

Course-poursuite dans l'espace des images

Cécile Louchet

IDP, Université d'Orléans

Centre Galois 2025

Cet exposé reprend très largement l'article paru dans CVPR 2017

<https://phillipi.github.io/pix2pix/>

Image-to-Image Translation with Conditional Adversarial Nets

Phillip Isola

Jun-Yan Zhu

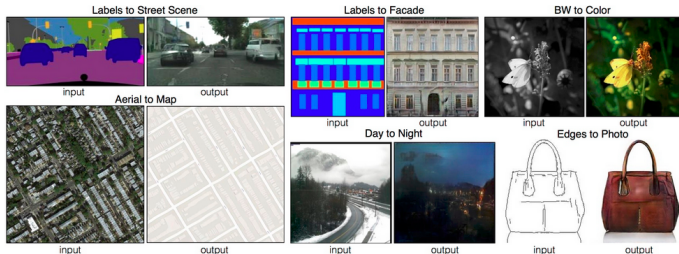
Tinghui Zhou

Alexei A. Efros

University of California, Berkeley
In CVPR 2017

[\[Paper\]](#)

[\[GitHub\]](#)



Example results on several image-to-image translation problems. In each case we use the same architecture and objective, simply training on different data.

Abstract

We investigate conditional adversarial networks as a general-purpose solution to image-to-image translation problems. These networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally would require very different loss formulations. We demonstrate that this approach is effective at synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images, among other tasks. As a community, we no longer hand-engineer our mapping functions, and this work suggests we can achieve reasonable results without hand-engineering our loss

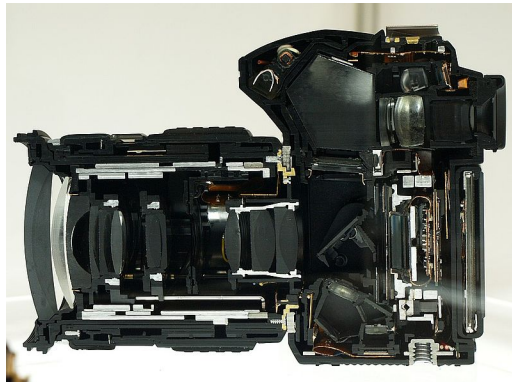
- 1 Pleins de trucs utiles en traitement d'image
 - Mais qu'est-ce qu'une image ?
 - Modèle sténopé, perspective, vision binoculaire
 - Couleur et applications
- 2 Contours
- 3 La course des faussaires contre la police
- 4 Construire le cerveau des faussaires et de la police

100



750×443 pixels. Source : <https://www.lartdelaphoto.fr/>

Appareil photo numérique



Source : <http://cgcamera.com/basics-of-a-digital-camera/>





ISO: 100 - 1,600 in full stops, also a trick ISO 3,200 mode.

Format: 23.6×15.8 mm. Source : <http://www.kenrockwell.com/nikon/d60.htm>

- niveau de gris entier, de 0 (noir) à 255 (blanc)

Source : <http://www.ie.wh.sdu.edu.cn>

- niveau de gris entier, de 0 (noir) à 255 (blanc) (codé sur 8 bits)

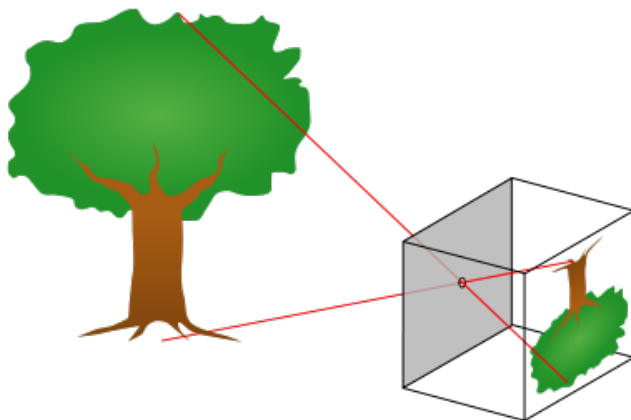
02

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺

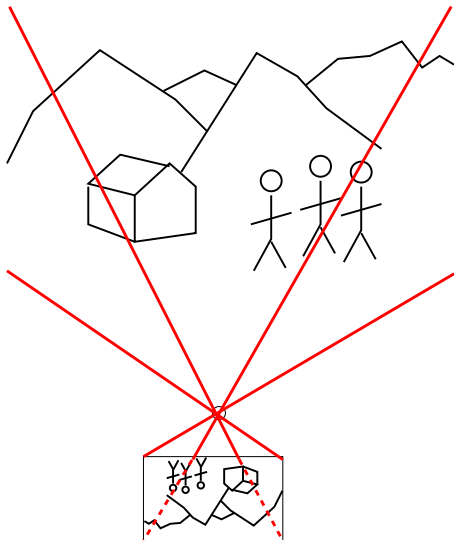
Row	Column																		
	223	208	194	182	172	164	159	158	159	164	172	182	194	208	223	240	225	213	202
	208	191	176	163	151	143	137	135	137	143	151	163	176	191	208	225	213	202	194
	194	176	159	144	131	121	115	113	115	121	131	144	159	176	194	213	202	194	182
	182	163	144	128	113	101	93	90	93	101	113	128	144	163	182	202	194	182	172
	172	151	131	113	96	81	71	68	71	81	96	113	131	151	172	193	182	172	164
	164	143	121	101	81	64	50	45	50	64	81	101	121	143	164	186	182	172	164
	159	137	115	93	71	50	32	23	32	50	71	93	115	137	159	182	182	172	164
	158	135	113	90	68	45	23	0	23	45	68	90	113	135	158	180	182	172	164
	159	137	115	93	71	50	32	23	32	50	71	93	115	137	159	182	182	172	164
	164	143	121	101	81	64	50	45	50	64	81	101	121	143	164	186	182	172	164
	172	151	131	113	96	81	71	68	71	81	96	113	131	151	172	193	182	172	164
	182	163	144	128	113	101	93	90	93	101	113	128	144	163	182	202	194	182	172
	194	176	159	144	131	121	115	113	115	121	131	144	159	176	194	213	202	194	182
	208	191	176	163	151	143	137	135	137	143	151	163	176	191	208	225	213	202	194
	223	208	194	182	172	164	159	158	159	164	172	182	194	208	223	240	225	213	202
	240	225	213	202	193	186	182	180	182	186	193	202	213	225	240	255	240	225	213

- ◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

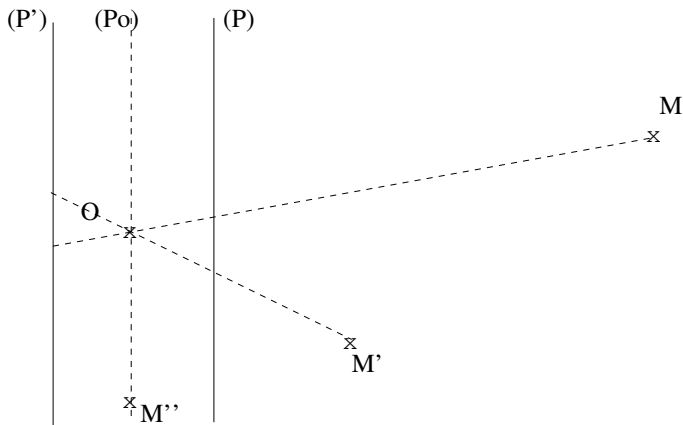




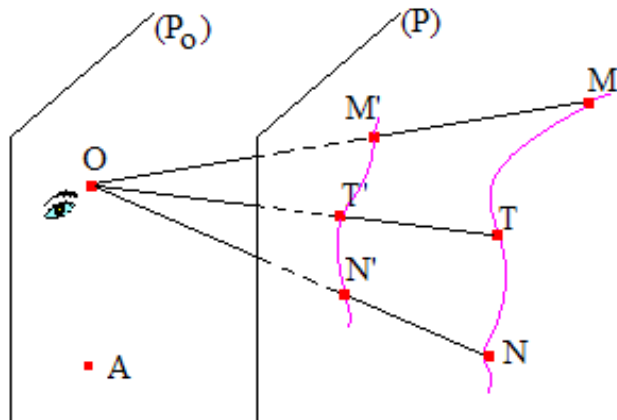
Sources : <http://www.diyphotography.net/23-pinhole-cameras-that-you-can-build-at-home/>
http://en.wikipedia.org/wiki/Pinhole_camera

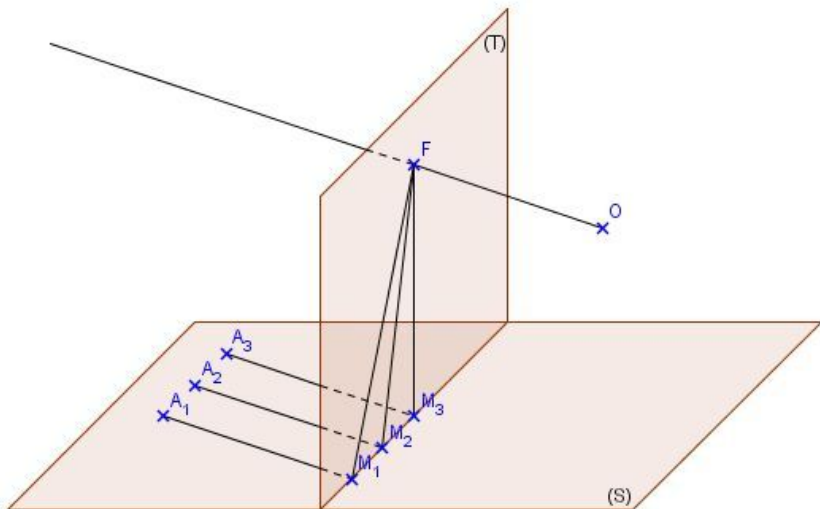


Perspective

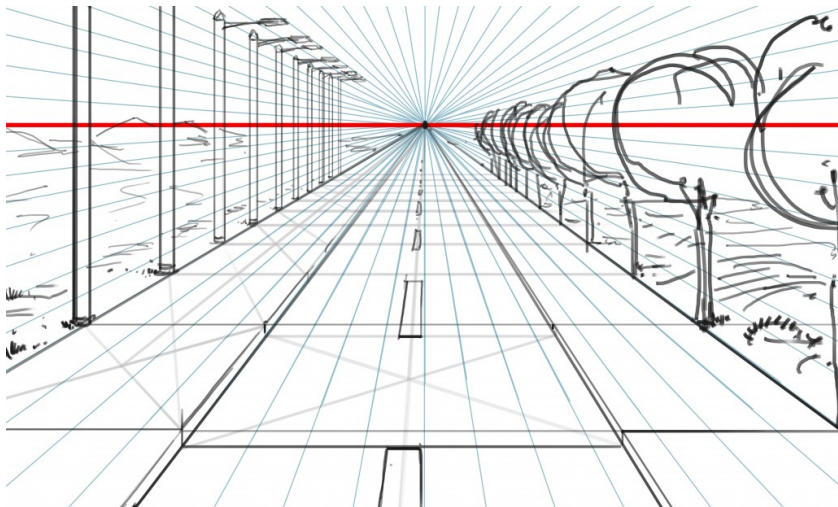


Perspective

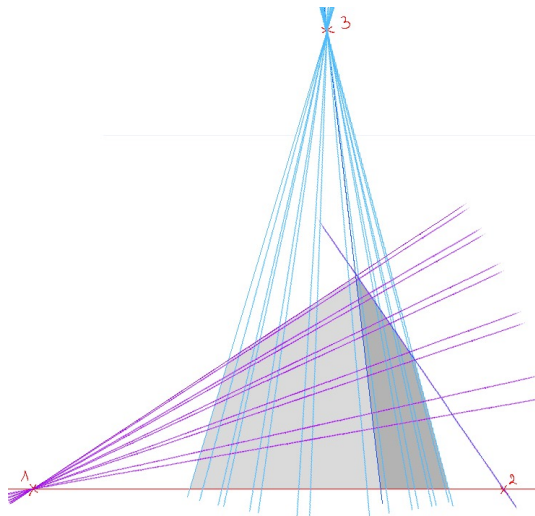




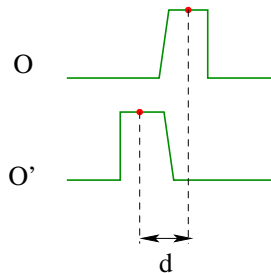
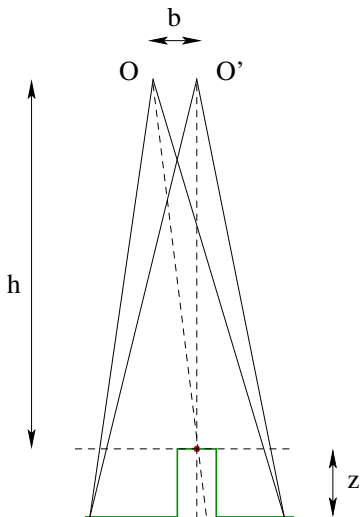
Perspective



