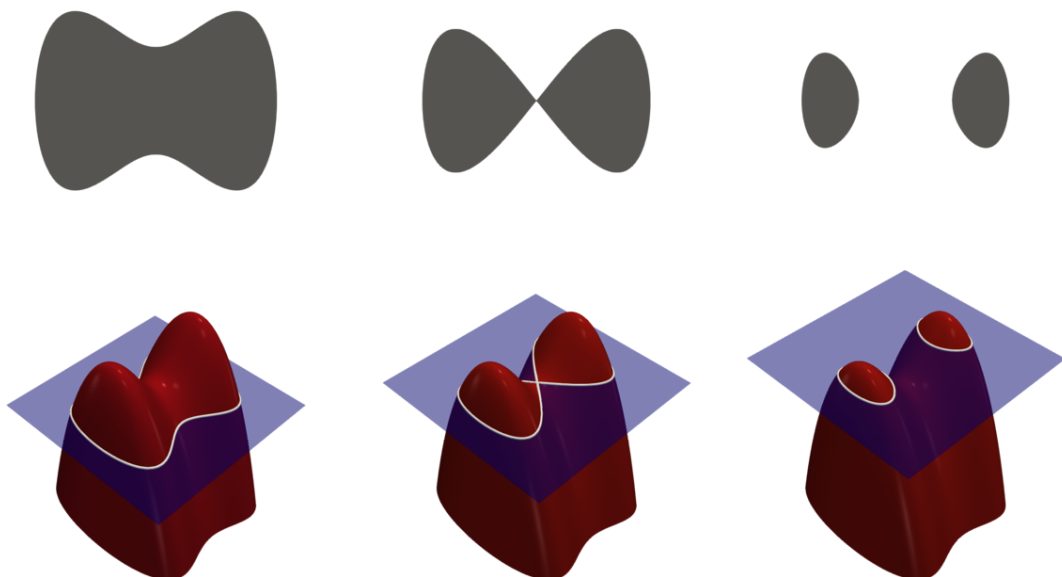


IMAGE PROCESSING

Active Contours using Level Sets



Rand ASSWAD
Génie Mathématique

A l'attention de :
M. Christian GOUT

2 mars 2020

Table des matières

1	Introduction	2
1.1	Exo 1	2
1.2	Exo 2	2
1.3	Exo 3	3
1.4	Exo 4	4
1.5	Exo 5	5
1.6	Exo 6	6
1.7	Exo 8	8
1.8	Exo 9	8

1 Introduction

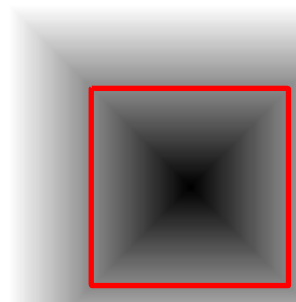
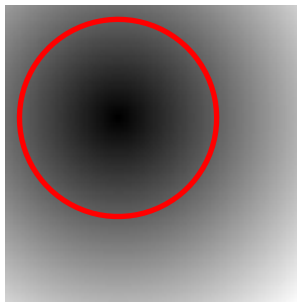
1.1 Exo 1

```
n = 200
Y, X = np.meshgrid(np.arange(1,n+1), np.arange(1,n+1))
r = n / 3

c = np.array([r,r]) + 10
phi1 = np.sqrt((X-c[0])**2 + (Y-c[1])**2) - r

c = n - c
phi2 = np.maximum(abs(X-c[0]), abs(Y-c[1])) - r

from nt_toolbox.plot_levelset import *
plt.figure()
plt.subplot(121)
plot_levelset(phi1)
plt.subplot(122)
plot_levelset(phi2)
plt.show()
```



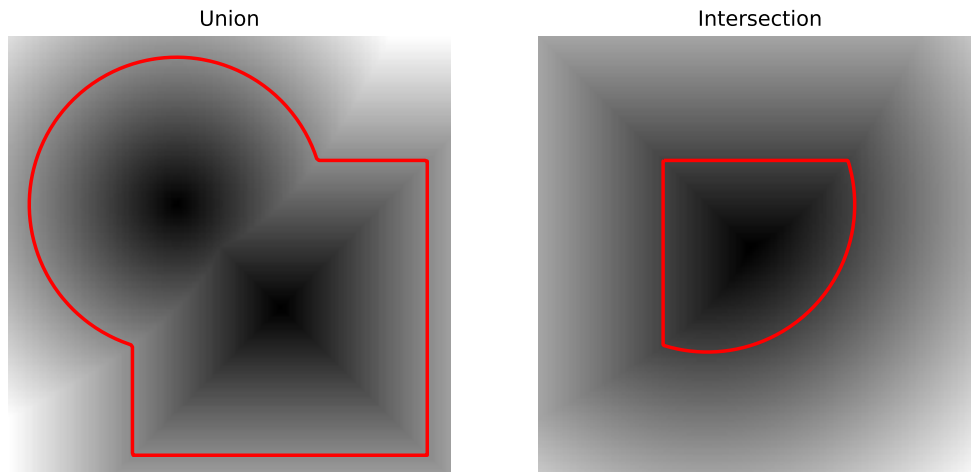
1.2 Exo 2

```
plt.figure(figsize = (10,5))
phi0 = np.minimum(phi1, phi2)

plt.subplot(1,2,1)
plot_levelset(phi0)
plt.title("Union")

plt.subplot(1,2,2)
plot_levelset(np.maximum(phi1, phi2))
plt.title("Intersection")

plt.show()
```



