University of Brescia
School of Engineering
Master Course in Communication Technologies and Multimedia

Multimedia Communication Services

Laboratory Experience, No.1

Students:
Casali Nicolò
Etansa Yonas
Ghitti Alberto
Cheima Ben Soltane
Lombardi Stefano
Mutti Andrea
Tadewos Somano

Academic Year 2013/2014

Contents

List of Figures

Abstract

Laboratory Experience, No 1.a

In the first part of this laboratory experience we read an image in RGB format and converted it to YCbCr format.

In the second part we subsampled the chrominance components to which human visual system is less sensitive than luminance one.

In the third part we read frames from a CIF video file and filtered the luminance component of frames with low pass filter before subsampling in order to avoid aliasing phenomena.

Laboratory Experience, No 1.b

In the first part of our laboratory experience we produced synthetic sinusoidal signal with varying frequency in order to test our visual system sensitivity to different frequency and tried to compare with contrast sensitivity theoretical function.

In the second part we produced synthetic moving sinusoidal signal whose frequency depends on time and space. As the velocity of the sinusoidal movement increases, our eye is not able to distinguish the direction of motion.

Laboratory Experience, No 1.a

Color space conversion

First we opened an image in the *.tiff* format and displayed the red, green and blue component separately.
After importing an image with *imread* instruction in Matlab language, we can display the components reading separately one of the three channels that compose the returned variable. The result obtained is shown in figure 1.

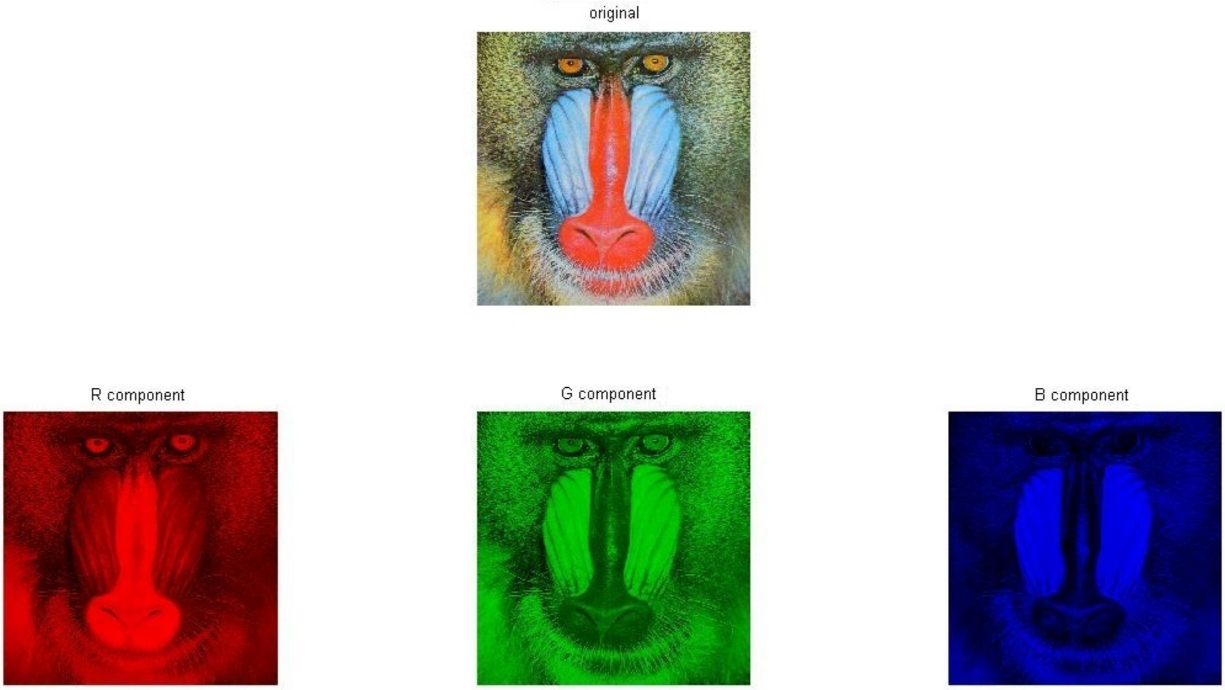


Figure 1: Read image with its RGB separate components.

