Exercici 5 de Laboratori

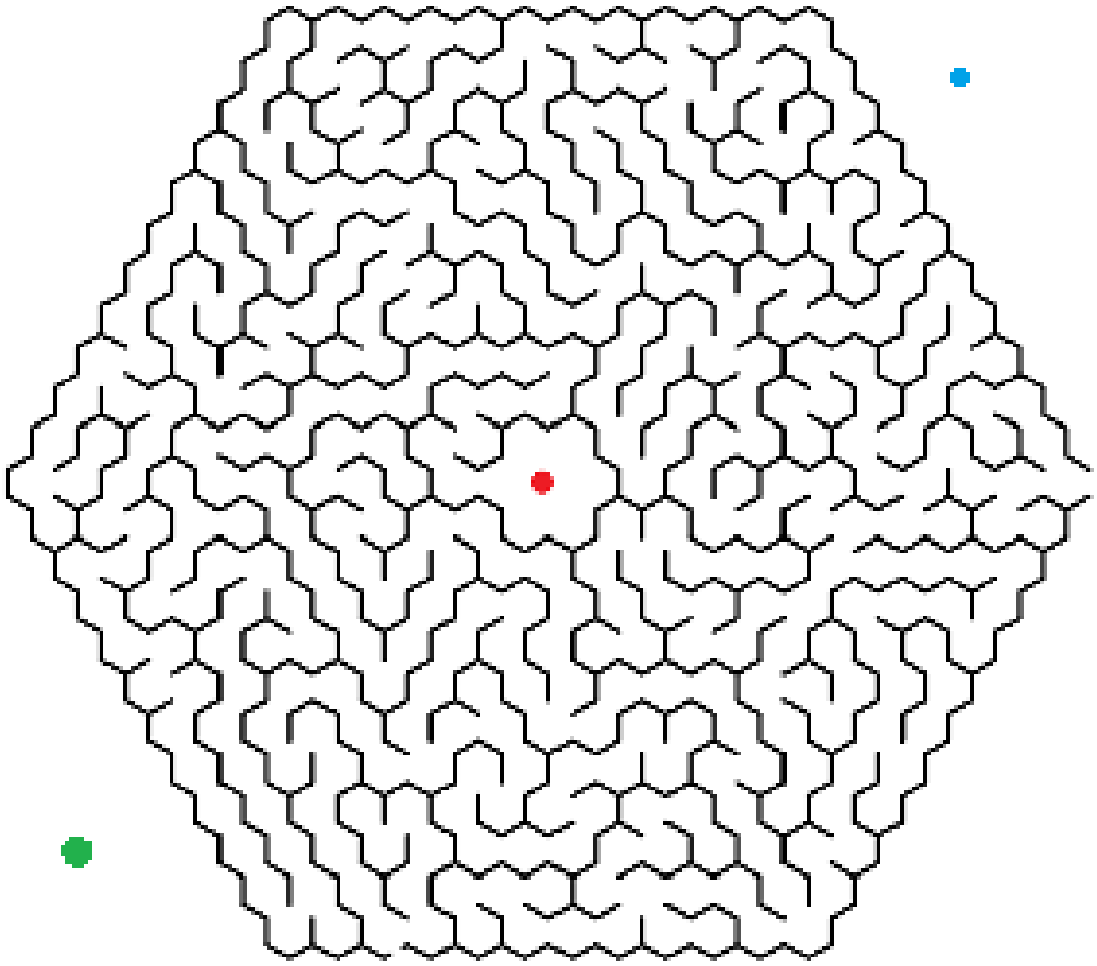
Cerca de camins

(Per Francesc Forn, Raúl Lumbreras i Martí Vall)

Es vol tenir una útil aplicació de visió per computador que trobi el camí més curt entre dos punts. Amb aquesta aplicació es vol discernir quin de dos punt és més proper al centre. Els dos punts a considerar estan indicats en verd i blau respectivament, i el centre està indicat en vermell, com es pot observar en la figura següent. La tasca consisteix en trobar el camí més curt, considerant que no es pot traspassar els murs del laberint i que l'objecte que es trasllada és puntual (de mida 1 píxel).

Primer de tot llegim la imatge i aïllem els murs del laberint i els tres punts:

```
I = imread("Laberint.png");
imshow(I);
```

