**Introducció:**

L'inpainting o restauració d'imatges es un procés que ens permet recuperar parts deteriorades de la imatge o completar parts de la imatge en la qual la informació esta perduda amb l'objectiu de millorar-les.

**Desenvolupament:**

En el nostre cas, aplicarem l'algoritme d’inpainting amb Onion-Peel.

1.Llegim la imatge

2.Sel.lecionem la zona a recuperar

3.Cada píxel de la zona seleccionada el modifiquem i donem valor -100

4.Per cada píxel de la zona que hem seleccionat, busquem el millor píxel

5.Recorrem tota la imatge

6.Calculem l'SSD entre la finestra que estem buscant i la que correspon al píxel de la imatge que estem estudiant en aquest moment

**1. Llegim la imatge**

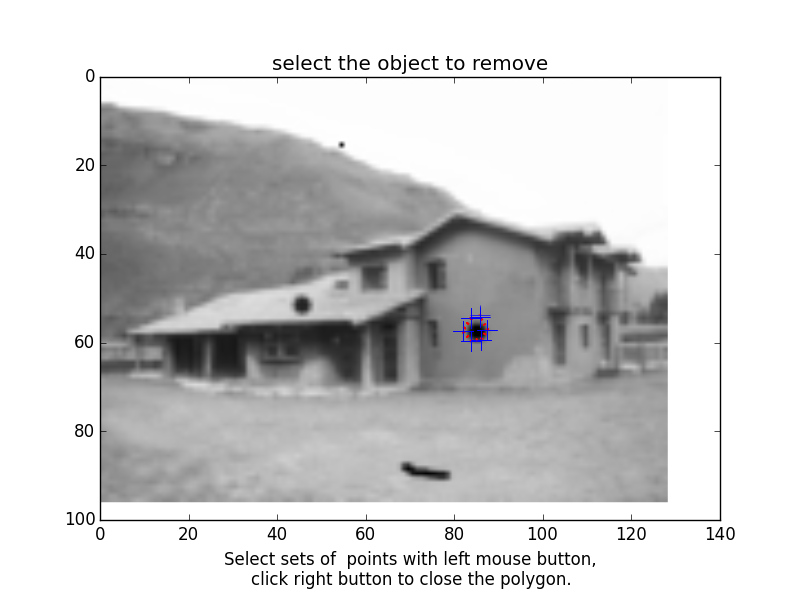
Llegim la imatge a restaurar i la re escalem per poder agilitzar els tests.

*rgb = imresize(rgb, ( rgb.shape[0]/scale,rgb.shape[1]/scale,3),interp='bilinear')*

seguidament convertim la imatge a blanc i negre img\_gris.

**2. Seleccionar zona a recuperar**

Utilitzem les funcions cedides pel professor en pràctiques anteriors per poder seleccionar els píxels que engloben la zona a recuperar amb la següent funció:



*get\_mouse\_click(img\_gris.astype('uint8'))*

**3.Donar valor a píxels a borrar**

Confinem la imatge amb la zona a restaurar amb valors a -100 per a poder tractar durant l’algoritme

*convine\_mask(zona\_delete,img\_gris)*

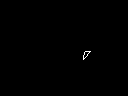
**4.Pixel a substituir:**

En aquest pas es on apliquem Onion-Pel, i el que fem es de la mascara seleccionada fem una erosió i obtenim una nova mascara, així al restar les dos mascares obtenim el contorn mes exterior que es sobre el primer que treballarem, i anirem repetint aquest procés fins que no tinguem píxels a substituir.

*zona\_delete\_erosio=ndimage.binary\_erosion(zona\_delete)*

*#busco el contorn mes extern de la part a substituir*

*delete\_ceva =zona\_delete - zona\_delete\_erosio*

**

*Tres onion contorns que tractaríem en un dels exemples aplicats*

**5.Millor píxel:**

En aquest punt recorrem els píxels de la imatge un a un i extrèiem la seva finestra de la mateixa mida de la que estem buscant. Apliquem un SSD sobre aquestes finestres i ens quedarem amb el resultat mínim.