After that the image has been converted in a new format called YCbCr. This can be done very easily with a function already implemented in the language (*rgb2ycbcr()*) or transforming the values of each pixel using the following transformation matrix (2).

$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0 & 1.4025 \\ 1.000 & -0.344 & -0.7142 \\ 1.000 & 1.773 & 0 \end{pmatrix} \begin{pmatrix} Y \\ C_b - 128 \\ C_r - 128 \end{pmatrix}$$

Figure 2: RGB - YCbCr transformation matrix.

The results are proposed in the following figure (3). To correctly visualize a single YCbCr component using a RGB format it's necessary to set on the average level other 2 components and then come back to RGB space.

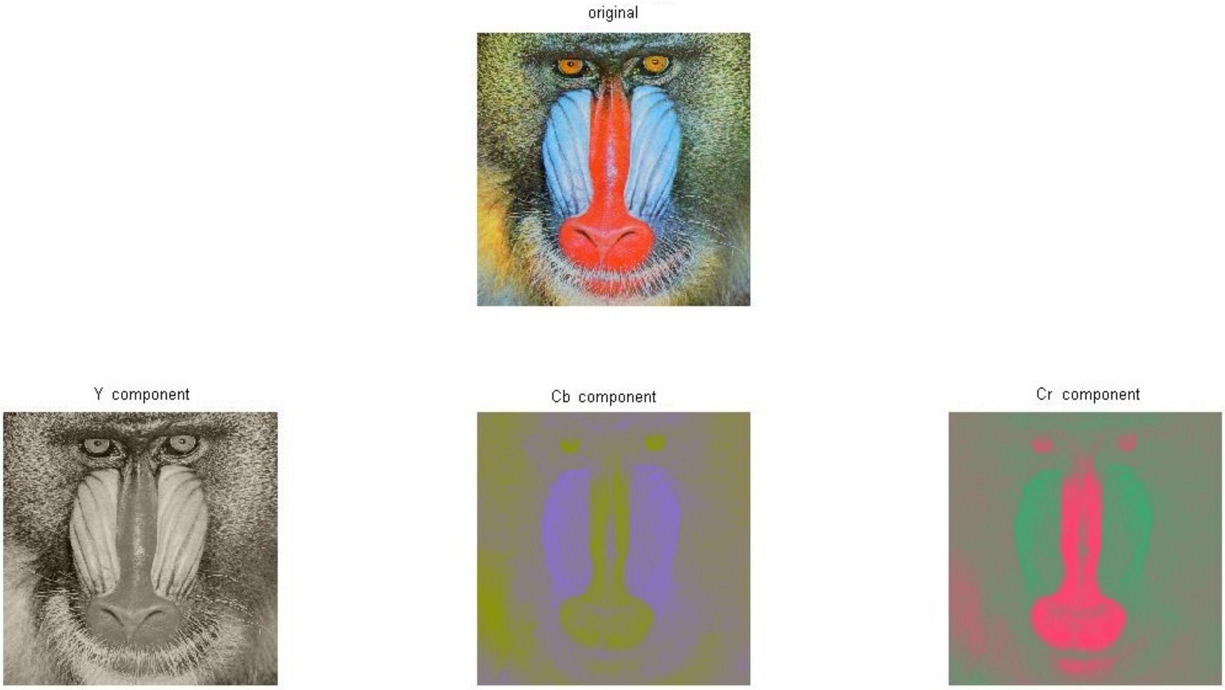
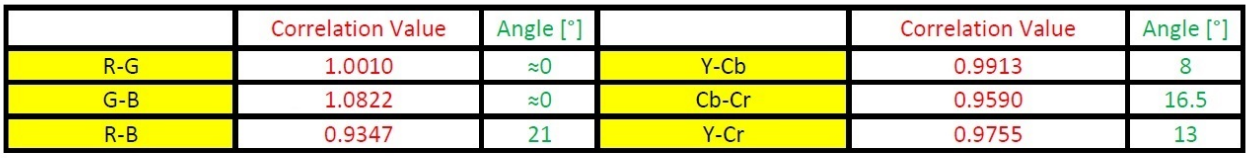


Figure 3: Read image with its YCbCr separate components.

