FIB Facultat d'Informàtica de Barcelona

Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA

VISIÓ PER COMPUTADOR

Exercici 2 de Laboratori

Facultat d'Informàtica de Barcelona

Adrian Cristian Crisan

Barcelona, Septembre de 2021

Exercici 2.

Aquest exercici està basat en una simplificació de la contribució Astrophotography with MATLAB: Imaging the Orion Nebula per Loren Shure,6 de Novembre 2020.

La idea de l'exercici es basa en fusionar dues imatges de la nebulosa d'Orion en una única imatge per aconseguir millor contrast i reducció del soroll (fig.1).

a) b)

Fig. 1 Nebulosa d'Orion: a) Imatge original, b) Imatge resultant

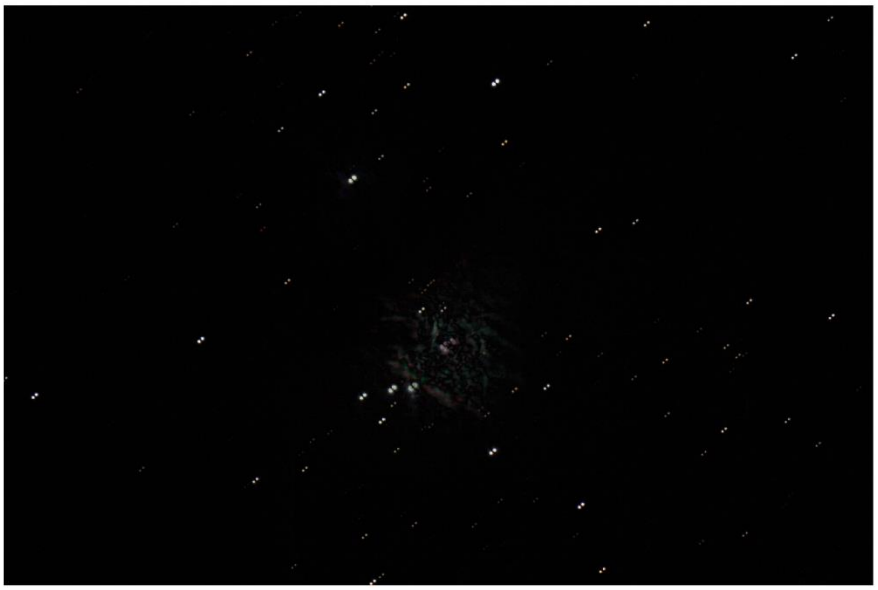
Les imatges les trobareu annexes a aquest document.

1. El primer que farem és llegir les imatges i les convertim a *double*.

```
A = double(imread('_MG_7735.JPG'))/255;
B = double(imread('_MG_7737.JPG')) /255;
```

2. Comprovem que passaria si superposem les imatges directament. Per veure-ho, restem les imatges píxel a píxel i el resultat el re-escalem entre 0-1.

```
DIF = abs(A-B);            % imatge diferencia
maxim = max(DIF(:));
DIF = DIF/maxim;           % dividim pel seu valor màxim
imshow(DIF);
```



S'observa en la imatge diferència que les imatges apareixen mogudes una respecte a l'altre. Això és degut a la rotació terrestre i a que les imatges han estat preses en instants de temps diferents. Entre una imatge i l'altre han transcorregut alguns minuts i en conseqüència les imatges apareixen desplaçades aproximadament 20 píxels en horitzontal i 20 píxels en vertical.

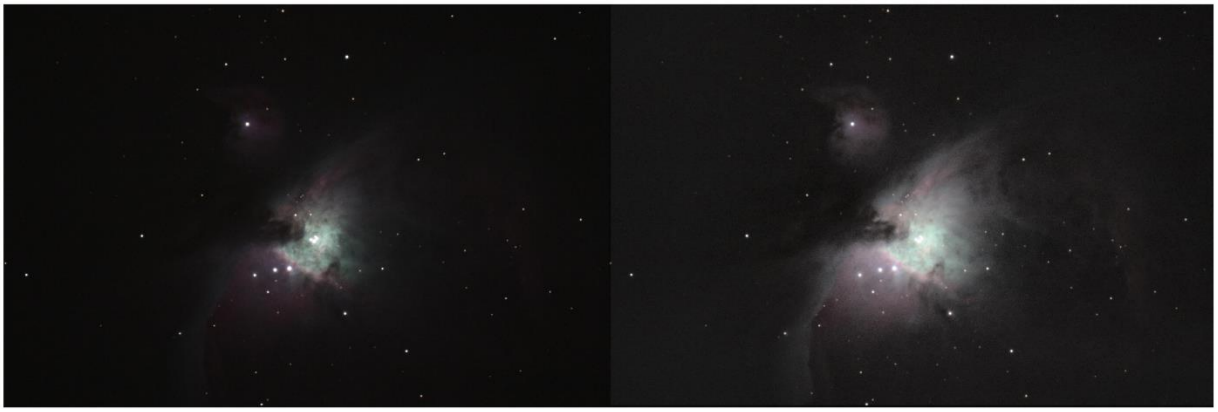
3. Traslladem per codi la imatge B 20 píxels en diagonal i observem que les imatges s'ajusten prou bé. Nota: Aquest desplaçament d'una imatge sobre l'altre es pot fer automàticament i en temes posteriors es veurà com fer-ho. Penseu que utilitzant centenars d'imatges obtindríem una imatge molt millorada pel que fa a l'exposició (captació de llum).

```
Bd = imtranslate(B,[20, -20]);
DIF = abs(A-Bd);
maxim = max(DIF(:));
DIF = DIF/maxim;
imshow(DIF);
```