1.3 Exo 3

```

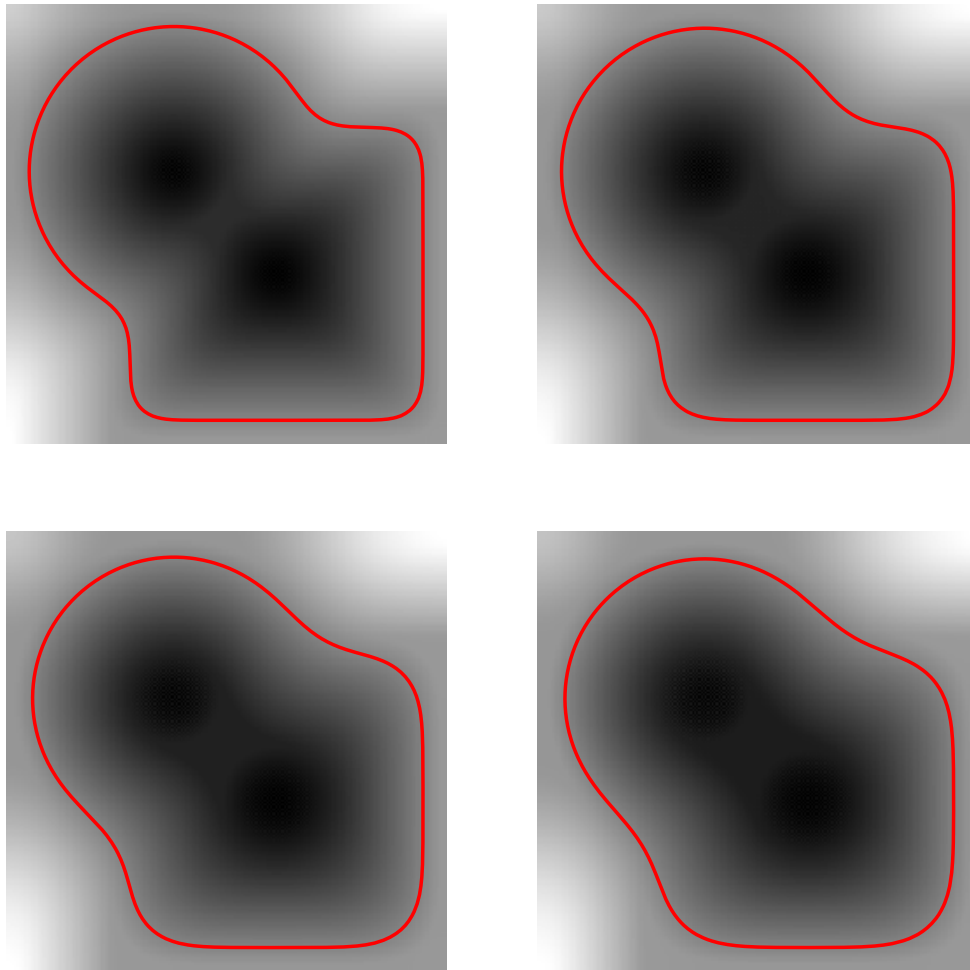
from nt_toolbox.grad import *
from nt_toolbox.div import *

Tmax = 200
tau = .5
niter = int(Tmax/tau)

plt.figure(figsize=(10,10))
phi = np.copy(phi0) #initialization
eps = np.finfo(float).eps
k = 0

for i in range(1,niter+1):
    g0 = grad(phi, order=2)
    d = np.maximum(eps*np.ones([n,n]), np.sqrt(np.sum(g0**2, 2)))
    g = g0/np.repeat(d[:, :, np.newaxis], 2, 2)
    K = d*div(g[:, :, 0], g[:, :, 1], order=2)
    phi = phi + tau*K
    if i % int(niter/4.) == 0:
        k = k + 1
        plt.subplot(2, 2, k)
        plot_levelset(phi)