To understand the reason for using such a format, we have evaluated the correlation between couples of components in each space. The results are proposed in the table (4).



	Correlation Value	Angle [°]		Correlation Value	Angle [°]
R-G	1.0010	≈0	Y-Cb	0.9913	8
G-B	1.0822	≈0	Cb-Cr	0.9590	16.5
R-B	0.9347	21	Y-Cr	0.9755	13

Figure 4: Correlation between components.

From the analysis of this image, the results obtained shows that the *red* and *blue* components are very uncorrelated due to the big zones near the baboon's nose which are purely red or blue. For *red* and *green* we see a very high correlation, that is almost one, so vectors representing them are almost parallel. The evaluation has been done approximately because the value was a little bigger than one due to approximations so cannot be evaluated directly. Relating YCbCr components in general the correlation among them is usually lower than the one between RGB components: this isn't true in this particular case because of the low correlation of *red* and *blue* channels explained before.

Subsampling of the chrominance components

The second part focused on subsampled image formats with YCbCr representation. We used both 4:2:2 and 4:2:0 formats to represent the previous image.
In the first case (4:2:2) we keep the luminance component untouched while we use only half of columns for both crominance components, while for 4:2:0 we sample also by rows.
After producing those images using the function *downsample*, we reconstructed the images by interpolating the downsampled components and after that we returned to RGB representation to show them on screen. The result is the following one (5).

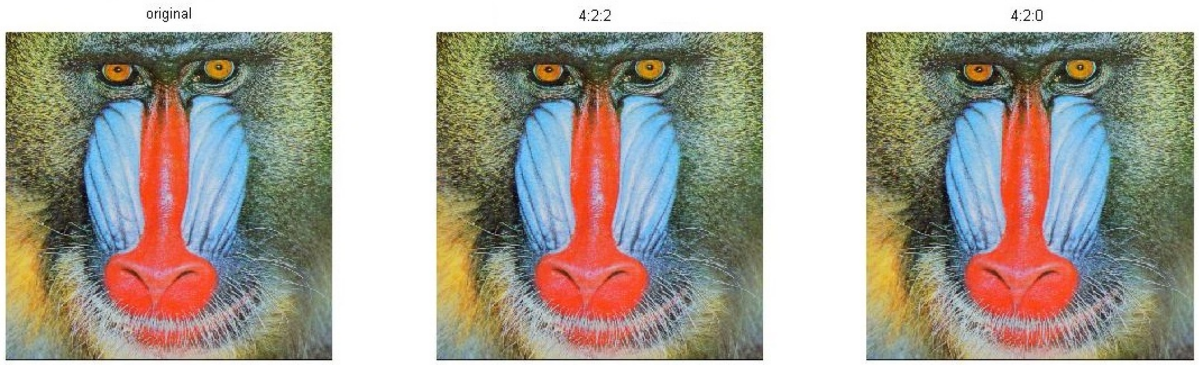


Figure 5: Original and subsampled images (4:2:2 and 4:2:0).

As we can see also by downsampling the components, due to their low correlation, the loss in the quality isn't appreciable in normal dimension. We could try to zoom it in some zone, for example in the zone of the nose (6).

