling Application



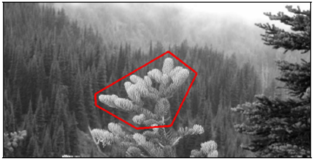
[Barnes 2009]

Deformation Application



(a) building marked by user

(b) scaled up, preserving texture



(c) bush marked by user

(d) scaled up, preserving texture.

[Barnes 2009]

Outline

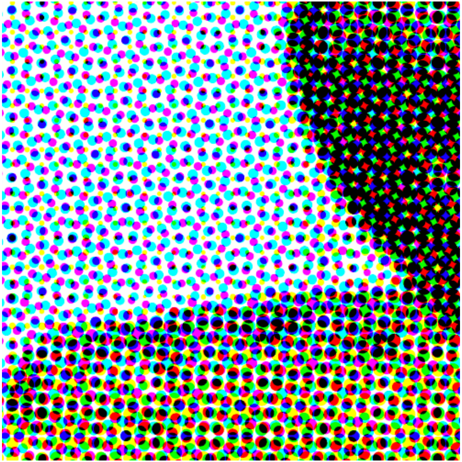
Half-toning problem

- Generating half-toning images: *Dithering*
- For example: using only black ink for printing a grayscale image
- Color case: 4 inks instead of 255^3 possible values.

Half-toning problem

- Generating half-toning images: *Dithering*
- For example: using only black ink for printing a grayscale image
- Color case: 4 inks instead of 255^3 possible values.
- We will study only grayscale images.

Color Example



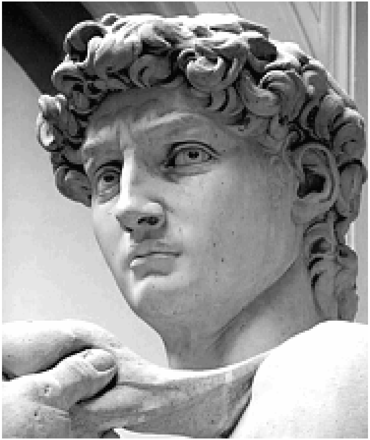
TrameQuadri Zewan — Image Wikimedia

Half-toning in a nutshell

Half-toning

Create a binary approximation of a grayscale image which appears to be *continuous*.

Example



Original Image
(Wikipedia, user:Gerbrant)

Example

Quantification: No continuity effect.
(Image Wikipedia, user:Gerbrant)

Principle

Half-toning Principle

Print black dots to give the illusion of gray values.

Principle

Half-toning Principle

Print black dots to give the illusion of gray values.

- The human eye integrates the dots and perceives a flat colour.

Principle

Half-toning Principle

Print black dots to give the illusion of gray values.

- The human eye integrates the dots and perceives a flat colour.

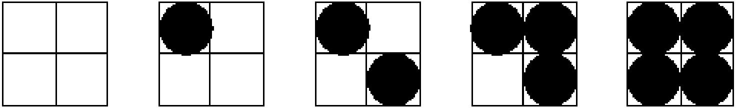
Problem

Choose a layout of points that minimize visual artefacts.

Dithering patterns

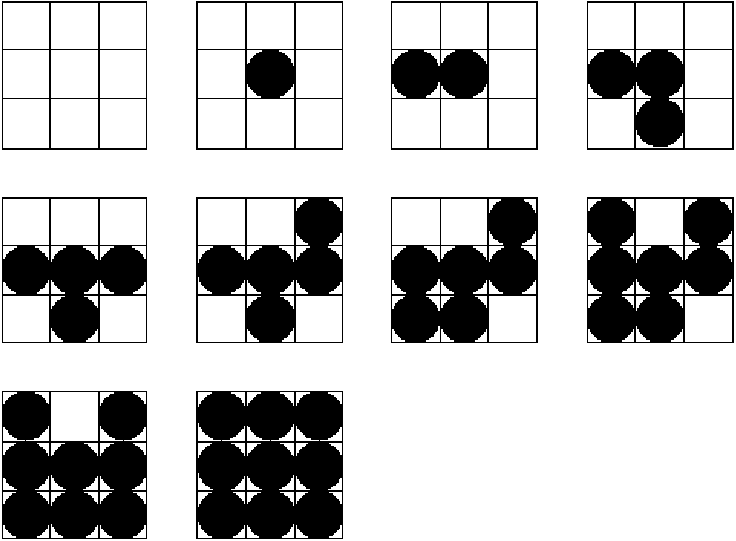
- The printed image is paved by dithering patterns
- Each dithering pattern contains a distribution of black and white dots.
- Each pattern gives a gray level corresponding to the ratio of black/white pixels.

Pattern example



2x2 Patterns

Pattern example



3x3 Patterns

First method

- Each pixel corresponds to a square pattern
- The pixel value is encoded by the corresponding pattern

Remark

Printing

- On a professionnal printing device 1200 dpi,4*4 binary dots per pixel.
- On a 300dpi printer, only 1 binary dot pixel.