*t, r = reccorrer\_img(finestra, finestra\_validar, img\_gris\_convine, i, j, vent)*

**6. Càlcul diferencia SSD**

Calculem l'SSD recorrent els píxels de les dos finestres i operant el segant:

*ssd += (finestra\_validar[i][j]\*((img\_finestra[i][j] - finestra[i][j])\*\*2))*

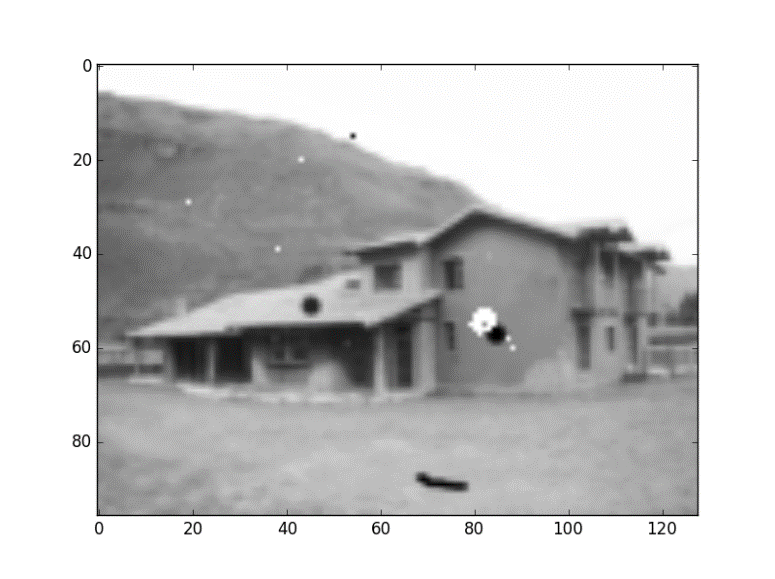
El que estem fent es el sumatori de la diferència de píxels al quadrat. Per evitar comptabilitzar els píxels que volem eliminar multipliquem per finestra\_validar, on els píxels que volem borrar tenen valor 0 i els que no tenen valor 1.

|  |  |
| --- | --- |
| [[0 0 0 0 0 0 0 0]  [0 0 0 0 0 0 0 0]  [0 0 0 0 0 0 0 0]  [0 0 0 0 0 0 0 0]  [0 0 0 0 1 0 0 0]  [0 0 0 0 1 1 1 1]  [0 0 0 0 1 1 1 0]  [0 0 0 0 0 0 0 0]] | [[1 1 1 1 1 1 1 1]  [1 1 1 1 1 1 1 1]  [1 1 1 1 1 1 1 1]  [1 1 1 1 1 1 1 1]  [1 1 1 1 0 1 1 1]  [1 1 1 1 0 0 0 0]  [1 1 1 1 0 0 0 1]  [1 1 1 1 1 1 1 1]] |

*Exemple de mascares la primera amb 1 on tenim àrea a restaurar segona (finestra\_validar) amb 0 en la mateixa àrea per a SSD*

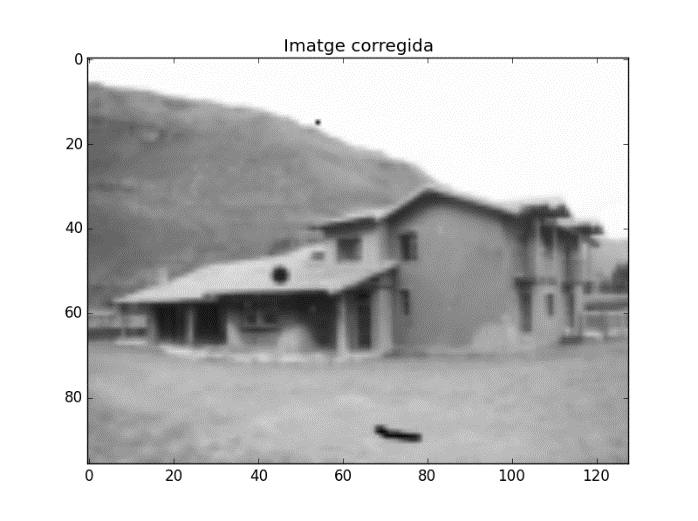
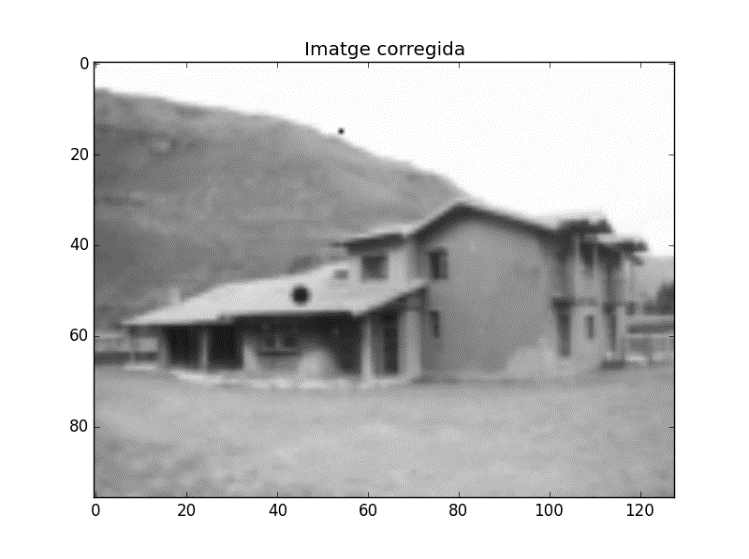
**7.Final**