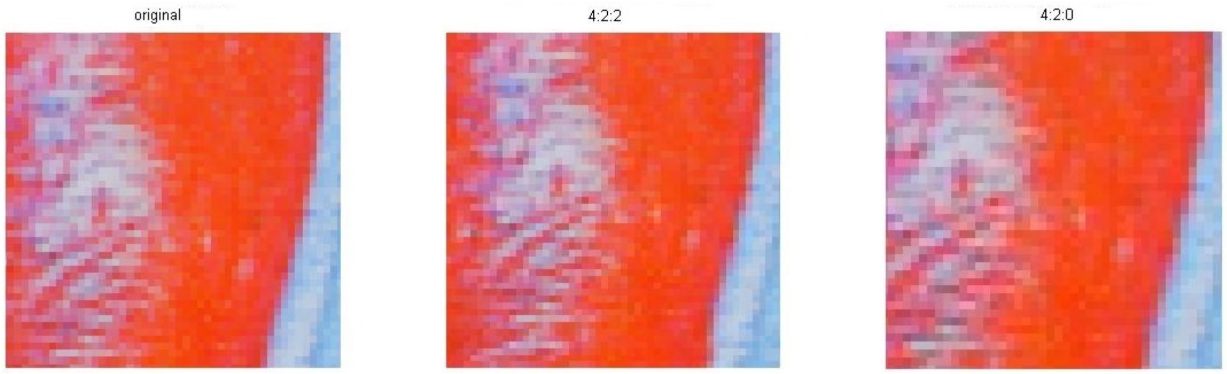


Figure 6: Particular of the original and subsampled images.

In this case by zooming the image we can see some little difference in the image but, again, the overall aspect of the image still remains pretty clear.

Video formats

In the third part we focused our attention on YUV videos. In this coding each frame is represented by its YCbCr components in format 4:2:0.
First we read the files by extracting the luminance component Y by carefully adding an offset of 51 bytes for each frame: the first part in fact is a string containing the number of the frame and some informations about the format of the image.
After isolating Y component we saved it in a .y file.
After this we focused on the first frame of the Y component and filter it with a median filter with kernel dimension 3x3 using the function *medfilt2*.
In the following image (7) we can see the result and also the residual image, difference between filtered and unfiltered frames.

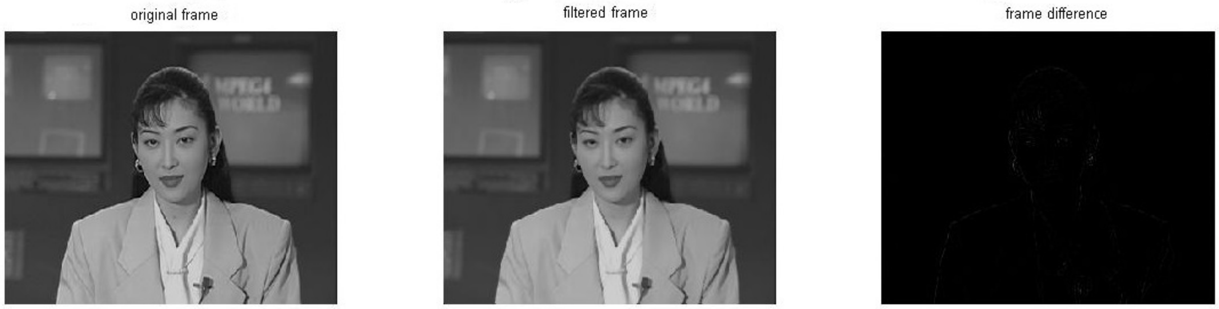


Figure 7: First video frame, filtered and unfiltered.

As can be seen, the filter has the effect to blur the image, reducing abrupt changes in luminance so the residual is somehow like an edge detection result.
Finally we converted the image to 4:2:0 QCIF format. After filtering all components with the median filter previously mentioned, each component has been subsampled, spatially and temporally, using *downsample* function, obtaining a frame half in size and also has been kept only one frame each two ones. After that each component has been wrote in binary way in a file called *akiyoQCIF.yuv* and also converted in *.avi* format to been easily displayed by standard media players.

Laboratory Experience, No 1.b

Contrast Sensitivity Function

Contrast sensitivity is a measure of the ability to discern between luminance of different levels in a static image. It varies between individuals, reaching a maximum at approximately 20 years old and at spatial frequencies of about 2–5 cycles/degree.
In our image (8), the contrast amplitude depends only on the vertical coordinate, while the spatial frequency depends on the horizontal coordinate. Observe that for medium frequency you need less contrast than for high or low frequency to detect the sinusoidal function.