4. Ara ja podem sumar les dues imatge A i Bd per obtenir una nova imatge "millorada".

```
Am = (A+Bd)/2; % imatge millorada


Obtenim la lluminositat a partir de la imatge RGB
shadow_lab = rgb2lab(Am);


Passem l'escala de lluminositat del rang [0, 100] al rang [0, 1] que és amb el
rang que treballen la resta de funcions.
max_luminosity = 100; L = shadow_lab(:,:,1)/max_luminosity;
```

```
La funció imadjust augmenta el contrast de la imatge ajustant la intensitat
d'entrada i/o sortida a més de poder jugar amb el valor gamma, valor que
descriu la relació entre el valor d'entrada i sortida.
shadow_imadjust = shadow_lab;


El que hem fet aquí és limitar el contrast pels valors d'entrada, deixar-los per
defecte al valor de sortida ([0, 1], és a dir tots els valors) i ajustem el valor
gamma per afegir lluminositat a la imatge
shadow_imadjust(:, :, 1) = imadjust(L2, [0.05, 1], [], 0.75) * max_luminosity;


La funció següent és una altra funció de millora de contrast, en aquest cas hem
deixat tot per defecte excepte el "NumTiles" que l'hem canviat a [4, 4], és a dir
mosaics de 4x4, aquest valor s'ha escollit a partir d'anar experimentant, ja que
depèn de la imatge d'entrada.
shadow_adapthisteq = shadow_imadjust;
shadow_adapthisteq(:, :, 1) = adapthisteq(L, "NumTiles", [4, 4]) * max_luminosity;


Cambiem la imatge a RGB
Am = lab2rgb(shadow_adapthisteq);
montage({A, Am})
```



Observem que la imatge ha millorat, però no obtenim un resultat desitjat, el nivell de colors no queda massa clar i és lluminositat queda massa alta, després d'estar experimentant amb aquestes opcions, no hem obtingut un resultat prou satisfactori, per això hem contemplat altres opcions:

```
És una funció que ens ajuda amb la lluminositat en àrees on la imatge té poca
lluminositat. El valor de 0.8 indica la quantitat de lluminositat que volem
afegir, què té un rang [0, 1], en aquest cas l'hem deixat a 0.8 per no
sobresaturar la imatge.
B = imlocalbrighten(Am, 0.8);
```

Amb aquesta funció observem que la millora és molt notable i assolim l'objectiu satisfactòriament.

Un altre opció que hem seguit ha sigut la versió que obtenim a partir del següent link:

https://blogs.mathworks.com/loren/2020/11/06/astrophotography-with-matlab-imaging-the-orion-nebula/#33d27527-6983-4a05-b5b0-e612647b4e49