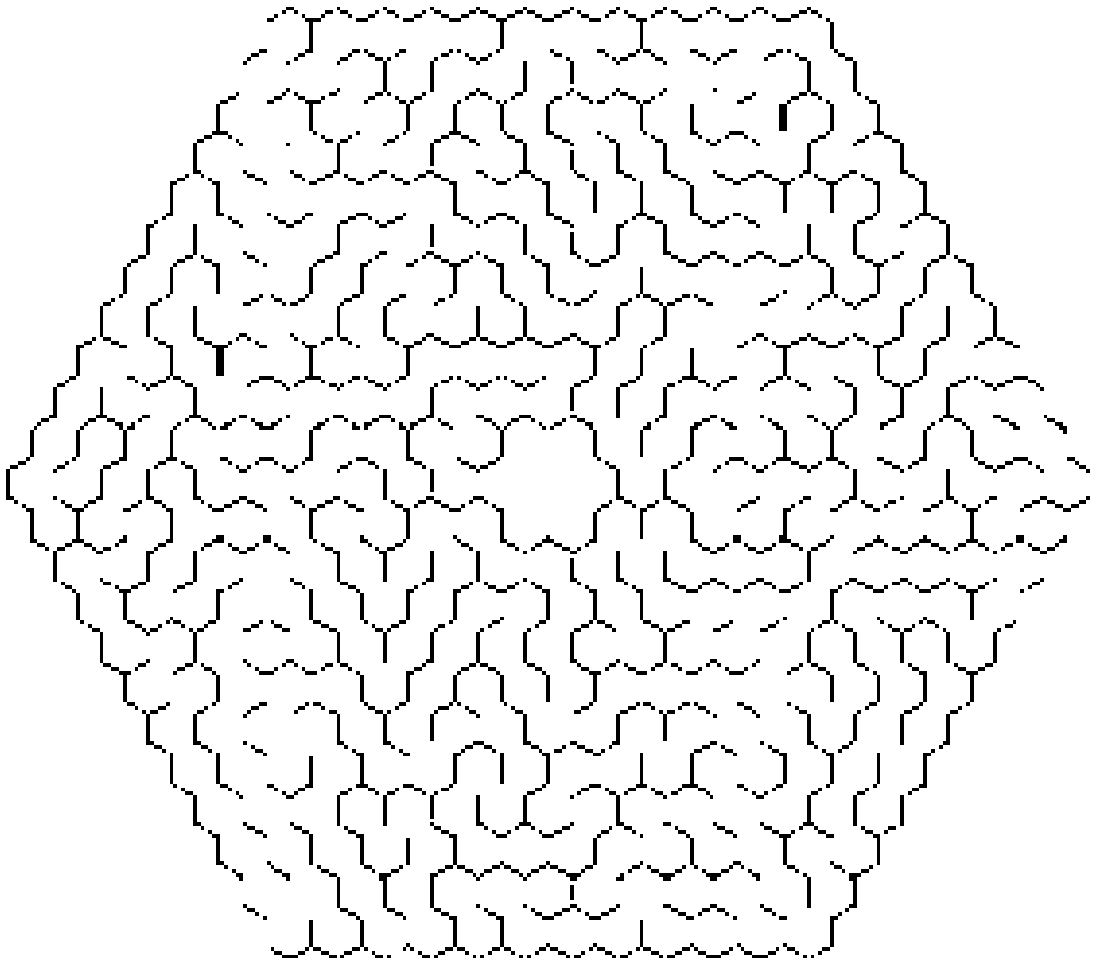


```
red = I(:,:,1) > 128;
```

```
green = I(:,:,2) > 128;
blue = I(:,:,3) > 128;

seed = red.*(1-green).*(1-blue);
greenDot = (1-red).*green.*(1-blue);
blueDot = (1-red).*green.*blue;

lab = I(:,:,1) < 32 & I(:,:,2) < 32 & I(:,:,1) < 32;
lab = 1-lab;
imshow(lab)
```



Emprem un element estructurant, SE, del tipus disc de radi 1 (4-way) per fer una reconstucció, que consisteix en una seqüència de dilatacions iterativa partint desde el punt vermell del centre del laberint respectant els murs. En cada iteració dilatem el resultat de l'anterior, fem una AND amb la imatge del laberint per delimitar els murs i ho sumem al resultat (dist), el que farà que les iteracions del principi siguin més blanques i les del final més fosques al normalitzar.

Aquest procediment el repetim fins que la zona dilatada es trobi amb un dels punts exteriors (blau o verd), fenòmen que es pot detectar fent la operació lògica AND i mirant si algun dels píxels resultants és no-nul (amb sum).

```
SE = strel("disk",1);
s1 = seed;
i = 0;
sumG = 0;
sumB = 0;
dist = im2double(seed);

while sumG + sumB == 0
    s1 = imdilate(s1,SE) & lab;
    sumG = sum(sum(s1 & greenDot));
    sumB = sum(sum(s1 & blueDot));
    i = i+1;
    dist = dist + s1;
end
fprintf('%d píxels al camí.',i);
```

```
1441 píxels al camí.
```

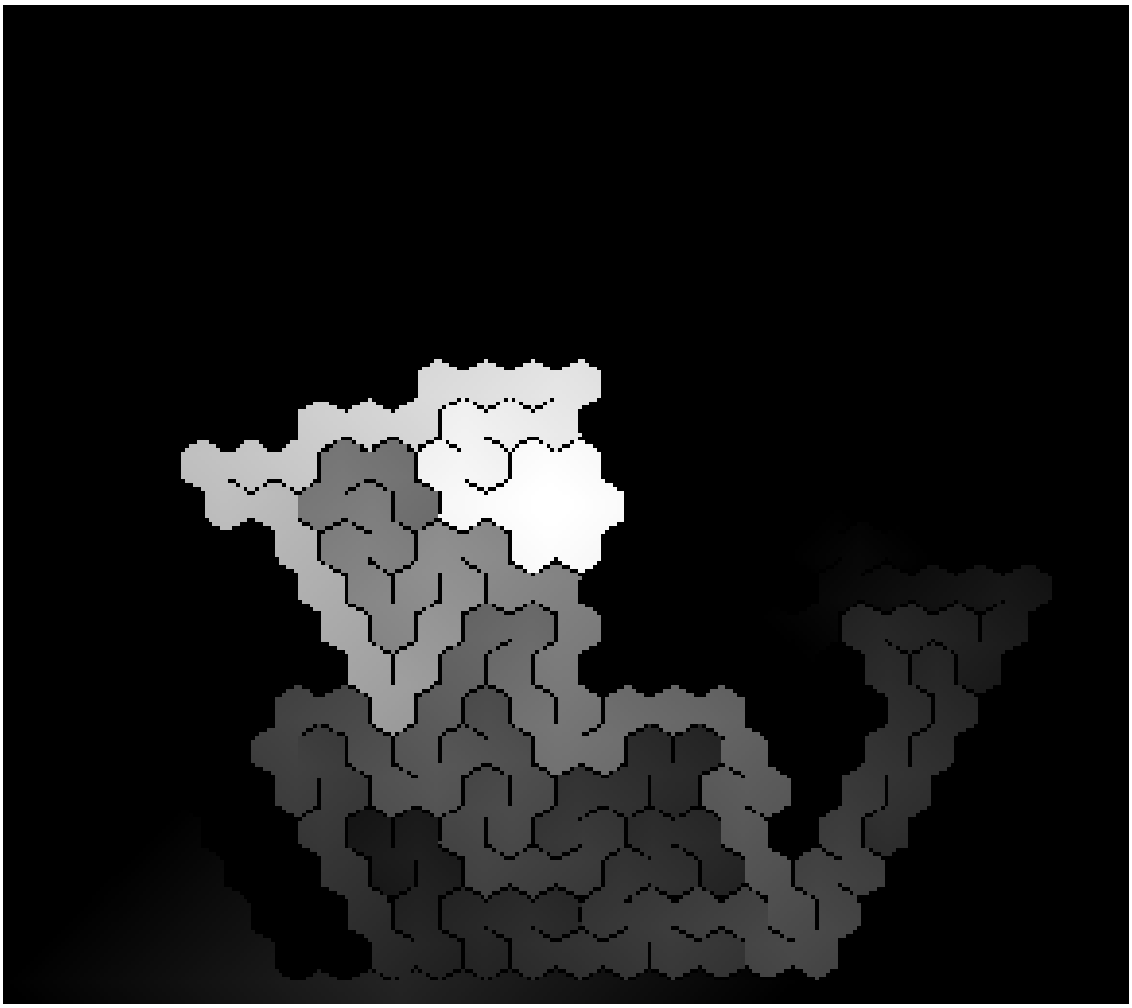
```
fprintf('%d píxels al punt verd.',sumG);
```

```
3 píxels al punt verd.
```

```
fprintf('%d píxels al punt blau.',sumB);
```

```
0 píxels al punt blau.
```

```
imshow(dist,[]);
```