Perspective



Vision binoculaire



$$z = ???$$

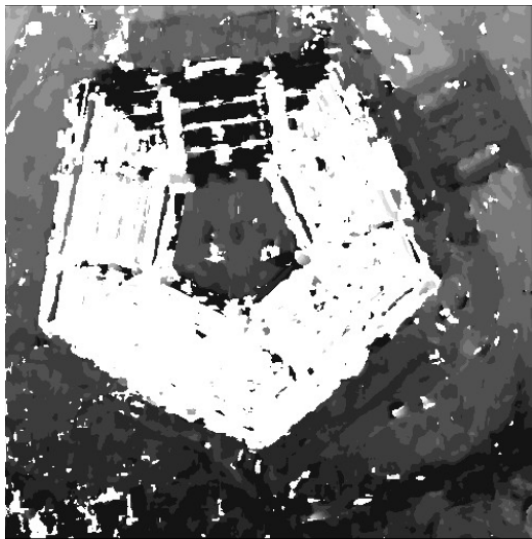
Reconstruction 3D



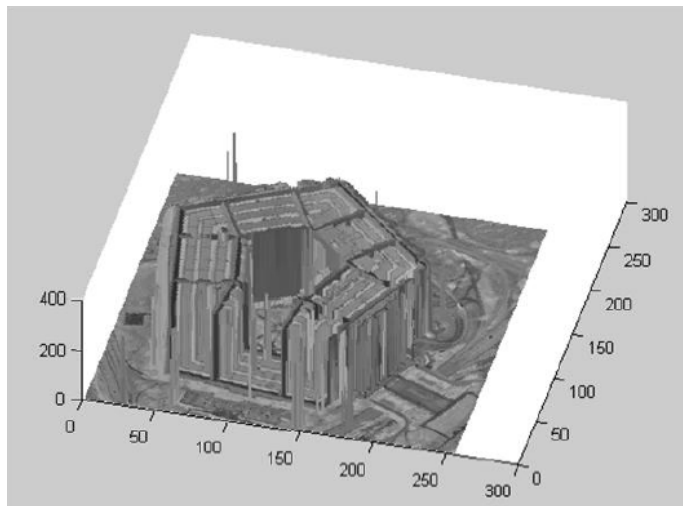
Reconstruction 3D



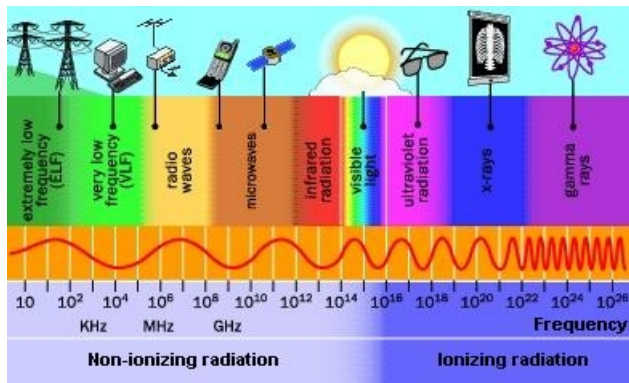
Reconstruction 3D



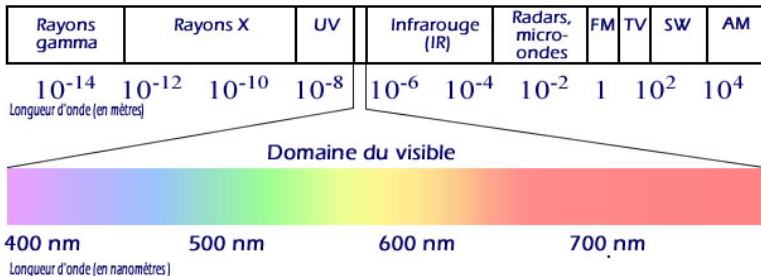
Reconstruction 3D



Spectre de la lumière

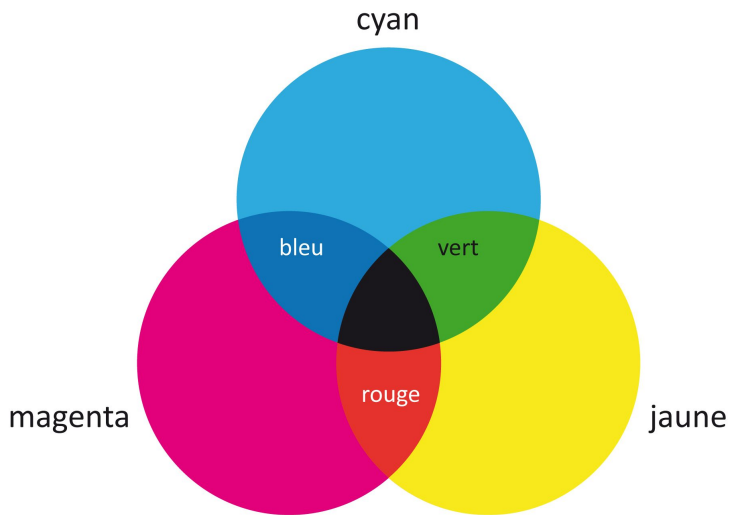


Source : http://lesantennespourlesnuls.pagesperso-orange.fr/page_notion_1.htm

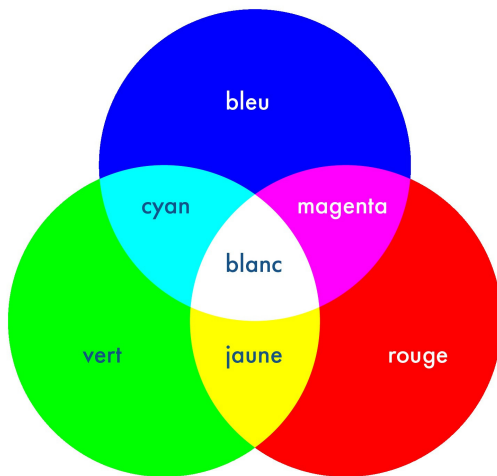


Source : http://physique.unice.fr/sem6/2008-2009/PagesWeb/SBM/D_un_point_de_vue_physique...html

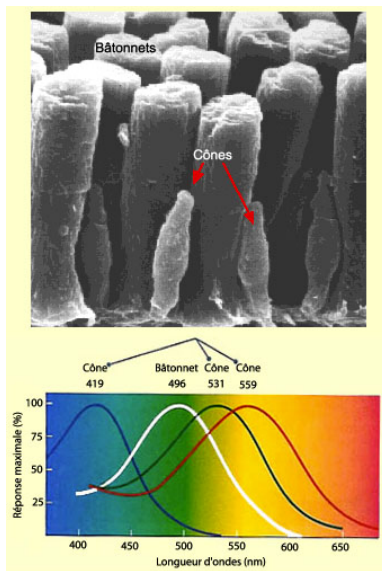
Cercle chromatique (synthèse soustractive)



Cercle chromatique (synthèse additive)



coloringmate.co



Vision trichromatique

- La rétine humaine comporte des “cônes” et des “bâtonnets”.
- Les **bâtonnets** des humains sont **monochromatiques** (*Dans la nuit, tous les chats sont gris*).
- Les **cônes** humains sont majoritairement **trichromatiques** (certains individus, surtout des femmes, seraient tétrachromatiques, avec une composante entre le rouge et le vert³).
- La plupart des animaux semblent trichromatiques. Néanmoins, certains insectes et oiseaux ont une vision tétrachromatique (avec une composante UV). D'autres animaux mono-, ou di-chromatiques.