```



1.4 Exo 4

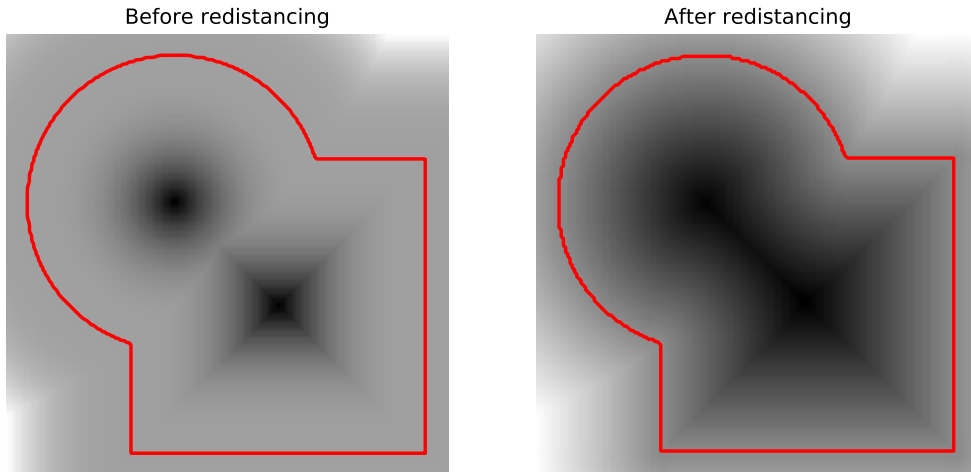
```
# problem cause
phi = phi0**3
from nt_toolbox.perform_redistancing import *
phi1 = perform_redistancing(phi0)

plt.figure(figsize=(10,5))

plt.subplot(1,2,1)
plot_levelset(phi)
plt.title("Before redistancing")

plt.subplot(1,2,2)
plot_levelset(phi1)
plt.title("After redistancing")

plt.show()
```



```

n = 200
f0 = rescale(load_image("nt_toolbox/data/cortex.bmp", n))
g = grad(f0, order=2)
d0 = np.sqrt(np.sum(g**2, 2))
a = 5
from nt_toolbox.perform_blurring import *
#d = perform_blurring(d0, np.asarray([a]), bound="per")
d = d0
epsilon = 1e-1
W = 1./(epsilon + d)
W = rescale(-d, 0.1, 1)

plt.figure(figsize=(10,5))
imageplot(f0, "Image to segment", [1,2,1])
imageplot(W, "Weight", [1,2,2])

Y,X = np.meshgrid(np.arange(1,n+1), np.arange(1,n+1))
r = n/3
c = np.asarray([n,n])/2
phi0 = np.maximum(abs(X-c[0]), abs(Y-c[1])) - r
plt.figure(figsize=(5,5))
plot_levelset(phi0, 0, f0)