Veient això, concluïm que es troba primer amb el punt verd.

Per tal de poder representar el camí, repetim aquest procés ara des de el punt verd, que ja sabem que és el més pròxim. En aquest bucle guardarem la distància al punt inicial per poder després trobar el camí més curt.

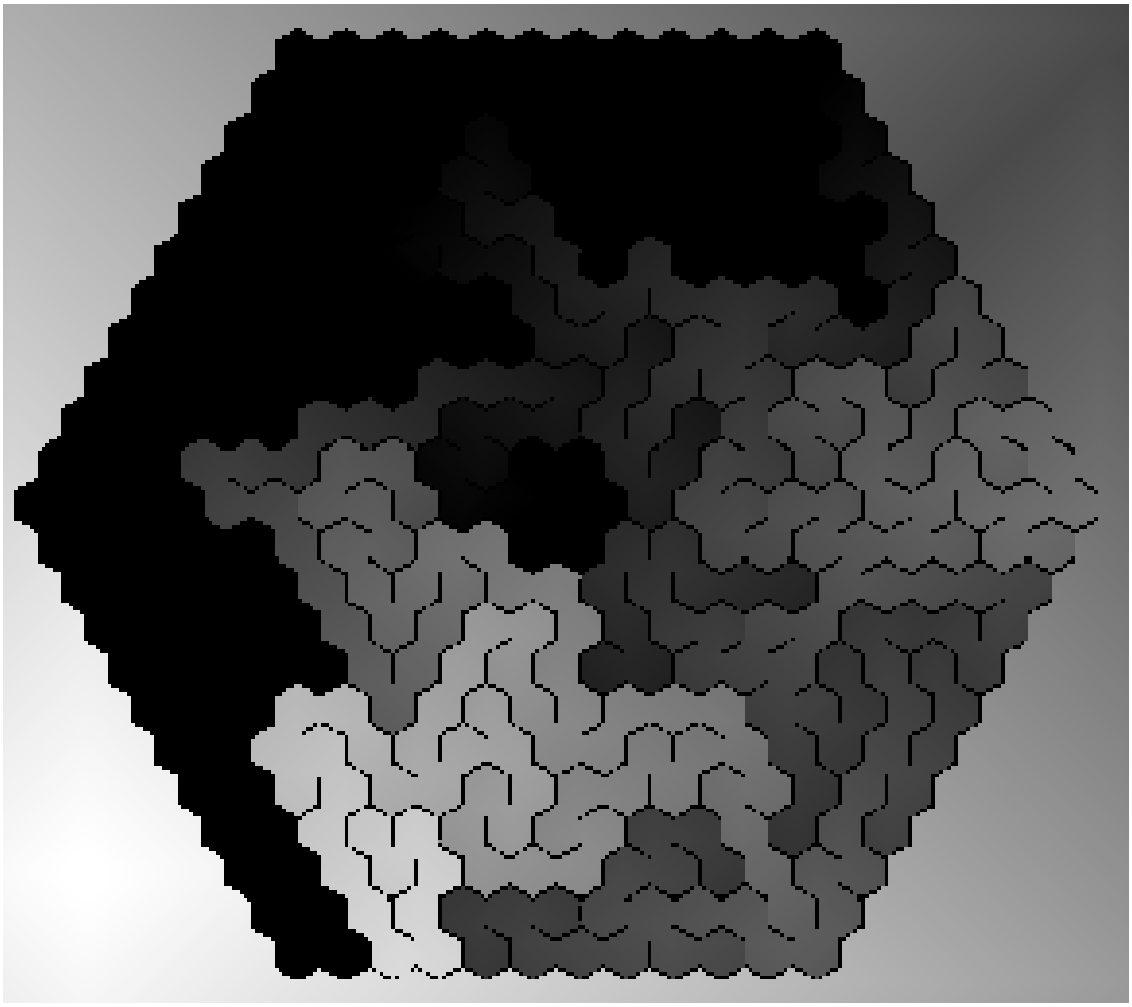
```
s2 = greenDot;
sumR = 0;
i = 0;
dist2 = im2double(greenDot);

while sumR == 0
    s2 = imdilate(s2,SE) & lab;
    sumR = sum(sum(s2&seed));
    i = i+1;
    dist2 = dist2 + s2;
end
```

```
fprintf('%d píxels en el camí.',i);
```