Le chat a une vision dichromatique (pas de rouge)

Le lapin est dichromatique (pas de rouge)

Le dauphin a une vision monochromatique (vers le bleu)

³<http://en.wikipedia.org/wiki/Tetrachromacy>

Métamérisme, daltonisme

- **Métamérisme**⁴ : une couleur *physique* correspond à un spectre, est donc une fonction et vit dans un espace de dimension infinie. Notre œil la verra à travers ses 3 (ou 4 filtres). Deux couleurs différentes peuvent avoir la même projection sur mon système visuel humain à 3 (ou 4) dimensions. On les appelle “métamères”.
- **Daltonisme** : les cônes verts et rouges ont des spectres trop proches. D’autres types existent mais sont plus rares.

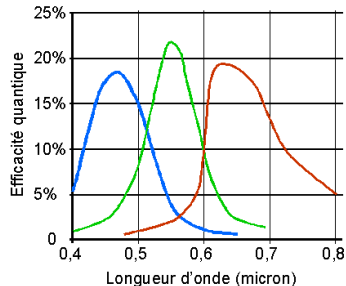
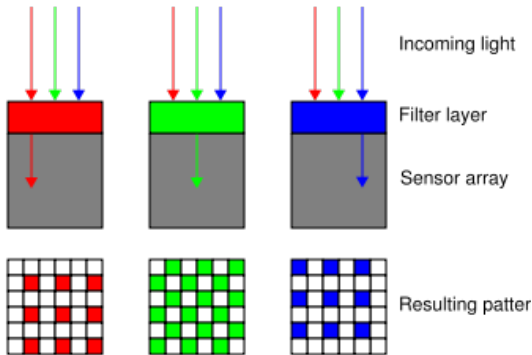
⁴ [http://en.wikipedia.org/wiki/Metamerism_\(color\)](http://en.wikipedia.org/wiki/Metamerism_(color))



⁵ http://en.wikipedia.org/wiki/Bayer_filter

6 <http://www.phaseone.com/en/Imaging-Software/Capture-One-Express-7/Features.aspx>

Filtres couleur

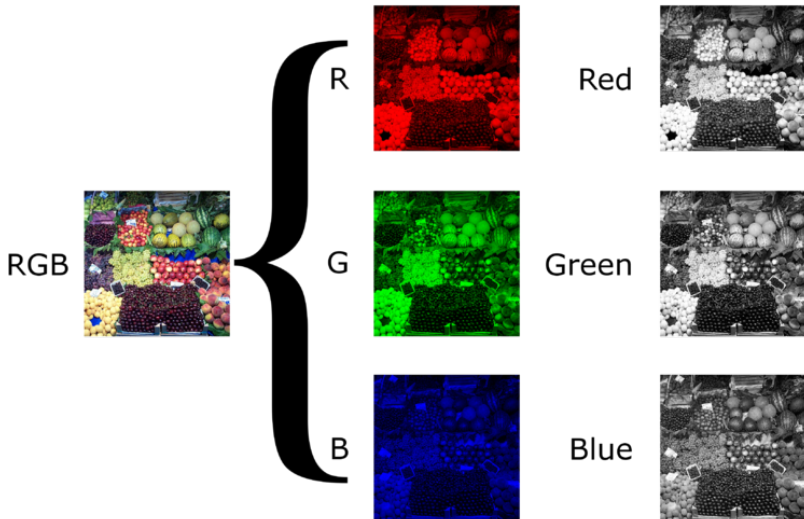
Récepteurs filtrés rouge, vert, bleu⁷ – sensibilité en fréquence⁸

⁷Source : http://commons.wikimedia.org/wiki/File:Bayer_pattern_on_sensor_profile.svg

⁸Source : http://commons.wikimedia.org/wiki/File:CCD_Pleine-Trame_Spectrale.png

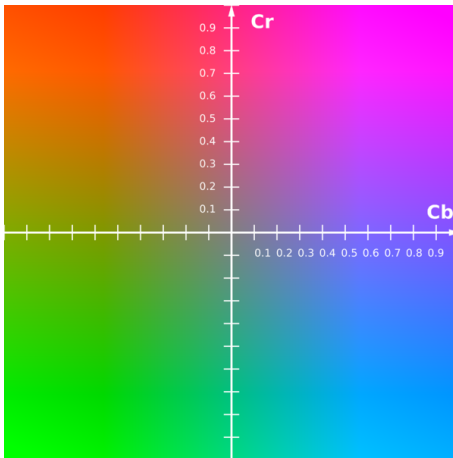
◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

Décomposition RGB



Décomposition YCbCr

Y : luminance ; Cb : chrominance bleue ; Cr : chrominance rouge.

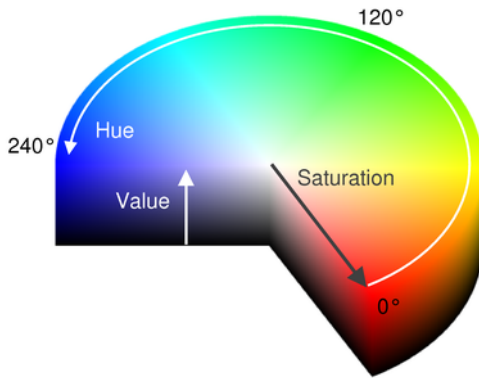


Plan Chrominance



Canaux Y-Cb-Cr⁹

Décomposition HSV



Espace HSV. H : *hue* (teinte). S : saturation. V : valeur.

Source <http://stackoverflow.com/questions/11598043/get-slightly-lighter-and-darker-color-from-uicolor>

Décomposition HSV



Hue

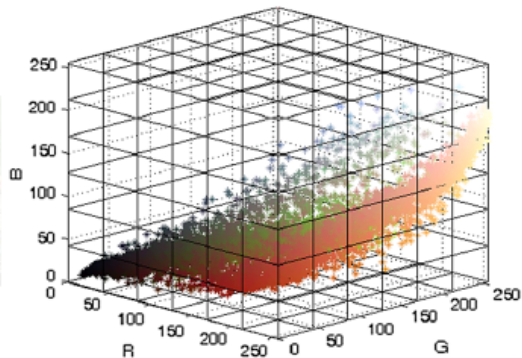


Saturation

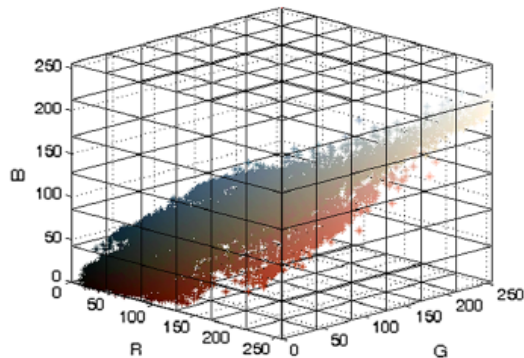


Value



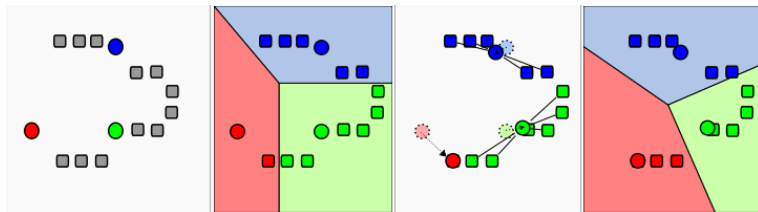


tiré de Vantaram, Saber, "Survey of contemporary trends in color image segmentation", *Journal of Electronic Imaging*