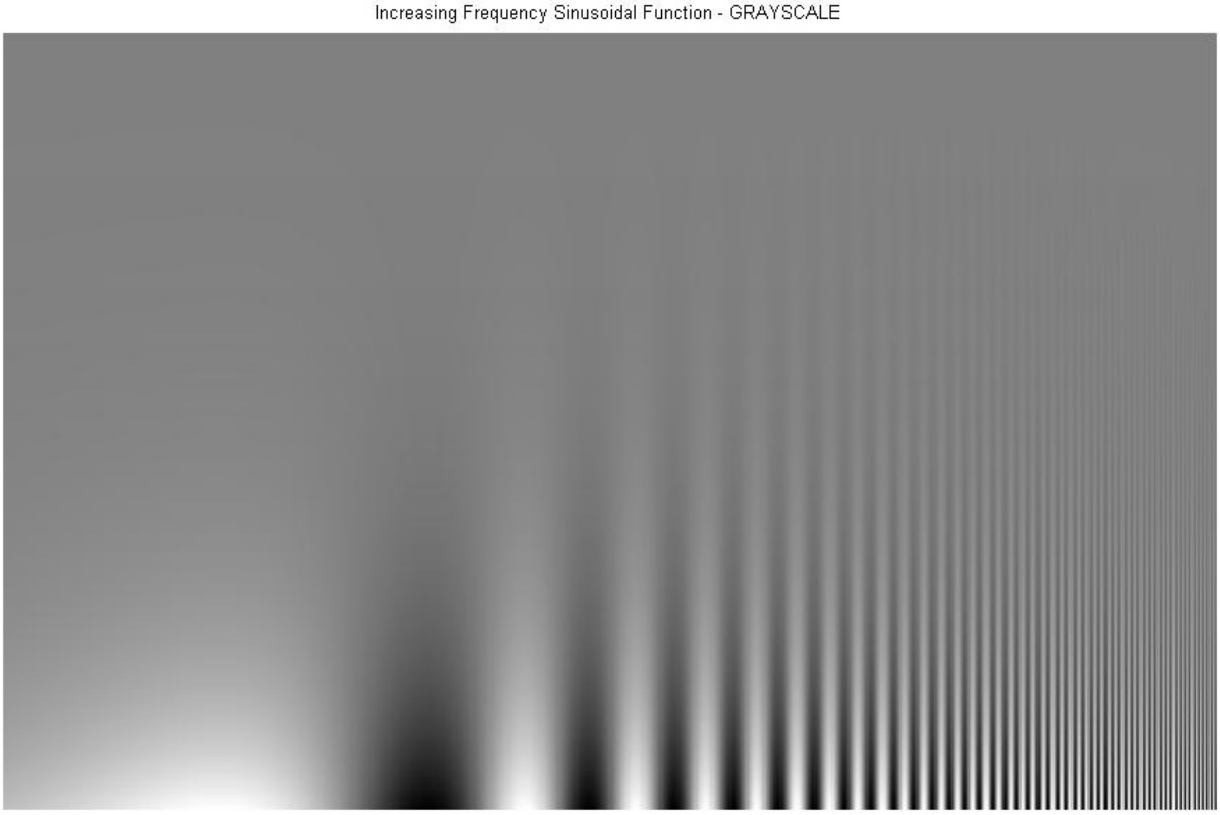


Figure 8: Increasing frequency sinusoidal function in grayscale color.

We can plot the same image in Cb(10) and Cr(11) components. We can see that the information about the contrast stays only in the Y(9) component because the contrast sensitivity describe the ability of visual system to distinguish bright and dim components of an image.

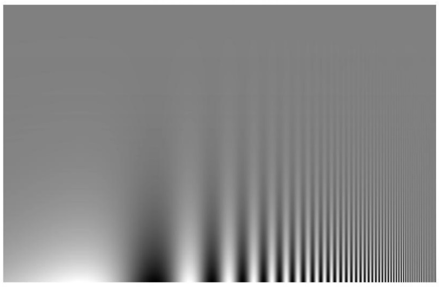


Figure 9: Y component of the test image.

Figure 10: Cb component of the test image.

Figure 11: Cr component of the test image.

We can check our sensitivity also with image with different colors, more or less with the same sensitivity result.