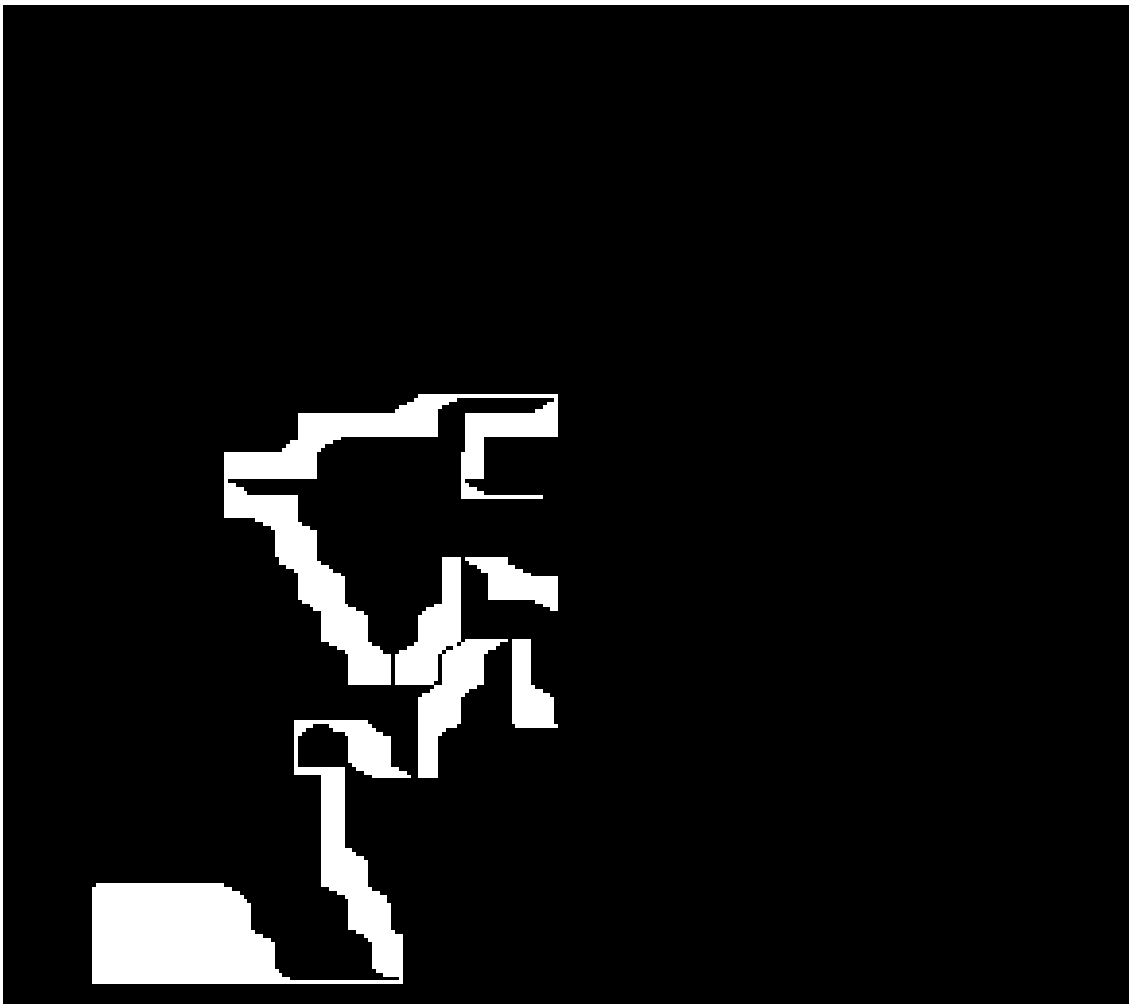
```
1441 píxels en el camí.
```

```
imshow(dist2,[]);
```



Tenint les distàncies obtingudes al fer el recorregut desde el punt vermell i desde el punt verd, sumem les distàncies en una nova matriu que contindrà el camí més curt. Aquest camí es trobarà entre aquells píxels que tinguin valor màxim, corresponent a la suma dels iterands finals dels bucles anteriors, que corresponen a quan les dilatacions es trobaven per fi amb el seu obejctiu. Definim per tant, una matriu on només hi guardem els píxels amb aquest valor exacte, ***distmax***:

```
m = max(max(dist+dist2));
distmax = dist+dist2 == m;
imshow(distmax,[]);
```



Seguidament, emprant la funció ***bwmorph*** amb paràmetre ***skel***, podem reduïr aquesta regió a un esquelet d'un sol píxel:

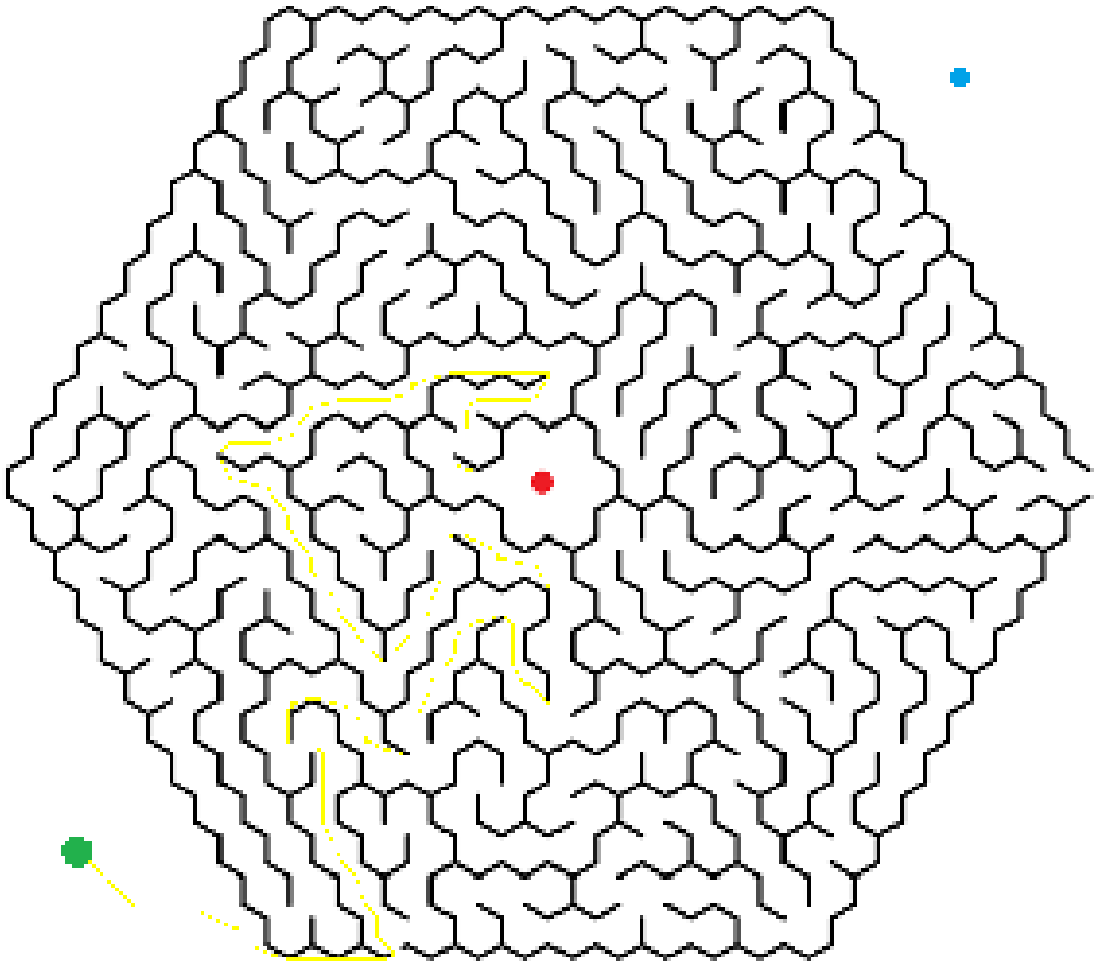
```
sk = bwmorph(distmax,"skel",inf);
imshow(sk);
```

Eliminem les branques residuals amb una altre aplicació de ***bwmorph***, marcant-hi el punts d'inici i fi amb una OR lògica:

```
sk = bwmorph(sk|seed|greenDot,"spur",inf);
imshow(sk);
```

Per últim, eliminem les marques corresponents al punt blau i punt verd afegits per tal de tenir només camí, i superposem el camí trobat a la imatge inicial per tal de poder representar-lo sobre el laberint inicial:

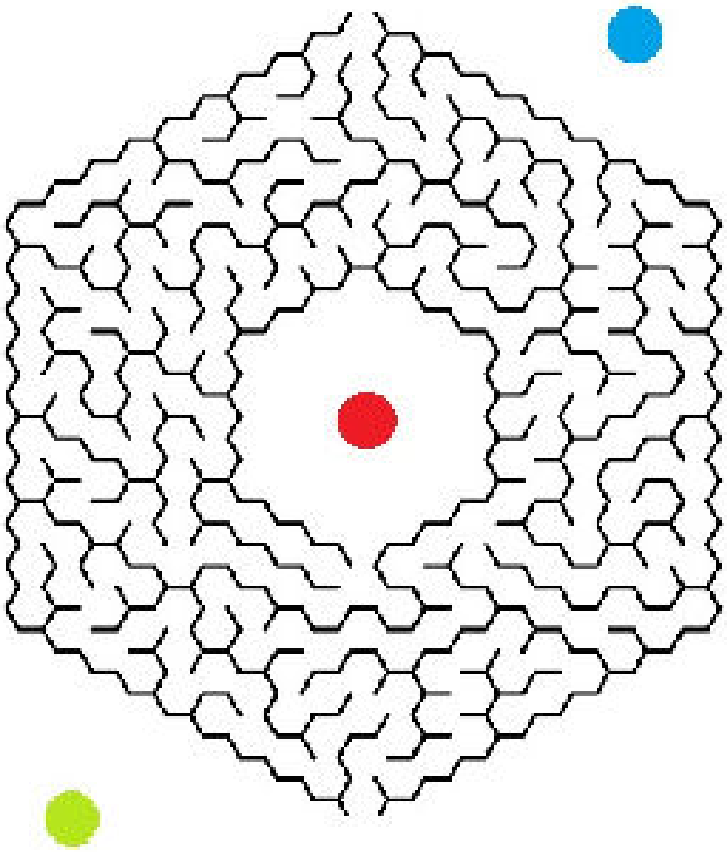
```
sk = sk-seed-greenDot;
imFinal = imoverlay(I,sk,"yellow");
imshow(imFinal);
```



Amb aquests passos hem estat capaços de trobar el camí més curt al centre d'un laberint amb dues entrades. La distància exacte en píxels pot variar segons l'element estructural que s'hagi emprat. En aquest cas hem optat per un del tipus disc.