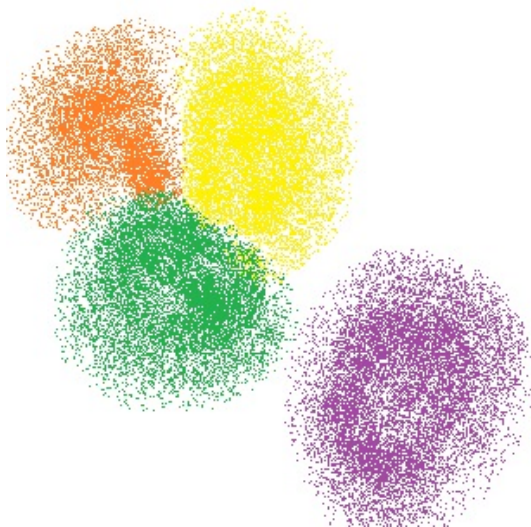


00

Algorithme des k -moyennes



Algorithme des k -moyennes



Résultats avec une version un peu améliorée



- 1 Pleins de trucs utiles en traitement d'image
 - Mais qu'est-ce qu'une image ?
 - Modèle sténopé, perspective, vision binoculaire
 - Couleur et applications
- 2 Contours
- 3 La course des faussaires contre la police
- 4 Construire le cerveau des faussaires et de la police

