Project Report

TOPIC 1: INTENSITY TRANSFORMATIONS & SPATIAL FILTERING

Image Processing (BEJ42903)

Ts. Dr. Suhaila Binti Sari

Universiti Tun Hussein Onn Malaysia

07/06/2023

Semester 2 2023

Moritz Hoehnel (JD220004), Mattis Ritter (JD220003)

Table of Contents

1. Introduction

Everybody has already been in the situation, where a picture was taken of themselves or of a sight during vacation. But afterwards the pictures turn out to have flaws, like being too dark or need smoothing. But now you can not share the pictures with family or friends. This project is addressing this issue by establishing an android application to process the images with flaws.

The implemented methods get described in the lecturer paper “INTENSITY TRANSFORMATIONS AND SPATIAL FILTERING” from Gonzalez, R. and Woods, R.. “Intensity transformations operate on single pixels of an image for tasks such as contrast manipulation and image thresholding. Spatial filtering performs operations on the neighborhood of every pixel in an image. Examples of spatial filtering include image smoothing and sharpening“. The methods can be very useful for every person, to help improve their own pictures and the application will do the work.

This report is the documentation of the app and its development. It will treat the transformation techniques, the development tool, an user instruction and an analyzation of its output.

2. Transformation Technique

The following chapter will describe the applied transformation techniques. The android application does one transformation of each of the topics, one intensity transformation and one spatial filter method.

2.1 Intensity Transformation

Gupta, B. and Agarwal, T. K. describe the need of an intensity transformation in their paper “New contrast enhancement approach for dark images with non-uniform illumination”. They state that pictures can be taken in in not optimal light conditions, that can lead to pictures that are too dark. Hussain, K. and others state in their research paper “A histogram specification technique for dark image enhancement using a local transformation method”, that wrong lightning can also be a result of a wrong capturing device operation. They say that an enhancement is important that the picture is still worthy to keep. As first solution they suggest histogram stretching, as it is one of the most common ways. But later they argument that it has several downsides for pictures which have been too dark and need brightening. The first issue is that low intensity pixels are at danger of being merged. This leads in a contrast lost in this intensity area. A second issue can be that pictures can start to look unnatural after the transformation, because of a shifted contrast. This is why the development team has opted to implement a transformation that uses addition. This will brighten the picture, not merge low level intensities and not shift the contrast.

While during histogram stretching a new value is assigned to each pixel according to the stretching that took place, based on the histogram analyzing, for addition transformation a certain value gets added to every pixel intensity value. If the value reaches the maximum intensity value, it can not increase anymore. This is called clipping. To avoid it the user can decide on its own how much brightening is needed.

2.2 Spatial Filter

For spatial filtering the team chose the concept of the average filter. The concept is described in Li, T., Wang, J. and Yao, K. describe in their paper “Visibility enhancement of underwater images based on active polarized illumination and average filtering technology”. Furthermore, it gives good insight into problems of image processing of underwater-pictures and how they need to be treated.

One of the biggest issues while taking photos underwater is the environment, the water. It has many particles that reflect light. That is why the picture consists of the light rays that hit the lens directly and of light rays that hit the lens through reflection. The goal of the paper’s processing process.is to eliminate the reflected light from the picture. Only then you can really see what the environment or a certain object looks like.

The average filter is used to approximate the backscattered light. It is used as an image blur filter. They are describing the process of averaging in putting a window of certain size over each pixel and its neighbors and then calculating the average value of it. For their special usage, they describe a coding method that increases the calculation speed. It is basically achieved by moving the window by one pixel and only remove the outside pixel, the others stay in the list, used for calculation. As they are already summed up and can be reused.

We assume that the standard user of the app is not doing underwater photos. In spite of that the filter is an important feature for the application. It is used as blur filter. The usage is simpler for this, as it can be used directly on the pictures, without further processing. It is needed for removing unwanted noise. While analyzing the sample pictures, it was clear that there are noise problems. For the implementation it was decided to use an average filter that uses a 5x5 mask and also no zero padding used. The mask size was determined after some test with different size masks to the sample pictures of the progress report.