Per tal de verificar que aquest procediment en efecte funciona, el repetim amb un altre laberint:

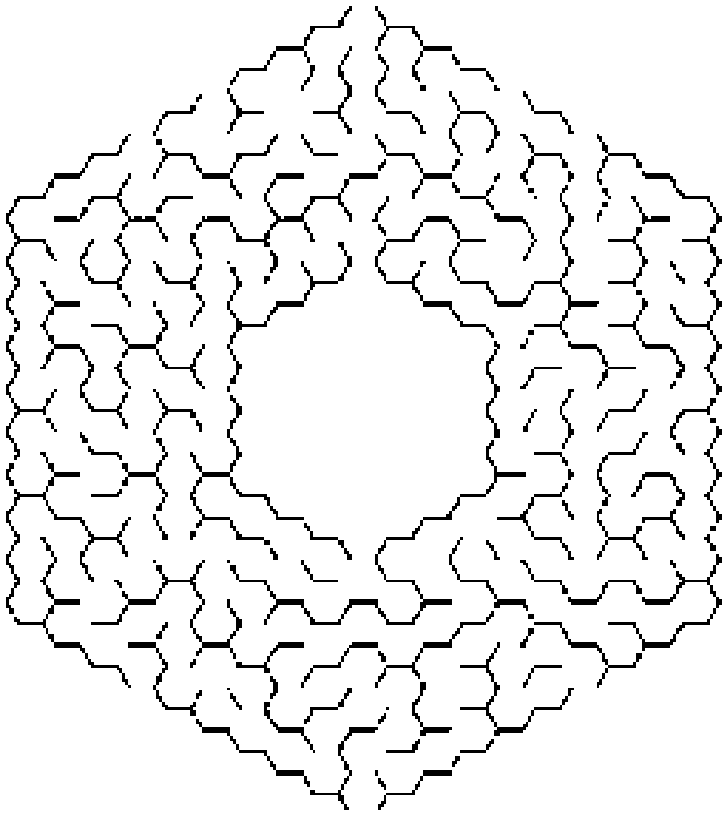
```
I = imread("Laberint2.jpg");
imshow(I);
```



```
red = I(:,:,1) > 128;
green = I(:,:,2) > 128;
blue = I(:,:,3) > 128;

seed = red.*(1-green).*(1-blue);
greenDot = (1-red).*green.*(1-blue);
blueDot = (1-red).*green.*blue;

lab = I(:,:,1) < 32 & I(:,:,2) < 32 & I(:,:,1) < 32;
lab = 1-lab;
imshow(lab)
```



```
SE = strel("disk",1);
s1 = seed;
i = 0;
sumG = 0;
sumB = 0;
dist = im2double(seed);

while sumG + sumB == 0
    s1 = imdilate(s1,SE) & lab;
    sumG = sum(sum(s1 & greenDot));
    sumB = sum(sum(s1 & blueDot));
    i = i+1;
    dist = dist + s1;
end
fprintf('%d píxels al camí.',i);
```

```
928 píxels al camí.
```

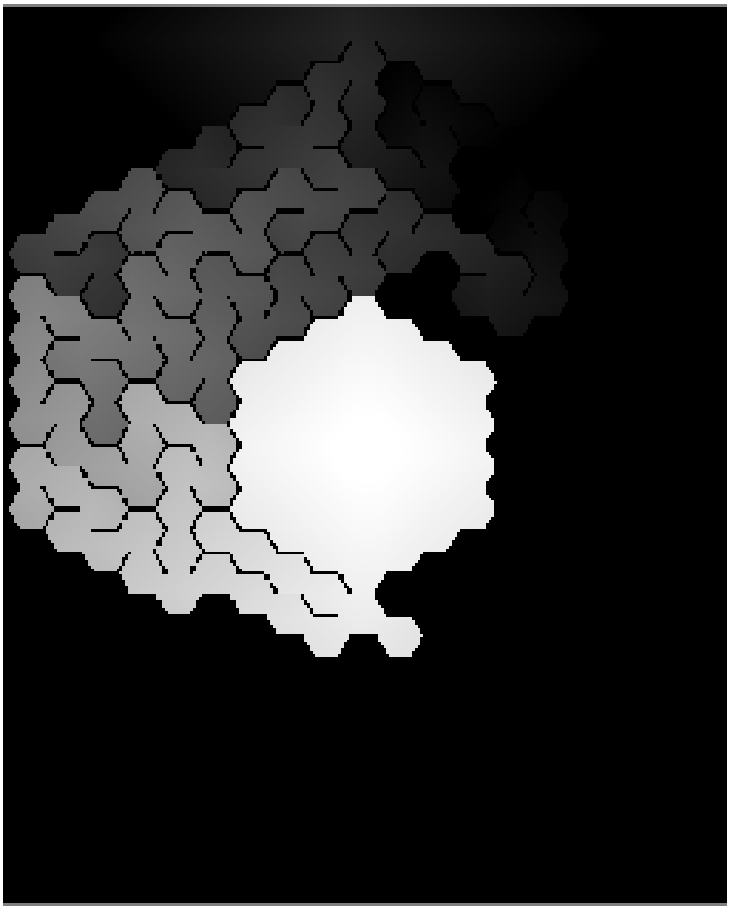
```
fprintf('%d píxels al punt verd.',sumG);
```

```
0 píxels al punt verd.
```

```
fprintf('%d píxels al punt blau.',sumB);
```

```
5 píxels al punt blau.
```

```
imshow(dist,[]);
```



En aquest cas és el punt blau el que és més pròxim al centre del laberint.