Finalment substituïm els píxels de la imatge original pel píxel obtingut de l’algoritme, i això ho apliquem recurrentment a tots els píxels de la mascara obtinguda.

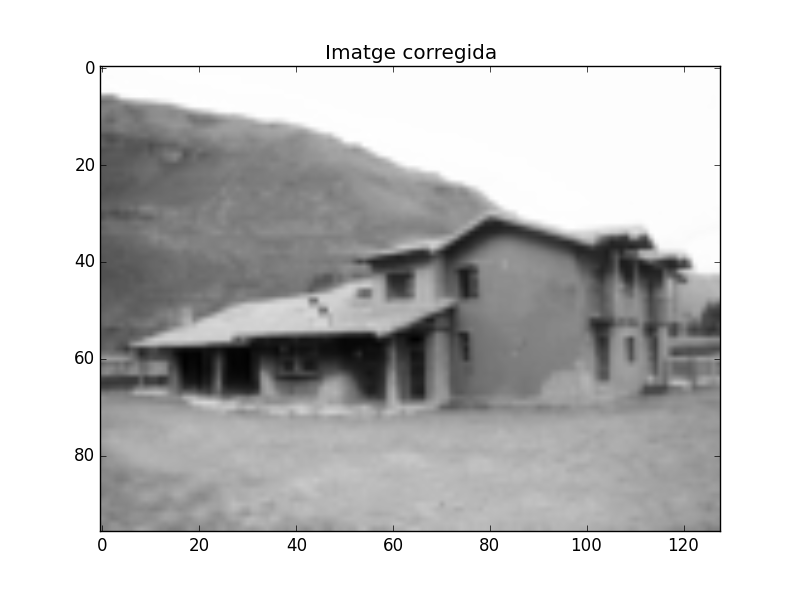


*Es poden veure resaltats en blanc els pixels escollits per substituir l’area seleccionada*

En la documentació podem trobar dos carpetes amb nom imatges i contorn on podem veure exemples de com es va substituint pixel a pixel i com es va recorrent el contorn seleccionat.

**

*Resultat final taca paret Resultat final taca herba*

****

*Resultat final amb les taques restaurades. Es pot observar que els resultats de la teulada son menys optims*

**Conclusions:**

Hem aconseguit un bon resultat d’inpainting tot i que depenem molt del format de la textura que reproduïm i de la mida de la finestra. Si la textura que conte la taca, la finestra per tractar ha de ser molt mes petita i els resultats bons son mes difícils d’obtenir. Ens hem trobat amb algun bug que no sabem solucionar i que ens produeix alguns resultats no del tot desitjables però en global aconseguim el desitjat.

Ha estat un projecte complicat per la manca d’informació sobre l’algoritme, i després d’algunes aclamacions teòriques sobre aquest hem aconseguit el resultat que es pot observar. Ens hagués agradat poder implementar la resta d’algoritmes d’inpainting per poder-ne comparar els resultats però això fa créixer el projecte de forma que no hi ha hores per realitzar-ho ja que la feina de documentació es molt gran.

Podem concloure que ha estat una assignatura molt interessant tot i que les practiques en moltes ocasions han anat mes avançades que el ritme de la teoria i això provoca un alentiment i major complicació en el desenvolupament d’aquestes. En alguns casos hem trobat documentació que ens permet avançar mes àgilment i en altres han suposat impediments que provoquen que les hores invertides per realitzar la pràctica siguin molt superiors al marcat per l’assignatura.