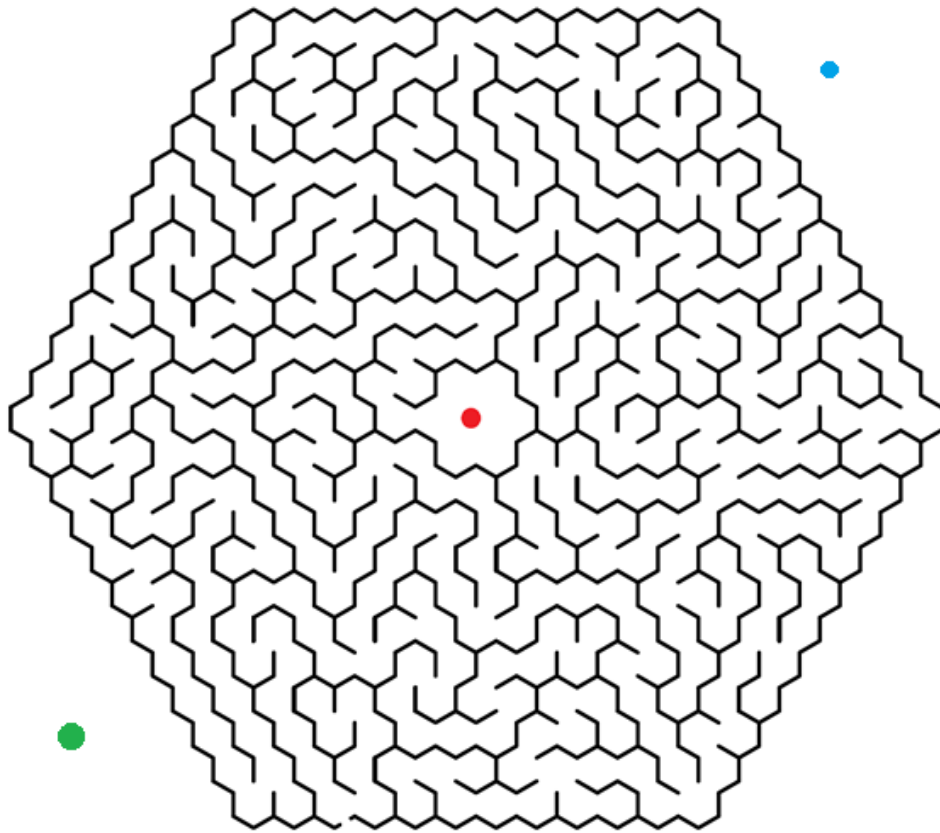


VC: Informe de Laboratori 5

Processat morfològic d'imatges II



Pere Ginebra Solanellas

22/3/2021 – Q2 Curs 2020-21

Visió per Computador, FIB UPC

1. Introducció

En aquesta sessió aplicarem els coneixements adquirits del processat morfològic d'imatges per resoldre un problema. En concret buscarem la distància i el camí més curt entre dos parells de punts en un laberint.

2. Exercici

Per començar importem la imatge del laberint i a partir de la seva representació en HSV extraïem vàries imatges binàries que ens seran útils pels següents apartats. Aquestes són la imatge *maze* que representa els passadissos del laberint i les imatges *r/g/bdot* que representen els 3 punts de color:

```
img = imread('Laberint.png');

imgHSV = rgb2hsv(img);
maze = imgHSV(:,:,3) > 0.6;
rdot = (imgHSV(:,:,1) < 0.1 | imgHSV(:,:,1) > 0.8) & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;
bdot = imgHSV(:,:,1) > 0.5 & imgHSV(:,:,1) < 0.8 & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;
gdot = imgHSV(:,:,1) < 0.4 & imgHSV(:,:,1) > 0.2 & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;
```

Figura 2.1. Importació del laberint i inicialització d'algunes imatges binàries auxiliars

2.1. Distància entre punts

Per calcular la distància entre els punts exteriors i el vermell de l'interior del laberint he decidit dilatar les imatges binàries dels punts exteriors amb un element puntual fins a obtenir un sol component connex al unir la imatge dilatada amb la imatge binària del punt vermell. Per això començarem definint una imatge per cada parell de punts que representarà la dilatació (*trailbr/gr*) i guardarem la informació dels components connexos del parell amb *bwconncomp* a *ccrg/rb*, també haurem de guardar el nombre de dilatacions que fem, ja que determinarà la distància en píxels entre cada parell:

```
trailbr = bdot;
trailgr = gdot;

ccrb = bwconncomp(trailbr|rdot);
ccrg = bwconncomp(trailgr|rdot);
trailRBsteps = 0;
trailRGsteps = 0;
```