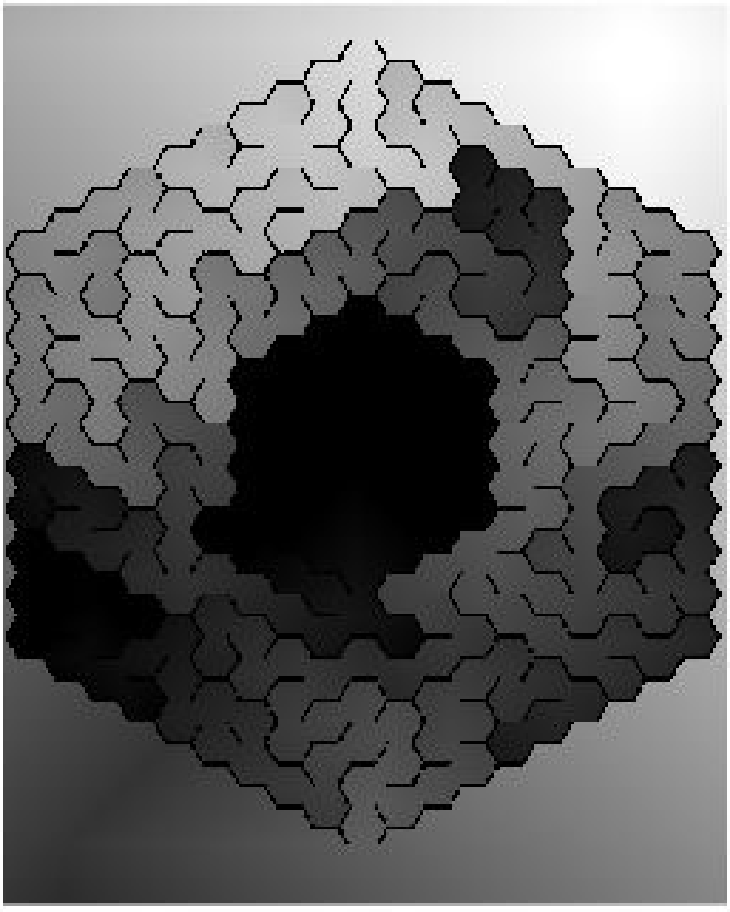
```
s2 = blueDot;
sumR = 0;
i = 0;
dist2 = im2double(blueDot);

while sumR == 0
    s2 = imdilate(s2,SE) & lab;
    sumR = sum(sum(s2&seed));
    i = i+1;
    dist2 = dist2 + s2;
end
```

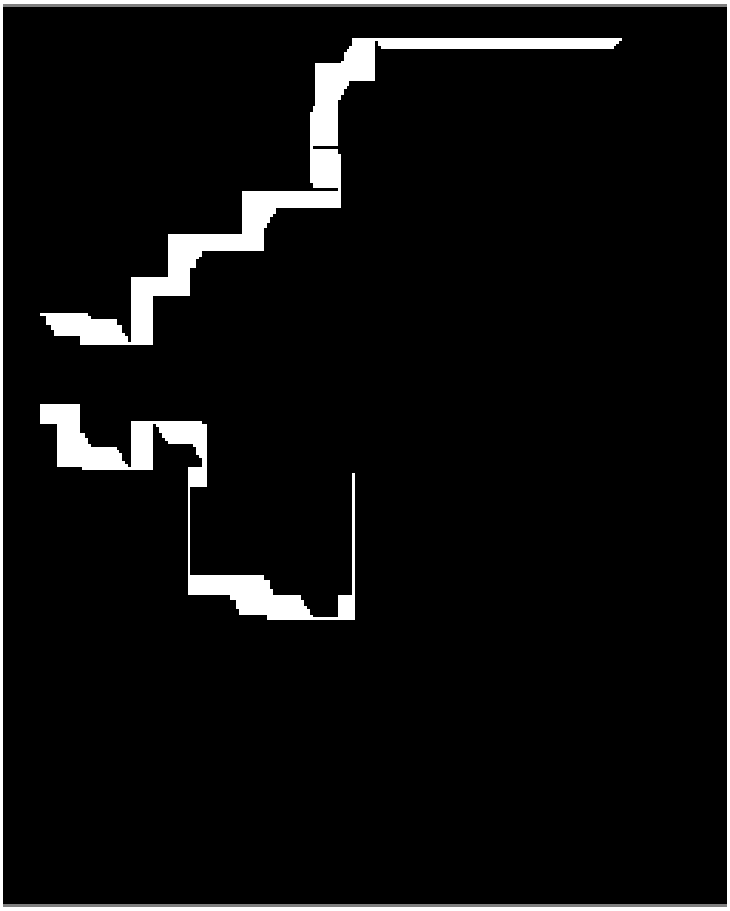
```
fprintf('%d píxels en el camí.',i);
```

```
928 píxels en el camí.
```