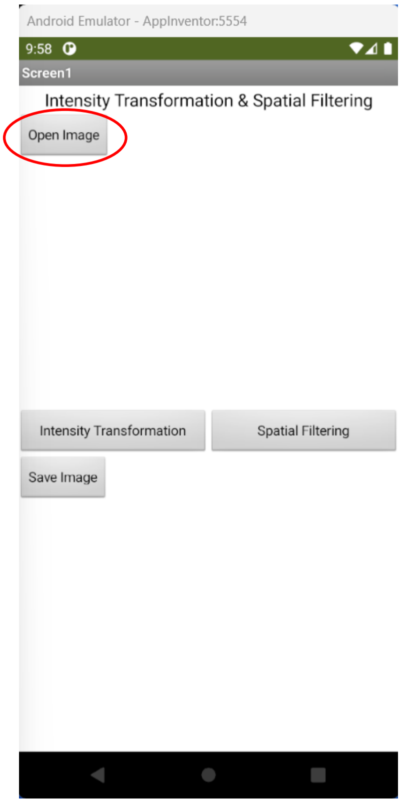
3. Development Software

For the application development the tool MIT Inventor was chosen. It is a web-based tool that was created by the Massachusetts Institute of Technology under the direction of Professor Hal Abelson. In his personal column it is stated that “the MIT App Inventor, stems from the idea that anyone should be able to take powerful computational tools and create meaningful, original mobile applications for smartphones and tablets that have impact on the world and in their daily lives”. Another advantage is that it is free to use.

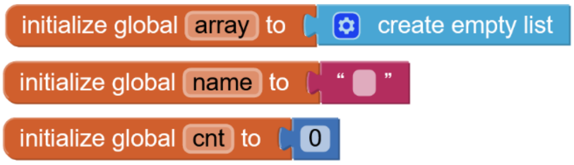
The general application development process is to implement the interface, then create the logic. For the interface, in a designer, you can drag and drop ready to use graphical items like a button or a label on the screen and position wherever desired. If all design items are placed, the developer can create the logic. The design language consists of blocks. The developer puts together different blocks which represent elements like an if questionnaire. The logic is event based. That means that as example a button triggers the execution of the logic.

In the following chapter the blocks of the image processing application will be explained while each function and its interface usage is discussed. For the blocks section only standard blocks have been used.

4. Application Instruction

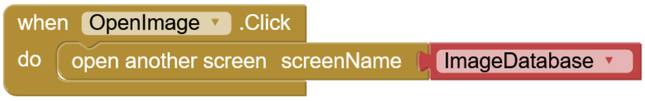

Android Emulator - AppInventor:5554
9:58
Screen1
Intensity Transformation & Spatial Filtering
Open Image
Intensity Transformation
Spatial Filtering
Save Image

Firstly, a number of global variables is initialized. The list “array” contains the intensity of each pixel from the input image. The other variables are responsible for naming the image once it is saved.



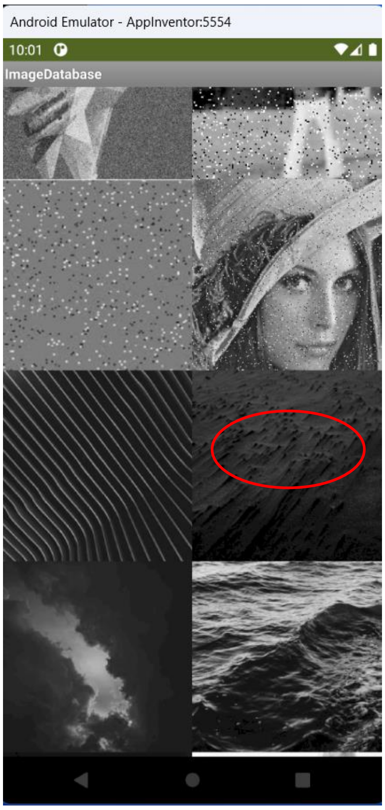
```
initialize global array to create empty list
initialize global name to " "
initialize global cnt to 0
```

The home screen includes the headline, the buttons for opening and saving the images, as well as the buttons for the different operations.