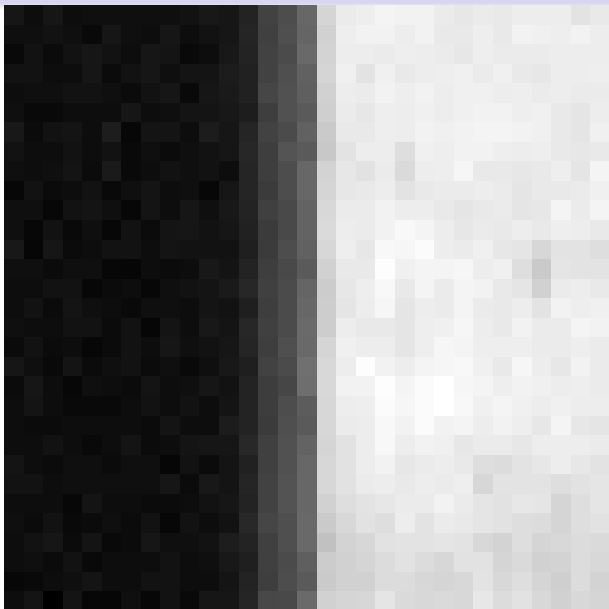
Gradient



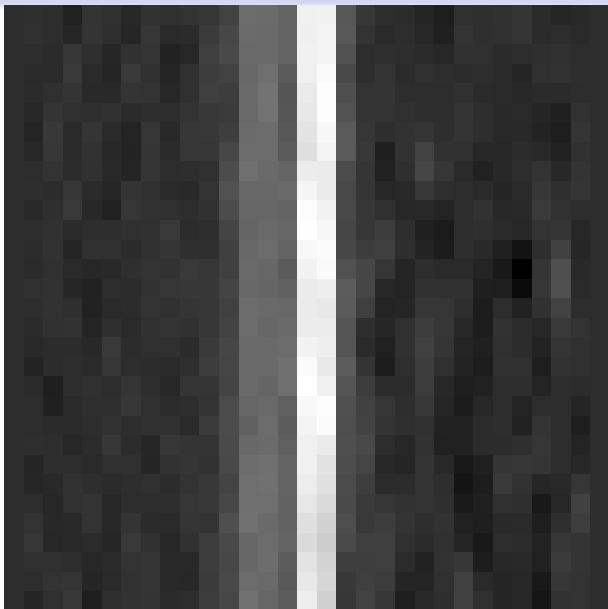
Gradient



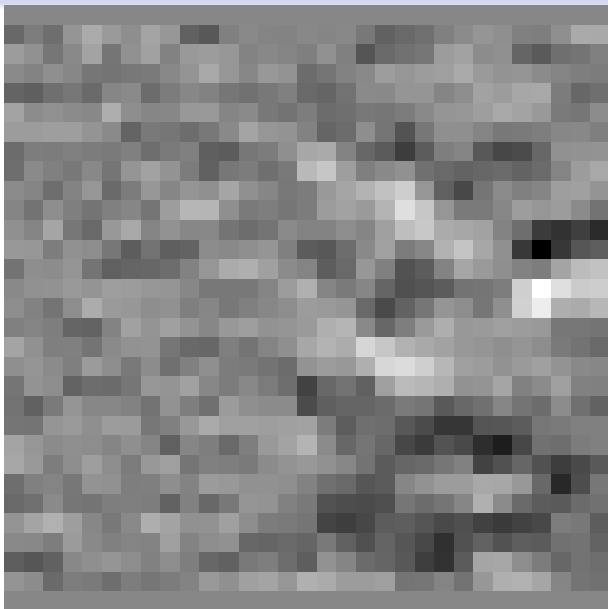
Gradient



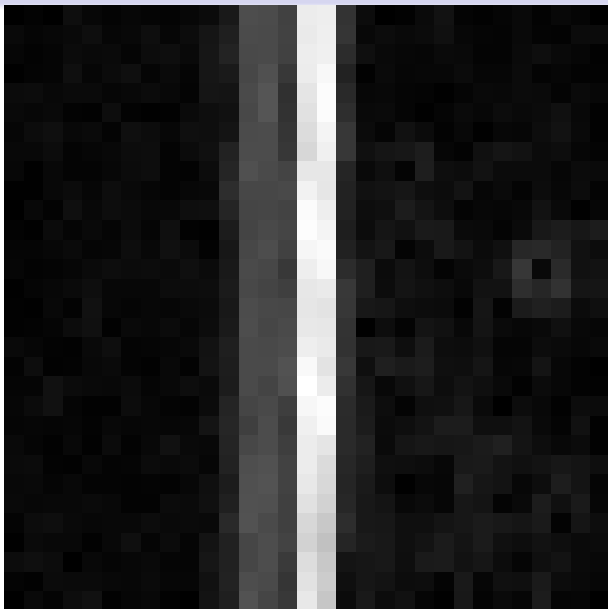
Gradient



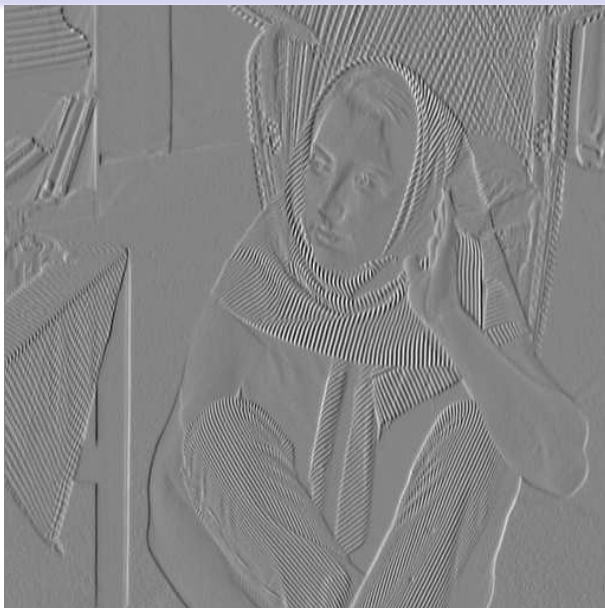
Gradient



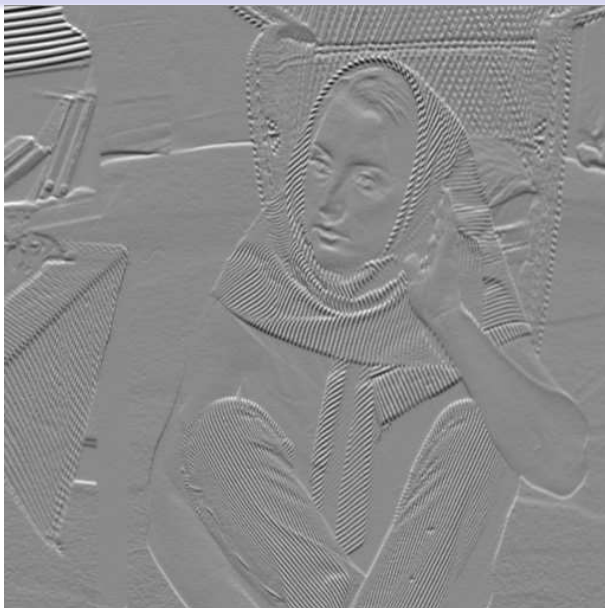
Gradient



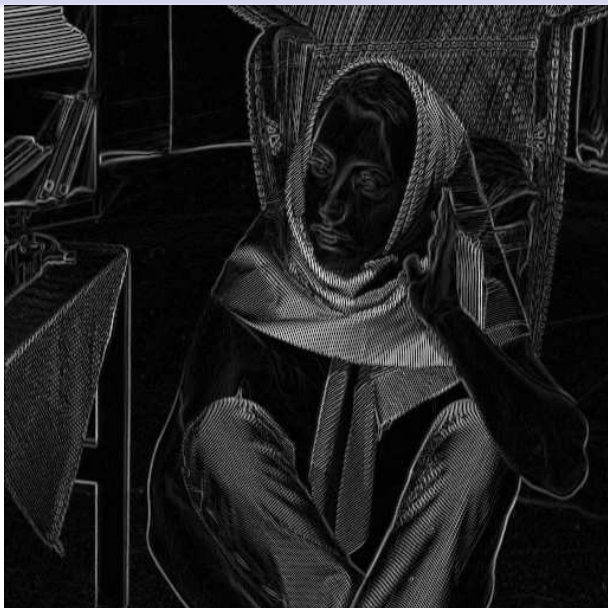
Gradient



Gradient



Gradient



Gradient

$$\begin{cases} D_h u(i,j) = u(i+1,j) - u(i,j) \\ D_v u(i,j) = u(i,j+1) - u(i,j) \\ \|Du\|(i,j) = \sqrt{(D_h u(i,j))^2 + (D_v u(i,j))^2} \end{cases}$$



Gradient

$$\begin{cases} D_h u(i,j) = u(i+1,j) - u(i,j) \\ D_v u(i,j) = u(i,j+1) - u(i,j) \\ \|Du\|(i,j) = \sqrt{(D_h u(i,j))^2 + (D_v u(i,j))^2} \end{cases}$$



Gradient

$$\begin{cases} D_h u(i, j) = u(i + 1, j) - u(i, j) \\ D_v u(i, j) = u(i, j + 1) - u(i, j) \\ \|Du\|(i, j) = \sqrt{(D_h u(i, j))^2 + (D_v u(i, j))^2} \end{cases}$$



Gradient + seuillage



Gradient + seuillage



Gradient + seuillage



Gradient + seuillage



Gradient + seuillage



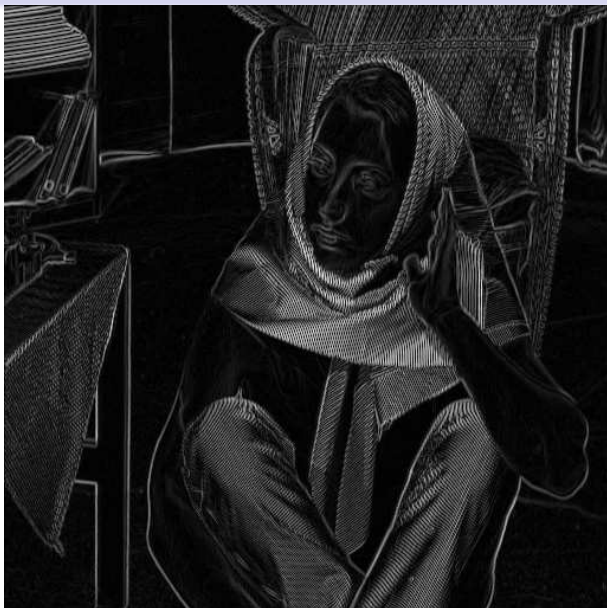
Gradient + seuillage



Gradient + seuillage



Gradient + seuillage



Gradient + seuillage



Gradient + seuillage

À quoi peut servir de connaître les contours d'objets?

- médical : volume d'une tumeur / d'un organe
- surveillance d'un hall de gare : bagage abandonné
- suivi d'objet dans un film (ballon de foot, balle de tennis)
- reconnaissance de formes
- comptage d'animaux, d'arbres, pour la préservation des espèces...

Question

Peut-on remonter à une image à partir de ses contours?

Contours → image

Input



Ground truth



Output



Input



Ground truth



Output



Contours → image



Contours → image

Input



Output



Input



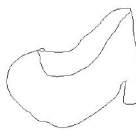
Output



Input



Output

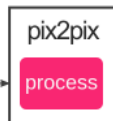


Contours → image

INPUT



OUTPUT



<https://affinelayer.com/pixsrv/>

- 1 Pleins de trucs utiles en traitement d'image
 - Mais qu'est-ce qu'une image ?
 - Modèle sténopé, perspective, vision binoculaire
 - Couleur et applications
- 2 Contours
- 3 La course des faussaires contre la police
- 4 Construire le cerveau des faussaires et de la police





Si la police fait bien son travail,
les faussaires ont intérêt à être bons.

Si la police se perfectionne, les faussaires
devront aussi toujours se perfectionner, jusqu'à
produire des objets indistinguables des vrais.



Si la police ne se perfectionne pas assez vite,
elle ne pourra jamais rattraper son retard.
Mais les faussaires resteront mauvais.

Conditional Generative Adversarial Networks (cGANs)

Base de données : plein de vrais couples (x, y)
 (x : contours, y : image associée).