Figura 2.2. Declaració d'algunes variables i mapes pel càlcul de les distàncies

Fetes les anteriors inicialitzacions, declaro l'element estructural puntual i a continuació el loop while que dilatarà els “camins” fins a unir les dos parelles. A cada iteració, si la imatge unió(*trail*, *maze*) té més d'un component connex (els dos punts encara estan separats), dilatam el *trail* adient i l'intersectem amb *maze* per respectar les parets (si hem dilatat sobre píxels que pertanyen a la paret la intersecció els “borarà”) i incrementem el comptador de dilatacions en un. Al final de cada dilatació recalculem els components connexos i repetim:

```
SE1 = strel('sphere', 1);
while ccrb.NumObjects > 1 || ccrg.NumObjects > 1
    if ccrb.NumObjects > 1
        trailbr = imdilate(trailbr, SE1) & maze;
        trailRBsteps = trailRBsteps + 1;
        ccrb = bwconncomp(trailbr|rdot);
    end
    if ccrg.NumObjects > 1
        trailgr = imdilate(trailgr, SE1) & maze;
        trailRGsteps = trailRGsteps + 1;
        ccrg = bwconncomp(trailgr|rdot);
    end
end
```

Figura 2.3. Declaració de l'element estructural per les dilatacions i el loop while que les realitza

Quan acaba el loop while les variables *trailRBsteps* i *trailRGsteps* contenen el nombre de dilatacions per les parelles red-blue i red-green respectivament i, per tant, la distància en píxels d'aquestes parelles. Si volem veure l'espai explorat amb aquestes dilatacions podem fer un *montage* de les imatges lògiques resultants:

```
montage({img, maze, trailbr, trailgr});
```

Figura 2.4. *Montage* per visualitzar els resultats de les dilatacions

2.2. Camins

Per pintar els camins entre cada parella de punts utilitzarem la funció *bwistgeodesic* que ens permet obtenir la distància de cada píxel d'una imatge a una zona de la mateixa tenint en compte que alguns píxels marcats per una imatge binària “no es poden travessar”. Començarem obtenint el mapa de les distàncies a cada punt de color aplicant la funció anterior sobre les imatges binàries de cada punt aïllat i usant *maze* per marcar les parets com a píxels no navegables:

```
distR = bwdistgeodesic(maze, rdot);
distB = bwdistgeodesic(maze, bdot);
distG = bwdistgeodesic(maze, gdot);
```

Figura 2.5. Mapes de les distàncies dels píxels del laberint a cada punt de color

Fet això sumarem els mapes de distàncies per cada parella de punts, ja que els punts que mantinguin la mínima distància als dos punts seran els que formen part d'algun camí mínim que els uneix. A continuació podem “aprimar” aquest conjunt de píxels que pertanyen a algun camí mínim amb la funció *bwmorph* i els paràmetres *'thin'* i *inf* que ens reduirà el conjunt “infinitament” fins a donar un camí d'un píxel d'amplada. Aquests camins els podem combinar amb el laberint original utilitzant *imoverlay*, que ens “pintarà” el camí binari del color que especifiquem:

```
RB = distR + distB;
RG = distR + distG;

%substituïm les distàncies no numèriques per infinit pq no falli el regionalmin
RB(isnan(RB)) = inf;
pathsB = imregionalmin(RB);
solution_pathB = bwmorph(pathsB, 'thin', inf);
solrb = imoverlay(img, solution_pathB, [0 0 1]);

RG(isnan(RG)) = inf;
pathsG = imregionalmin(RG);
solution_pathG = bwmorph(pathsG, 'thin', inf);
solrg = imoverlay(img, solution_pathG, [0 1 0]);
```

Figura 2.6. Càlcul i representació dels camins entre punts a partir dels mapes de distàncies

Com indica el comentari, $RB(isnan(RB)) = inf$ serveix per definir les distàncies no numèriques obtingudes de *bwdistgeodesic* com a infinites per tal de que *imregionalmin* funcioni.