```

1.5 Exo 5

```

tau = .4
Tmax = 1500
niter = int(Tmax/tau)
phi = np.copy(phi0)
gW = grad(W, order=2)

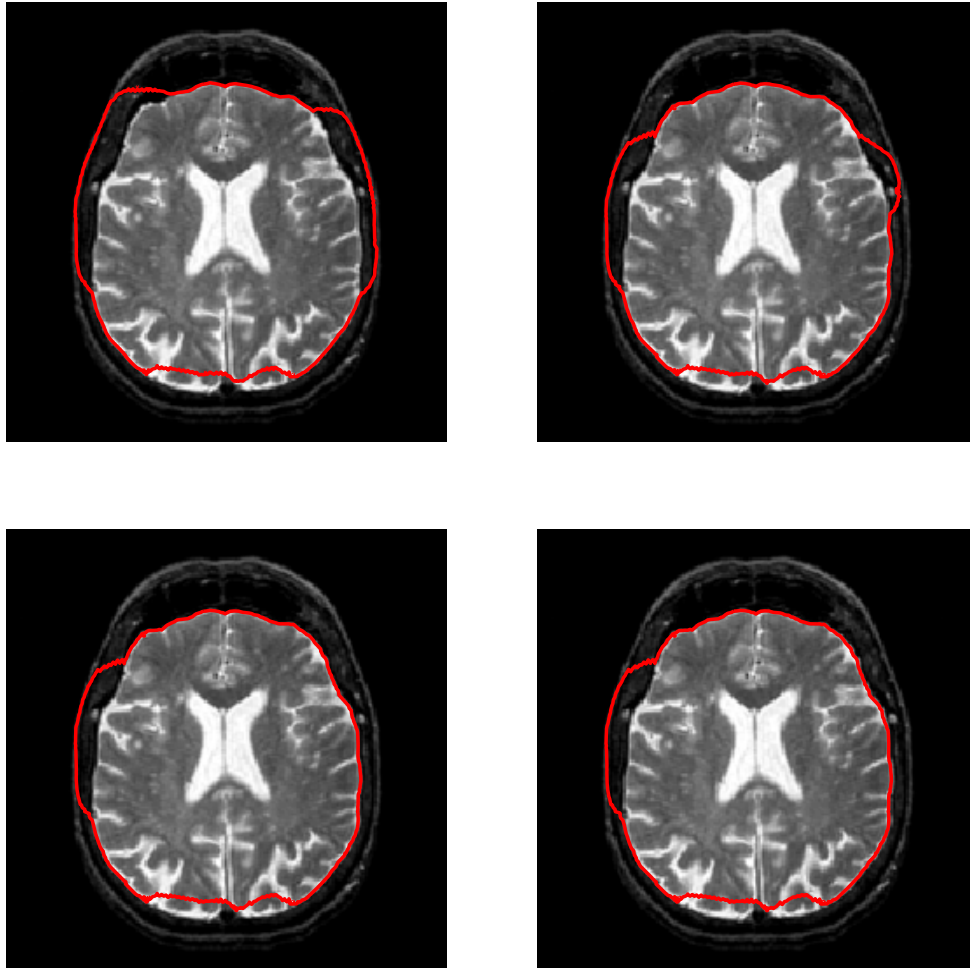
gD = grad(phi, order=2)
d = np.maximum(eps*np.ones([n,n]), np.sqrt(np.sum(gD**2, 2)))
g = gD/np.repeat(d[:, :, np.newaxis], 2, 2)
G = - W*d*div(g[:, :, 0], g[:, :, 1], order=2) - np.sum(gW*gD, 2)

```

1.6 Exo 6

```
plt.figure(figsize=(10,10))
phi = np.copy(phi0)
k = 0
gW = grad(W, order=2)

for i in range(1,niter+1):
    gD = grad(phi, order=2)
    d = np.maximum(eps*np.ones([n,n]), np.sqrt(np.sum(gD**2, 2)))
    g = gD/np.repeat(d[:, :, np.newaxis], 2, 2)
    G = W*d*div(g[:, :, 0], g[:, :, 1], order=2) + np.sum(gW*gD, 2)
    phi = phi + tau*G
    if i % 30 == 0:
        phi = perform_redistancing(phi)
    if i % int(niter/4.) == 0:
        k = k + 1
        plt.subplot(2, 2, k)
        plot_levelset(phi, 0, f0)
```



Exo 7

```
plt.figure(figsize=(10,5))
Y,X = np.meshgrid(np.arange(1,n+1), np.arange(1,n+1))
k = 4 #number of circles
r = .3*n/k
phi0 = np.zeros([n,n])+np.float("inf")

for i in range(1,k+1):
    for j in range(1,k+1):
        c = (np.asarray([i,j]) - 1)*(n/k) + (n/k)*.5
        phi0 = np.minimum(phi0,np.sqrt(abs(X-c[0])**2 + abs(Y-c[1])**2) - r)

plt.subplot(1,2,1)
plot_levelset(phi0,0)
plt.subplot(1,2,2)
plot_levelset(phi0, 0, f0)
```


1.7 Exo 8

```

lambd = 2
c1 = .7
c2 = 0
tau = .5

Tmax = 100
niter = int(Tmax/ tau)
phi = np.copy(phi0)

gD = grad(phi, order=2)
d = np.maximum(eps*np.ones([n,n]), np.sqrt(np.sum(gD**2, 2)))
g = gD/np.repeat(d[:, :, np.newaxis], 2, 2)
G = d*div(g[:, :, 0], g[:, :, 1], order=2) - lambd*(f0-c1)**2 + lambd*(f0-c2)**2

```

1.8 Exo 9

```

plt.figure(figsize=(10,10))
phi = np.copy(phi0)
k = 0

for i in range(1,niter+1):
    gD = grad(phi, order=2)
    d = np.maximum(eps*np.ones([n,n]), np.sqrt(np.sum(gD**2, 2)))
    g = gD/np.repeat(d[:, :, np.newaxis], 2, 2)
    G = d*div(g[:, :, 0], g[:, :, 1], order=2) - lambd*(f0-c1)**2 + lambd*(f0-c2)**2
    phi = phi + tau*G
    if i % 30 == 0:
        phi = perform_redistancing(phi)
    if i % int(niter/4.) == 0:
        k = k + 1
        plt.subplot(2, 2, k)
        plot_levelset(phi, 0, f0)

```