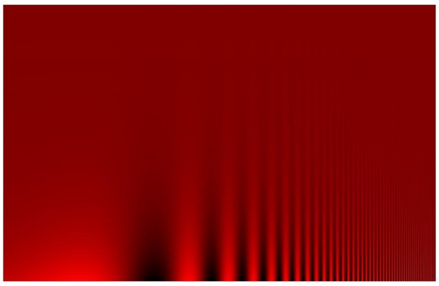


Figure 12: Red component of the test image.

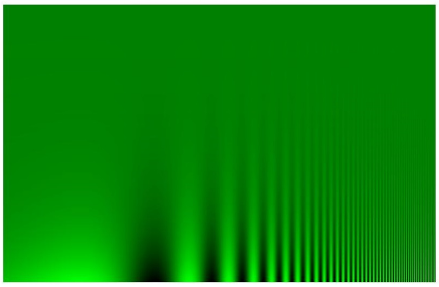


Figure 13: Green component of the test image.

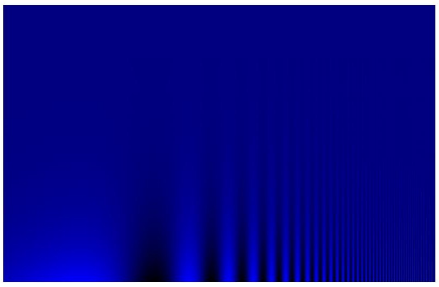


Figure 14: Blue component of the test image.

Now we can plot ours sensitivity function. The contrast sensitivity function tells us how sensitive we are to the various frequencies of visual stimuli. If the sinusoidal frequency of signal increases our sensitivity change and we will not be able to recognize the stimuli pattern any more. If the stripes then become wider and wider, or the sinusoidal frequency decreases, there is a threshold width, from which on we are able to distinguish the stripes. We took into account the contrast sensitivity function proposed by Manos and Sakrison (15).

$$A(f) = 2.6 \cdot (0.0192 + 0.114 \cdot f) \cdot e^{-(0.114 \cdot f)^{1.1}}$$

Figure 15: Contrast Sensitivity Function

Variable *f* in equation is the spatial frequency of the visual stimuli given in cycles/degree. The function has a peak of value 1 approximately at f=8.0 cycles/degree and is meaningless for frequencies above 60 cycles/degree (16).