Per acabar només ens cal representar els resultats, jo he decidit mostrar els dos camins per separat i després unir-los en una sola imatge per veure els seus recorreguts conjunts amb un altre *imoverlay*:

```
solution_path = solution_pathB | solution_pathG;
sol = imoverlay(img, solution_path, [1 1 0]);
figure
montage({solrb, solrg, sol})
```

Figura 2.7. Unió dels dos camins i representació dels resultats en un *montage*

3. Resultats

Segons el script, les distàncies entre els punts vermell-blau i vermell-verd son 1842 i 1460 píxels respectivament.



	trailRBsteps	1842
	trailRGsteps	1460

Figura 3.1. Distància entre els punts del laberint

Les imatges obtingudes de les dilatacions d'aquest nombre de píxels des de cada punt es pot veure en les imatges següents:

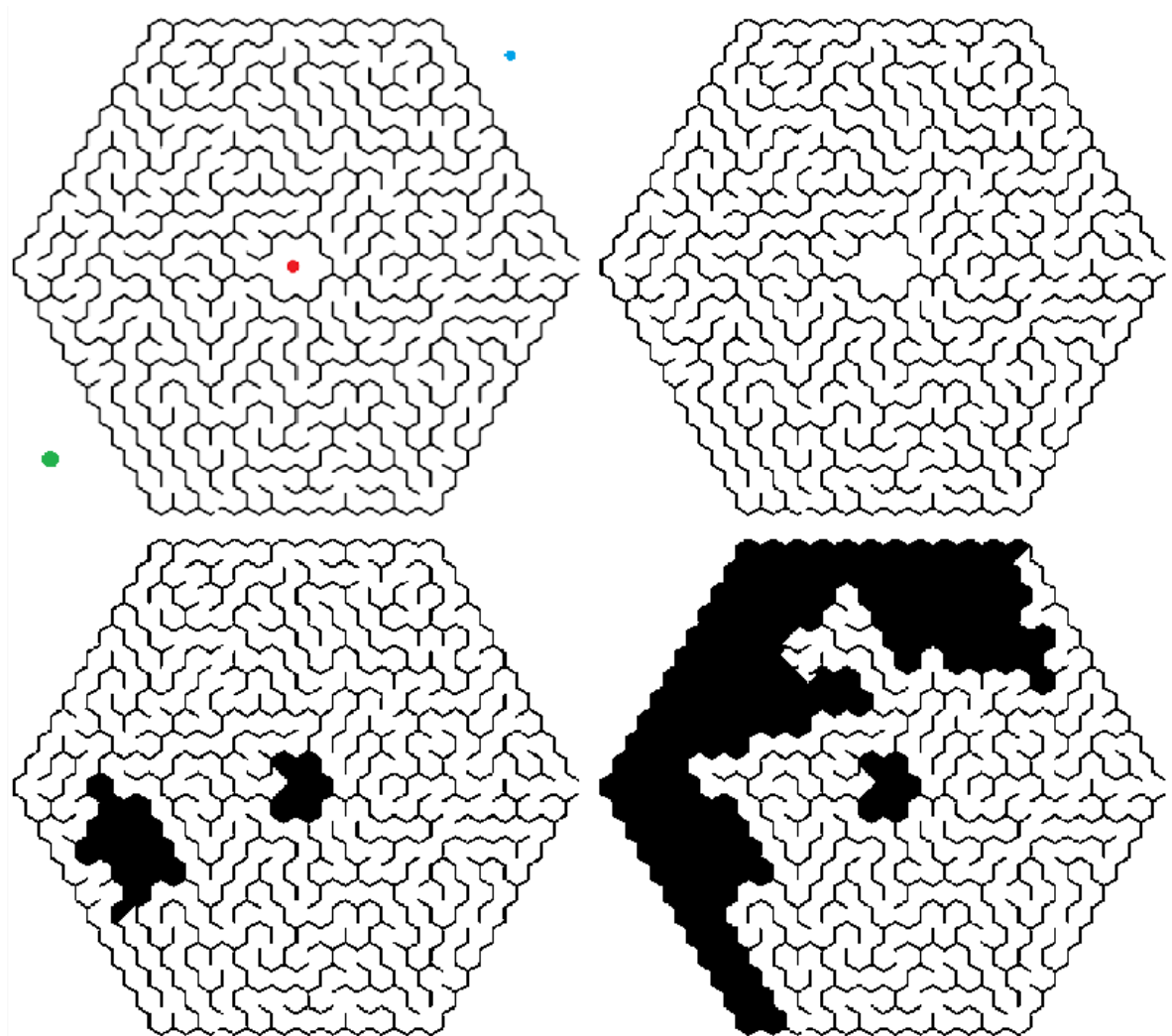


Figura 3.2. Imatges de les dilatacions del laberint pels dos punts de color

La primera imatge és la original, la segona la imatge binària *maze* i la tercera i quarta són les dilatacions de 1842 píxels des del punt blau i 1460 des del verd respectivament.

