Lab 4: Hybrid images

Cristian M. Amaya
Universidad de los Andes
cm.amaya10@uniandes.edu.co

1.. Introduction

Hybrid images are a fun way to obtain optical illusions from mixing two images, you can obtain one just mixing the low frequencies from one image and the high frequencies from a similar picture, at close distance you can perceive one of the images, but the furthest you observe it, the more it will resemble the second one.

This is based in an article from Oliva, Torralba, and Schyns, which proposed that we tend to relied on the high frequencies when we observe a picture, but as we observed from greater distances only the low frequencies are visible and recognizable.[1]

2.. Methods

2.1.. Original Images

First image: My father

It's an old photo from my father in his first years as a police officer, he was in his early twenties in this picture, refer to Figure 1.

Second image: Old picture of me

It's an photo that I took for a visa application back in 2014, the resemblance with my father's photo was really noticeable once my mother print my photo. Please refer to Figure 2.

2.2.. Modifications

To obtain a better result, both images were crop and scale to align both faces and obtain a similar distance between the borders and bodies. Refer to figures 3 and 4.

Additionally, inside Matlab workspace, both pictures were resize to a common size for latter addition and to implement more easily the low pass and high pass filter.

2.3.. Hybrid Image

To obtain a hybrid image, we need to add the low frequencies of the first image and the high frequencies of the second. To obtain this frequencies, I used the fast Fourier transformation and a binary circle matrix to filter both images with a low and a high pass filter. Refer to the first code snippet.

Before we multiply the binary circle matrix,we need to shift both transformed images to order the frequencies from low to high starting from the center. The multiplication will keep low frequencies with the binary circle and high frequencies with the binary circle counterpart.

In the end, we add each transformed channels to their corresponding pair, apply the inverse Fourier transformation apply the inverse shift function and concatenate the real part of the channels.Refer to Figure 5 for the result.

2.4.. Gaussian Pyramid

Using the second code snippet, we can obtain the Gaussian pyramid for the hybrid image, using a cell structure and impyramid, the original image can be reduce to different scales.

Using subplot and linking the axes of each figure, we can observe the elements from the Gaussian pyramid at their corresponding scale.Refer to Figure 6.

Additionally, a Gaussian pyramid was generated with the reduce images resize to the original size to appreciate better the hybrid image.Refer to Figure 7.

3.. Conclusions

The parameters in both filters are essential to obtaining a good hybrid image, and also the aligned of the pictures provides a more pleasant image to look at.It's a good exercise to comprehend the Fourier transformation and its used

to filter signals more easily.

References

[1] Hay, J. Project 1: Hybrid Images. Cs.brown.edu. Retrieved 23 February 2017, from http://cs.brown.edu/courses/cs143/proj1/

Images

Originals

Figure 1. Picture from my father in the police

Figure 2. Picture from a Visa application

Modified versions

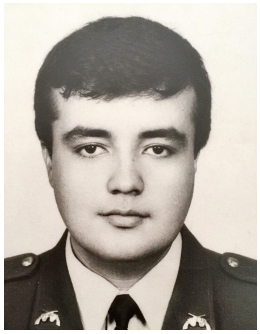

Figure 3. Cropped picture from my father

Figure 4. Cropped picture from me

Final Hybrid image

Figure 5. Hybrid image

Gaussian Pyramids

Figure 6. Gaussian pyramid with 4 elements

Figure 7. Gaussian pyramid with resize elements

Code Snippets

```
%Transformacion de Fourier
mf1=fft2(matt(:,:,1));
mf2=fft2(matt(:,:,2));
mf3=fft2(matt(:,:,3));
pf1=fft2(pop(:,:,1));
pf2=fft2(pop(:,:,2));
pf3=fft2(pop(:,:,3));


%Generacion filtro pasa bajas y pasa altas
circulo=ones(siz);
centrox=size(circulo,1)/2;
centroy=size(circulo,2)/2;
for i=1:size(circulo,1)
    for j=1:size(circulo,2)
        if((i-centrox)^2+(j-centroy)^2<=72)
            circulo(i,j)=0;
        end
    end
end
circulo2=ones(size(circulo))-circulo;
```

```
%Numero de imagenes en la piramide
n=7;

%Piramide
pir=cell(n,1);

%Piramide a la misma escala
pir2=cell(n,1);

%Guardo original
pir{1}=hyb;
pir2{1}=hyb;
h=subplot(1,n,1);
imshow(hyb);

for i=2:n
 %Generacion del siguiente elemento de la piramide
   pir{i}=impyramid(pir{i-1},'reduce');
   hi=subplot(1,n,i);
 %Almaceno el handle
   h=[h hi];
   pir2{i}=imresize(pir{i},siz);
   imshow(pir{i});
end

%Observar a sus respectivas escalas la piramide
linkaxes(h,'x');

%Guardo piramide a la misma escala
Pir2=cell2mat(pir2);
imwrite(Pir2,'pyr2.jpg');
```