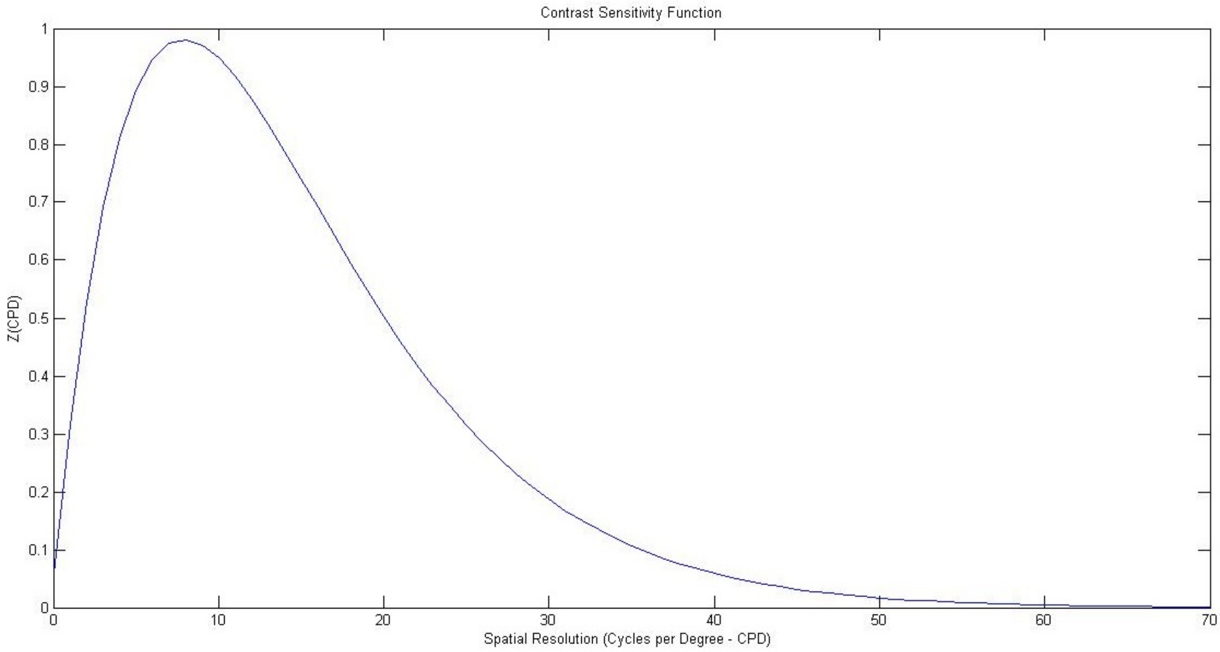


Figure 16: Plot of the Contrast Sensitivity Function.

As we can see from the function plot, it is similar to low-pass filter. The reason why we can not distinguish patterns with high frequencies is the limited number of photoreceptors in our eye.

Spatio-Temporal Sensitivity

In this part of our laboratory experience we have created a video with moving sinusoids (17). We can see that increasing the velocity of sinusoids, we can't trace in which direction the sinusoid is moving, due to in successive frames the sinusoid's amplitude changes too rapidly for the display sampling.

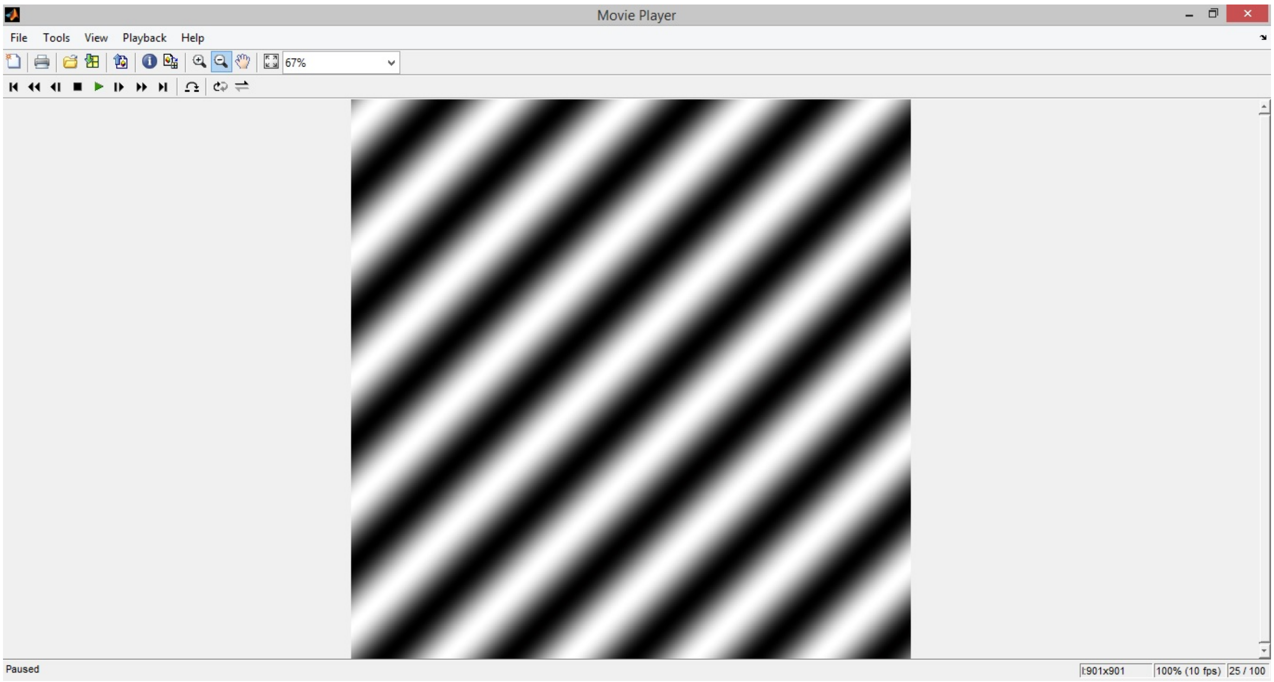


Figure 17: Frame of the video with moving sinusoids.