Els camins que ens dona el script per anar des de cada punt exterior fins al vermell del centre són els següents:

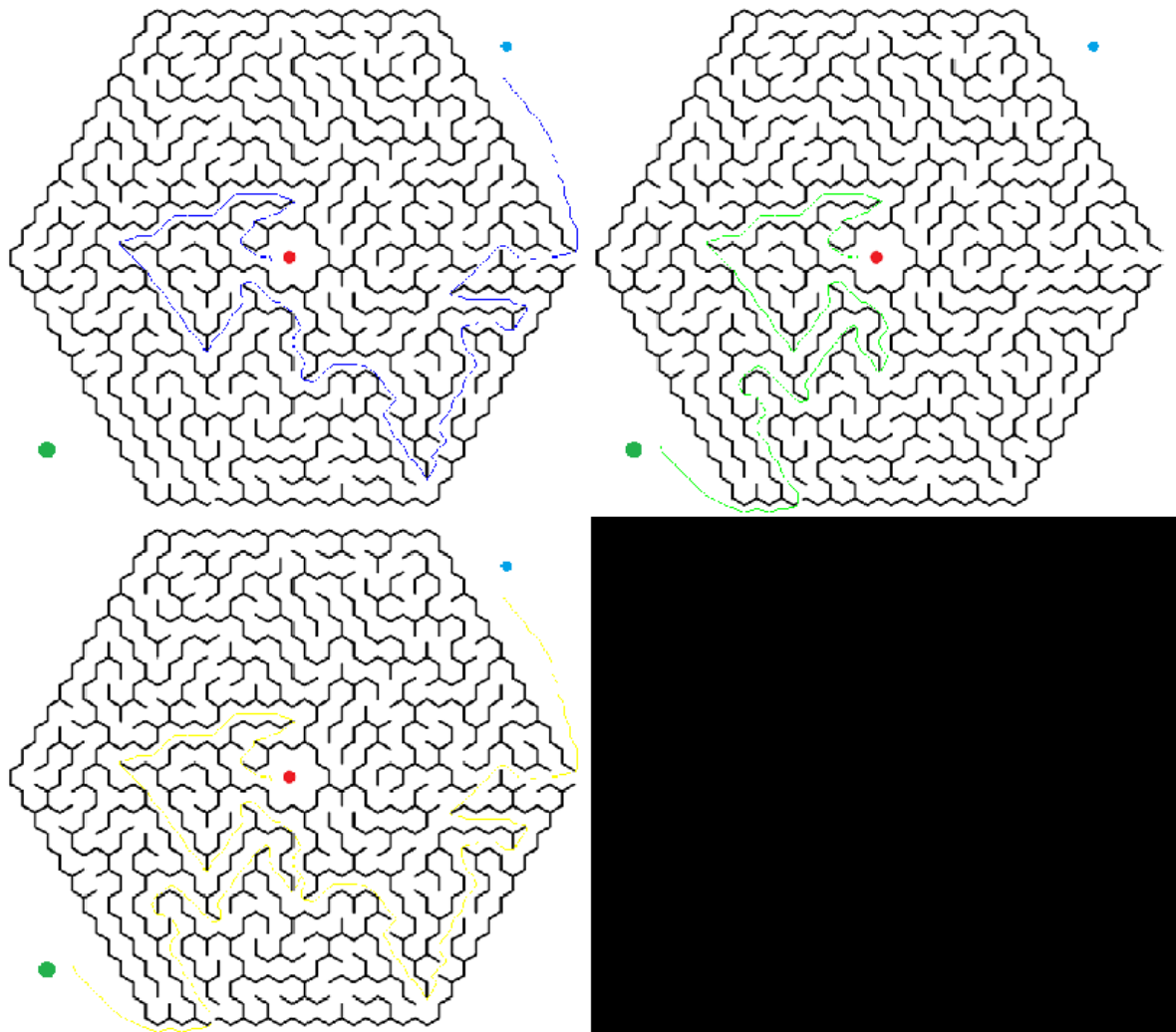


Figura 3.3. Camins entre els punts exteriors i el centre, individualment i combinats

Als camins anteriors sembla que els hi faltin alguns píxels a causa de la forma que queden representats amb el *montage*, per això adjunto també la següent imatge que mostra el camí combinat final amb millor resolució i amb una mida més gran:

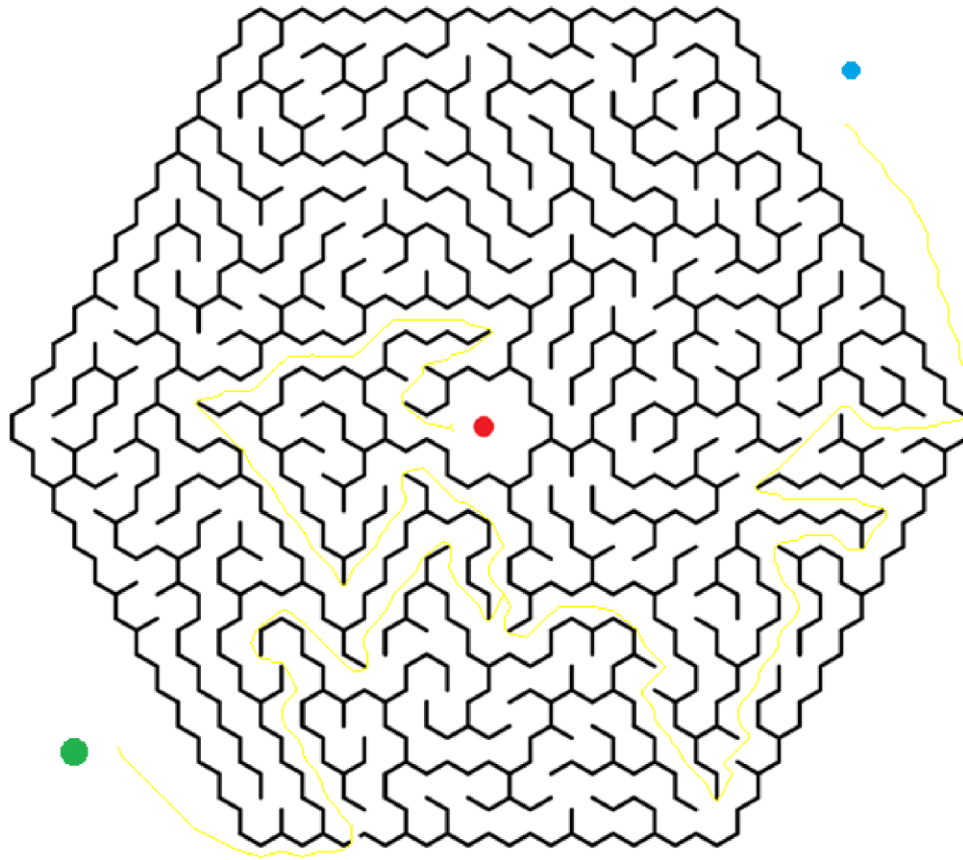


Figura 3.4. Camins entre els punts exteriors i l'interior vermell

4. Annexos

```
clear
close all

img = imread('Laberint.png');

imgHSV = rgb2hsv(img);
maze = imgHSV(:,:,3) > 0.6;
rdot = (imgHSV(:,:,1) < 0.1 | imgHSV(:,:,1) > 0.8) & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;
bdot = imgHSV(:,:,1) > 0.5 & imgHSV(:,:,1) < 0.8 & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;
gdot = imgHSV(:,:,1) < 0.4 & imgHSV(:,:,1) > 0.2 & imgHSV(:,:,2) > 0.6 & imgHSV(:,:,3) > 0.6;

% EXERCICI 1

trailbr = bdot;
trailgr = gdot;

ccrb = bwconncomp(trailbr|rdot);
ccrg = bwconncomp(trailgr|rdot);
trailRBsteps = 0;
trailRGsteps = 0;
SE1 = strel('sphere', 1);
while ccrb.NumObjects > 1 | ccrg.NumObjects > 1
    if ccrb.NumObjects > 1
        trailbr = imdilate(trailbr, SE1)& maze;
        trailRBsteps = trailRBsteps + 1;
        ccrb = bwconncomp(trailbr|rdot);
    end
    if ccrg.NumObjects > 1
        trailgr = imdilate(trailgr, SE1)& maze;
        trailRGsteps = trailRGsteps + 1;
        ccrg = bwconncomp(trailgr|rdot);
    end
end
montage([img, maze, trailbr, trailgr]);

% EXERCICI 2

distR = bwdistgeodesic(maze, rdot);
distB = bwdistgeodesic(maze, bdot);
distG = bwdistgeodesic(maze, gdot);

RB = distR + distB;
RG = distR + distG;

%substituim les distancies no numeriques per infinit pq no falli el regionalmin
RB(isnan(RB)) = inf;
pathsB = imregionalmin(RB);
solution_pathB = bwmorph(pathsB, 'thin', inf);
solrb = imoverlay(img, solution_pathB, [0 0 1]);

RG(isnan(RG)) = inf;
pathsG = imregionalmin(RG);
solution_pathG = bwmorph(pathsG, 'thin', inf);
solrg = imoverlay(img, solution_pathG, [0 1 0]);

solution_path = solution_pathB | solution_pathG;
sol = imoverlay(img, solution_path, [1 1 0]);
figure
montage([solrb, solrg, sol])
figure
imshow(sol)
```

Figura 4.1. Script de la sessió