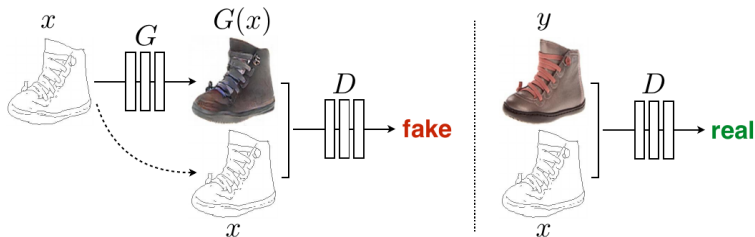
Générateurs

Les faussaires. À partir d'un x ils fabriquent des faux $y = \mathcal{G}(x, z)$
 (z aléatoire)

Discriminateurs

Les policiers. Ils essaient de distinguer les faux des vrais y et donnent une note entre 0 et 1.

$$\mathcal{D}(x, y) = \begin{cases} 1 & \text{si } y \text{ est un vrai,} \\ 0 & \text{si } y \text{ est un faux.} \end{cases}$$



$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_{x,y} [\log \mathcal{D}(x, y)] + \mathbb{E}_{x,z} [\log(1 - \mathcal{D}(x, \mathcal{G}(x, z)))]$$

Notations :

$\mathbb{E}_{x,y}$ moyenne sur plein de vraies paires de la base de données

$\mathbb{E}_{x,z}$ moyenne sur plein de fausses paires

$\log$ logarithme népérien

oo

oooooooo

oooooooo●oooooooooooooooo

oooooooo



$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_x [\log(\mathcal{D}(x))] + \mathbb{E}_y [\log(1 - \mathcal{D}(\mathcal{G}(y)))]$$

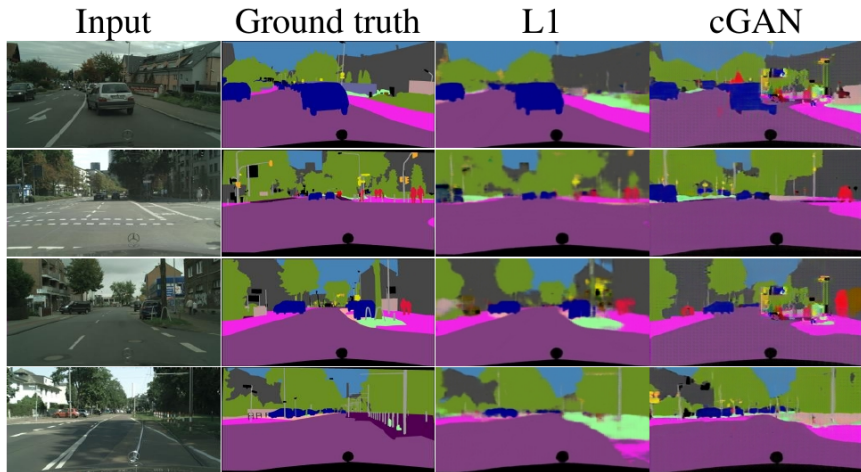
Aerial photo to map



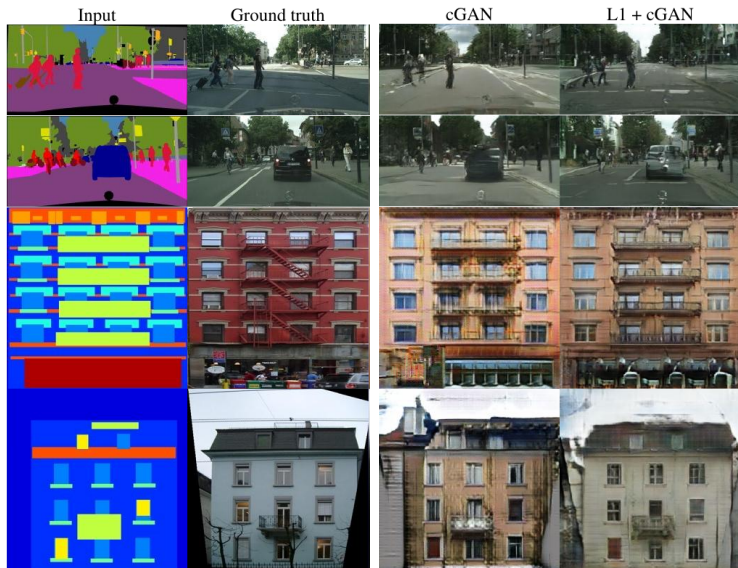
Application 3 : colorisation



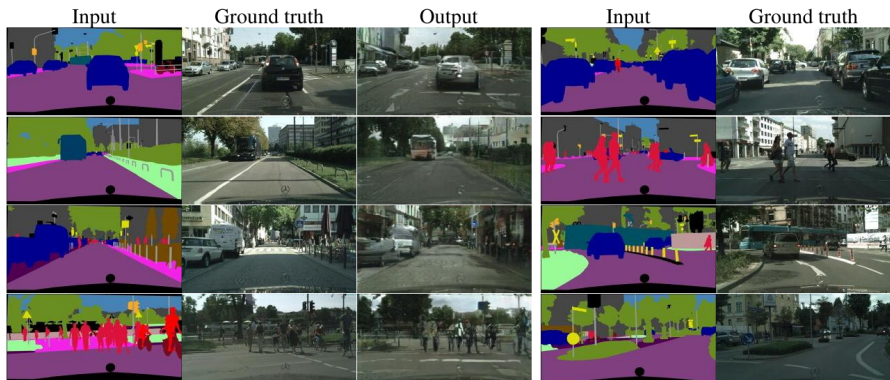
Application 4 : image $\rightarrow$ segmentation sémantique



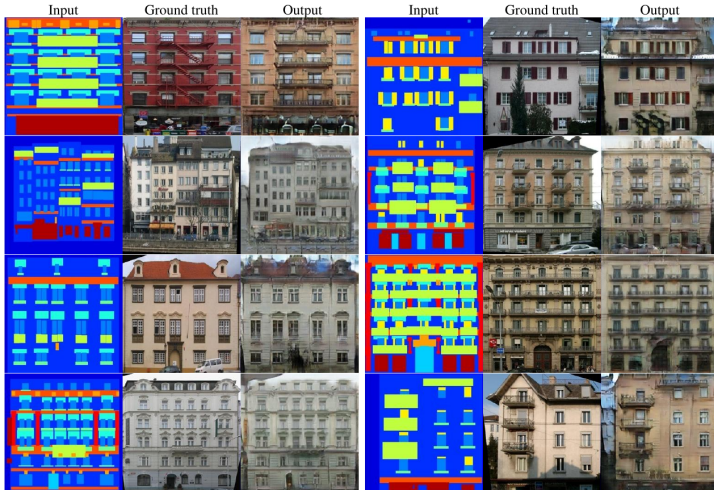
Application 5 : segmentation sémantique → image