Choosing the layout of the dots

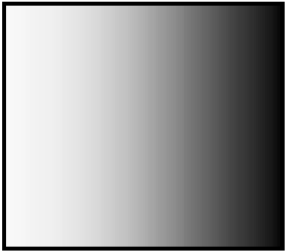
Goals

Layout algorithms aim at obtaining good gray values while minimizing the artefacts

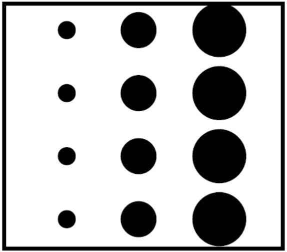
- Several algorithms exist (regular layout, irregular layout, dots centered or not centered in the patterns...).

Classical Halftoning

Dot areas are proportional to the image intensity.

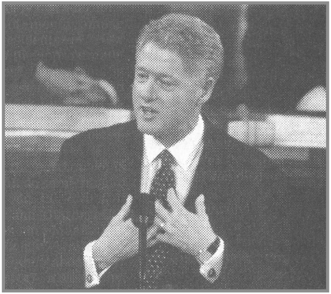


I(x,y)

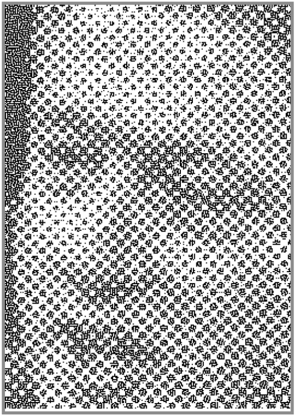


P(x,y)

Example



Newspaper Image



From New York Times, 9/21/99

Image T.A. Funkhouser

Dithering

- Random dithering
- Ordered dithering
- Error-diffusion dithering

Random Dithering

- Instead of using a fixed threshold, use a random one

Random Dithering

- Instead of using a fixed threshold, use a random one per pixel

Random Dithering

- Instead of using a fixed threshold, use a random one

Ordered Dithering

- The random thresholds are replaced by local schemes stored in matrices

For dithering patterns of size 2×2:

$$D_2 = \begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix}$$

Ordered Dithering

Algorithm 1: Ordered Dithering

Input: Grayscale image I, matrix D_n ($\mathbb{R}^{n \times n}$)

Output: A binary image J

for *all pixels* x, y **do**
  $i = x$ modulo n;
  $j = y$ modulo n;
  if $I(x, y) > D(i, j)$ **then**
    $J(x, y) = 1$;
  else
    $J(x, y) = 0$;

Ordered Dithering

- Bayer matrices for dithering

$$D_n = \begin{bmatrix} 4D_{n/2} + D_2(1,1)U_{n/2} & 4D_{n/2} + D_2(1,2)U_{n/2} \\ 4D_{n/2} + D_2(2,1)U_{n/2} & 4D_{n/2} + D_2(2,2)U_{n/2} \end{bmatrix}$$

$$D_2 = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \qquad D_4 = \begin{bmatrix} 15 & 7 & 13 & 5 \\ 3 & 11 & 1 & 9 \\ 12 & 4 & 14 & 6 \\ 0 & 8 & 2 & 10 \end{bmatrix}$$

Ordered Dithering

Often used for journal printing.

Dithering with error diffusion: Floyd-Steinberg

Principle

Distribute the error on neighboring pixels.

Dithering with error diffusion: Floyd-Steinberg

Principle

Distribute the error on neighboring pixels.

- Threshold intensity value of $\textit{threshold}(I(x,y))$

Dithering with error diffusion: Floyd-Steinberg

Principle

Distribute the error on neighboring pixels.

- Threshold intensity value of $\mathit{threshold}(I(x, y))$
- Error: $e = I(x, y) - \mathit{threshold}(I(x, y))$.

Dithering with error diffusion: Floyd-Steinberg

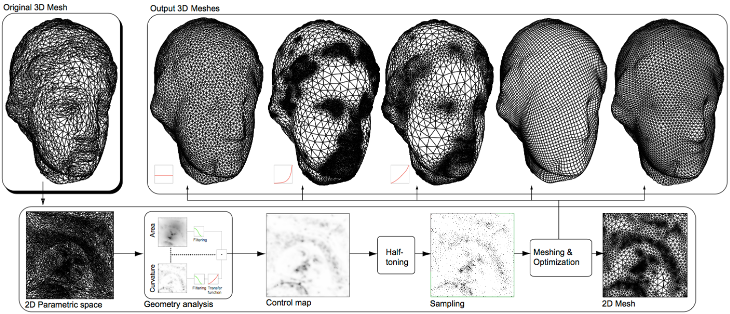
Principle

Distribute the error on neighboring pixels.

- Threshold intensity value of $\textit{threshold}(I(x,y))$
- Error: $e = I(x,y) - \textit{threshold}(I(x,y))$.
- Error distribution:
  - $I(x, y+1) = I(x, y+1) + \alpha e$
  - $I(x+1, y-1) = I(x+1, y-1) + \beta e$
  - $I(x+1, y) = I(x+1, y) + \gamma e$
  - $I(x+1, y+1) = I(x+1, y+1) + \delta e$
  - with $\alpha + \beta + \gamma + \delta = 1$

Dithering with error diffusion: Floyd-Steinberg

Remeshing via halftoning



Pierre Alliez et al. 2003.

- Some methods use optimal transportation to generate density samplings [DeGoes et al. 2012]