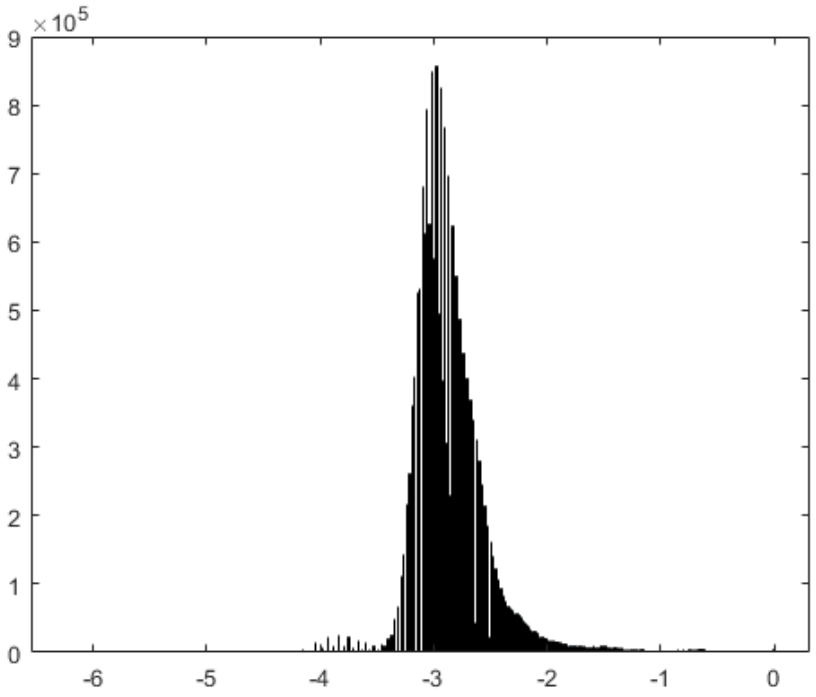
Hem modificat alguns valors per obtenir el resultat per la nostra imatge.

Primer de tot convertim la imatge RGB a una imatge HSL (*Hue, Saturation and Lightness*)

```
[H,S,L] = convertToHSL(Am);
```

Seguidament modifiquem la lluminositat de la imatge per una escala logarítmica, que com hem vist a teoria realça el contrast en regions més fosques i representem un histograma amb el resultat.

```
L2 = log(L);
histogram(L2);
```

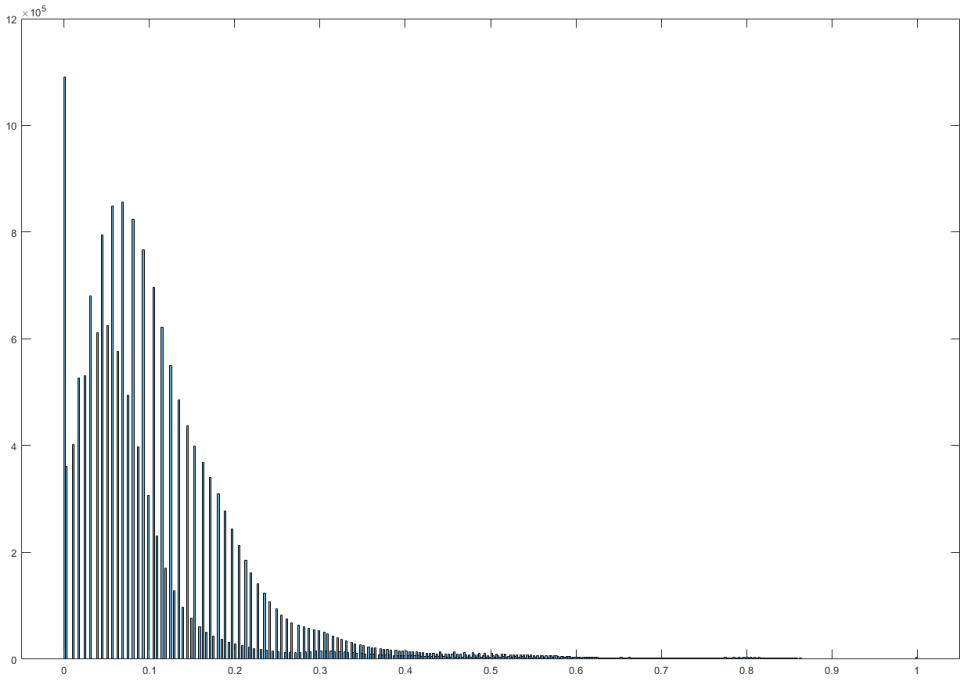


Ajustem el nivell de negre de la imatge lluminosa a la intensitat de la regió més fosca i reescalem els valors a un rang [0, 1].

```
blacklevel = -3.2;
L2 = rescale(L2,"InputMin",blacklevel);
```

Representem el resultat amb un histograma.

```
histogram(L2);
```



Obtenim la imatge amb tot el que hem aplicat fins ara, es poden observar molts més detalls que en la imatge original.

```
imshow(L2,[0 1])
```

Per realçar els colors, augmentem la saturació de la imatge i convertim la imatge HSL a RGB.

```
saturation = 2.5;
S2 = saturation*S;
Am = convertFromHSL(H,S2,L2);

montage({A, Am})
```

En aquesta versió podem veure que el resultat obtingut és satisfactori, la intensitat del color i del contrast és notable comparada amb la imatge original.

Finalment ajuntem la imatge original i les 3 versions:

Podem concloure que la millor versió és l'última de totes on es veu que no sobresaturem el contrast i assolim el nivell d'intensitat desitjat.