Application 5 : segmentation sémantique → image



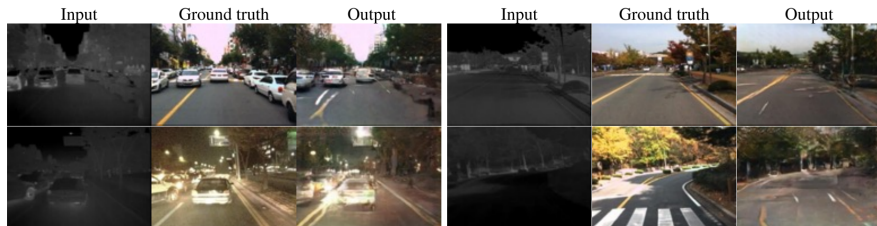
Application 5 : segmentation sémantique $\rightarrow$ image



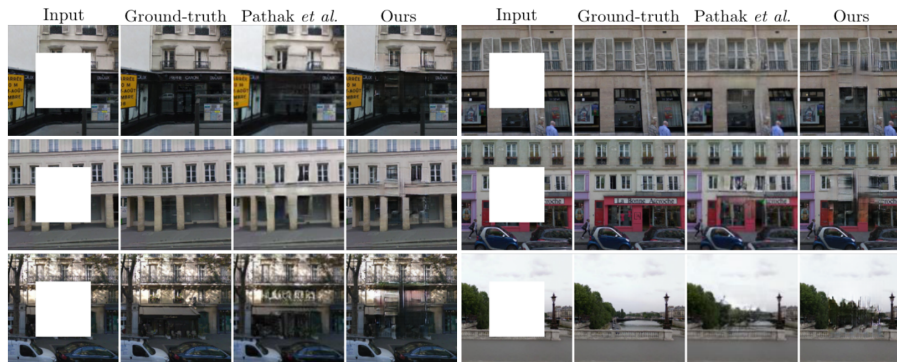
Application 6 : jour → nuit



Application 7 : image thermique → image couleur

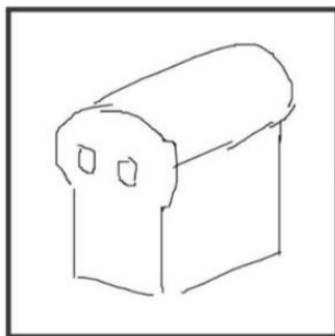


Application 8 : désocclusion

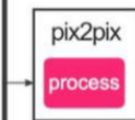


Application 9 : contours → chat

#edges2cats by Christopher Hesse



sketch by Ivy Tsai



Application 10 : artistique...



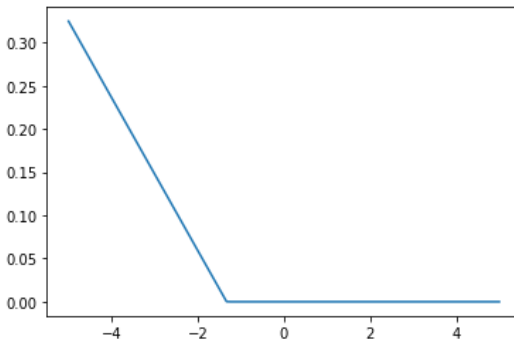
<https://vimeo.com/260612034>

- 1 Pleins de trucs utiles en traitement d'image
 - Mais qu'est-ce qu'une image ?
 - Modèle sténopé, perspective, vision binoculaire
 - Couleur et applications
- 2 Contours
- 3 La course des faussaires contre la police
- 4 Construire le cerveau des faussaires et de la police

A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

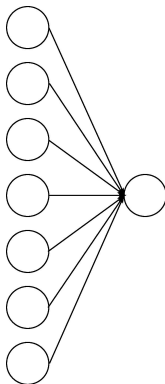
Un neurone

```
1 def affine(x,a,b):
2     return(a*x+b)
3
4 def neurone(inf_bound,sup_bound,N):
5     a = np.random.randn(1)
6     b = np.random.randn(1)
7     lx = np.linspace(inf_bound,sup_bound,num=N)
8     ly = [relu(affine(x,a,b)) for x in lx]
9     plt.plot(lx,ly)
10    plt.show()
11
12 neurone(-5,5,200)
```



A diagram of a star graph with 8 nodes. One central node is connected to 7 peripheral nodes arranged in a vertical line.

Fonction affine en dimension quelconque



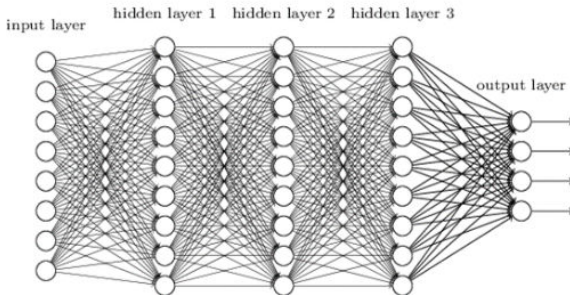
1 variable $y = ax + b$

2 variables $y = a_1x_1 + a_2x_2 + b$

3 variables $y = a_1x_1 + a_2x_2 + a_3x_3 + b$

n variables $y = a_1x_1 + \cdots + a_nx_n + b$

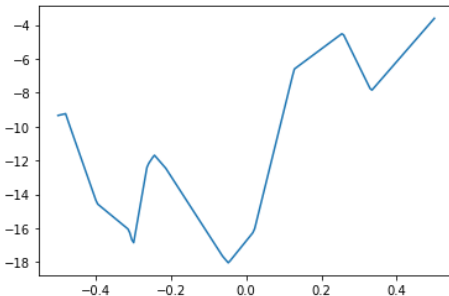
Deep neural network



```

1 def neurones_double(inf_bound,sup_bound,N,dim,depth):
2     a = np.random.randn(dim,dim,depth)
3     b = np.random.randn(dim,depth)
4     lx = np.linspace(inf_bound,sup_bound,num=N)
5     ly = np.zeros((dim,N))
6     lz = ly
7     ly_f = lx
8     for j in np.arange(dim) :
9         ly[j,:] = lx
10    for i in np.arange(depth) :
11        for j in np.arange(dim) :
12            for k in np.arange(N) :
13                lz[j,k] = relu(np.dot(a[:,j,i],ly[:,k])+b[j,i])
14        ly = lz
15    a_f = np.random.randn(dim)
16    b_f = np.random.randn(1)
17    for k in np.arange(N) :
18        ly_f[k] = np.dot(a_f,ly[:,k])+b_f
19
20    plt.plot(np.linspace(inf_bound,sup_bound,num=N), ly_f)
21    plt.show()
22
23 neurones_double(-0.5,0.5,200,9,4)

```





Conclusion

Messages à emporter

- les réseaux de neurones sont très puissants pour paramétrer les fonctions
- ça marche en n'importe quelle dimension (ouf ! on fait de l'image !)
- ça permet de construire des $\mathcal{G}$ et des $\mathcal{D}$

Difficultés occultées

- comment trouver les paramètres ? → optimisation
- plein de détails techniques
- il faut aider la police !