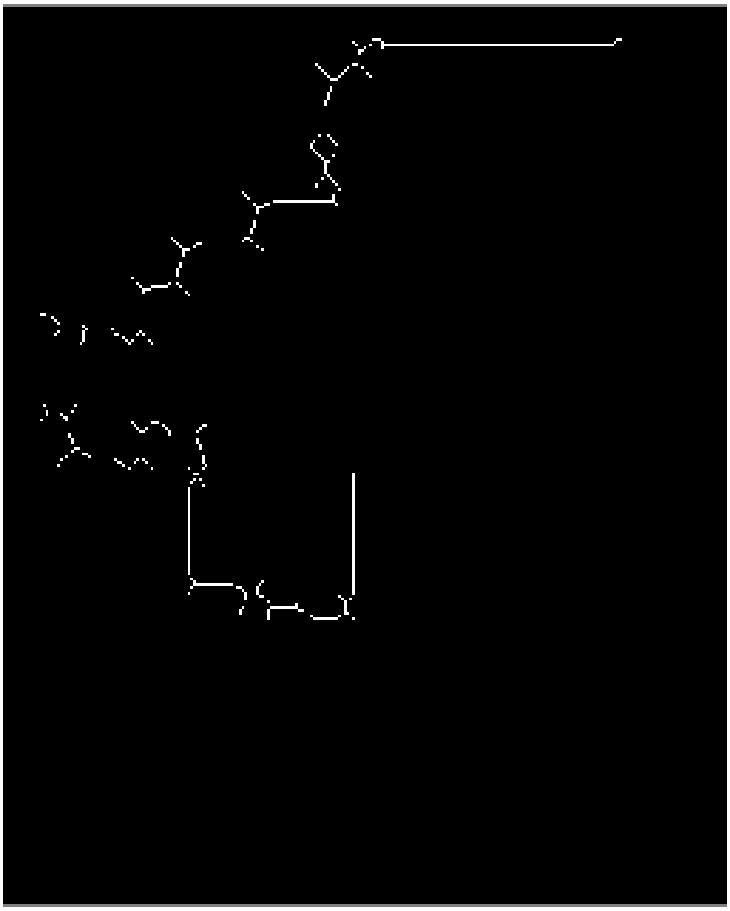
```
imshow(dist2,[]);
```



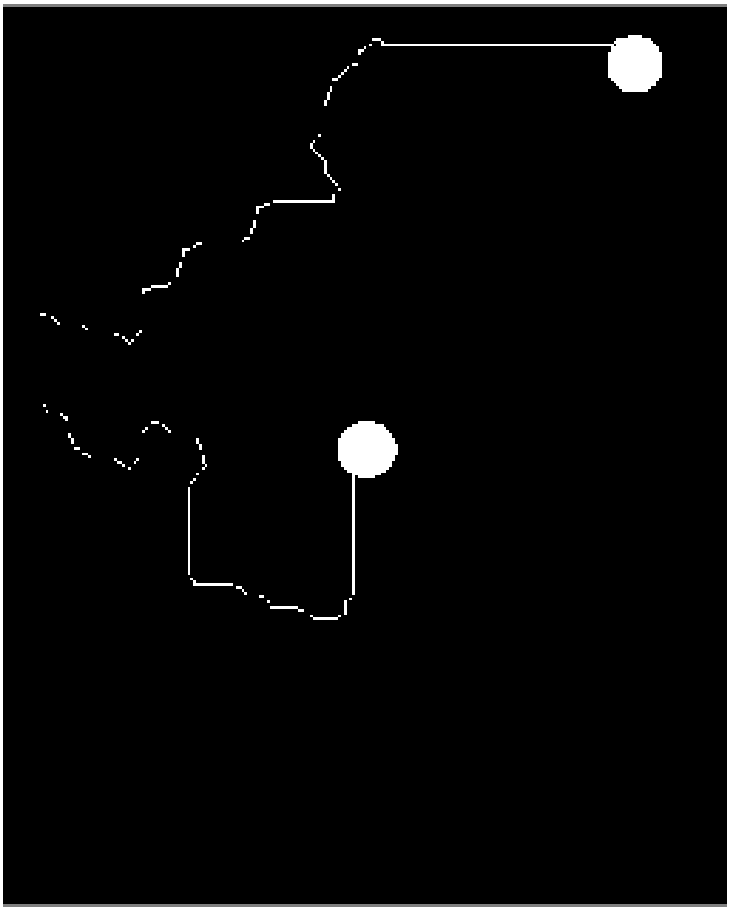
```
m = max(max(dist+dist2));
distmax = dist+dist2 == m;
imshow(distmax,[]);
```



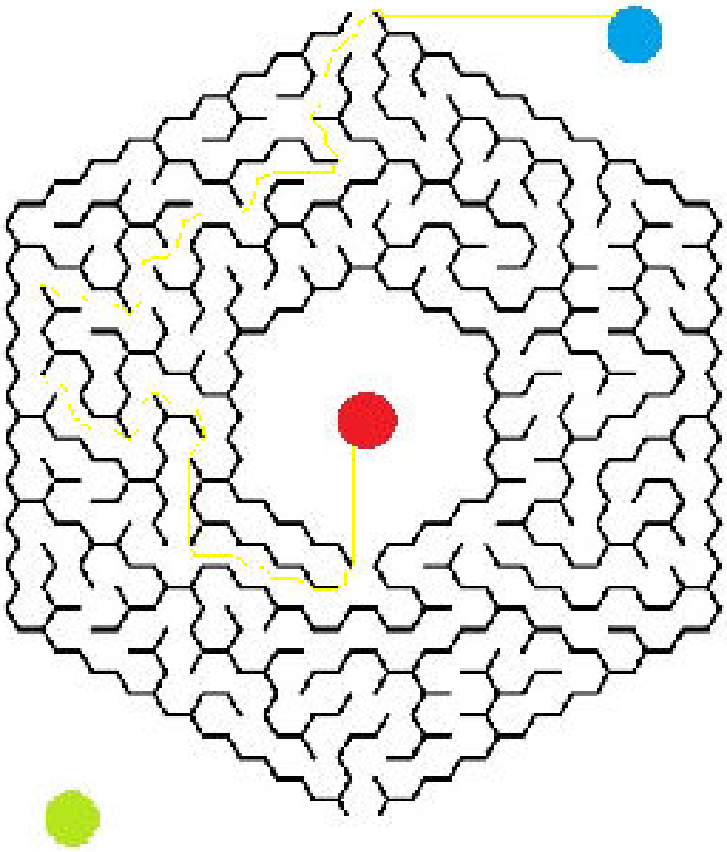
```
sk = bwmorph(distmax,"skel",inf);
imshow(sk);
```



```
sk = bwmorph(sk|seed|blueDot,"spur",inf);
imshow(sk);
```



```
sk = sk-seed-blueDot;
imFinal = imoverlay(I,sk,"yellow");
imshow(imFinal);
```



De nou, s'ha marcat amb groc el camí que uneix el punt més pròxim amb el centre. Queda vist, doncs, que l'algoritme sembla funcionar.