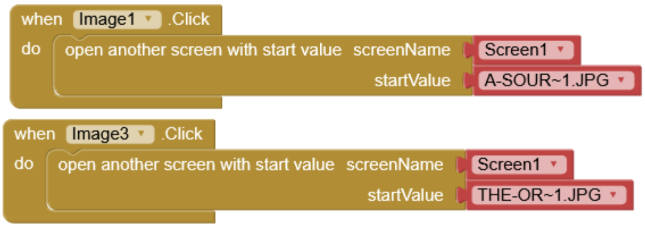


```
when OpenImage .Click
do  open another screen screenName ImageDatabase
```

When clicking on the “Open Image” button another screen, that contains the images from the database is opened.



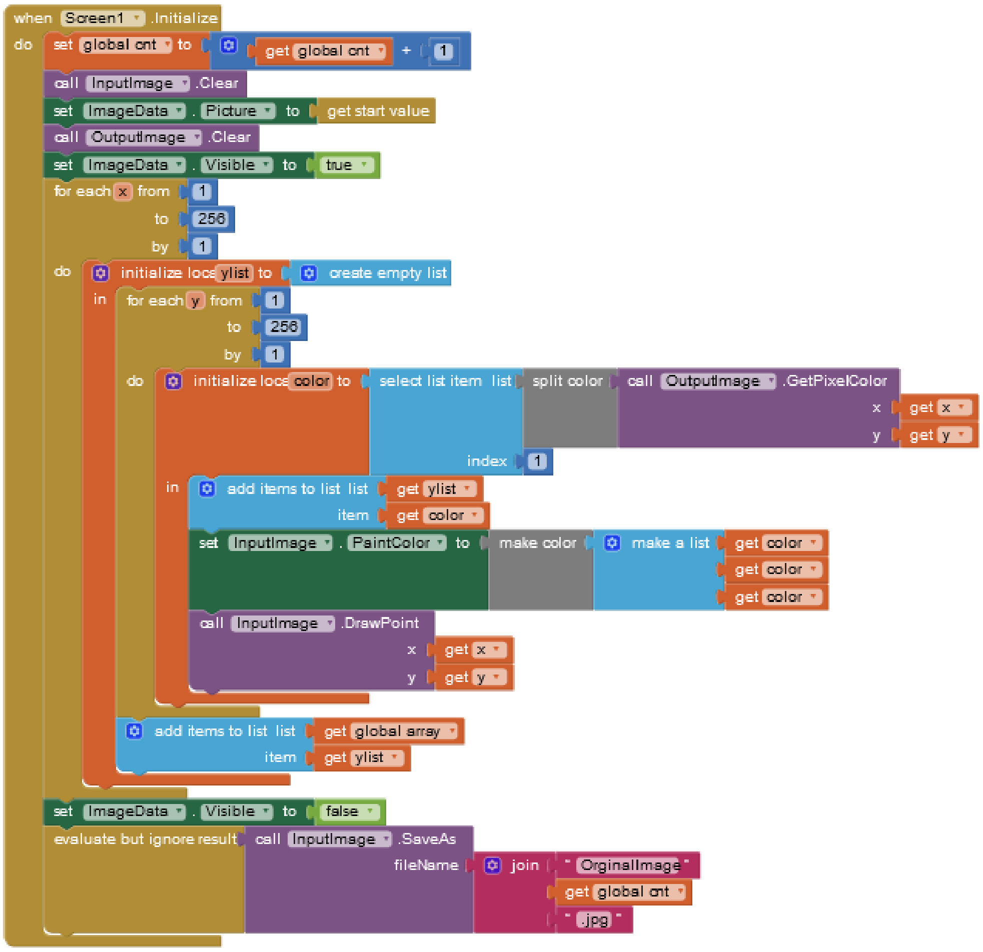
The screen “ImageDatabase” is scrollable and shows all the images. When clicking on one of the images, the user returns to “Screen1” and the chosen image is carried as a start value, as shown in the snippet below.



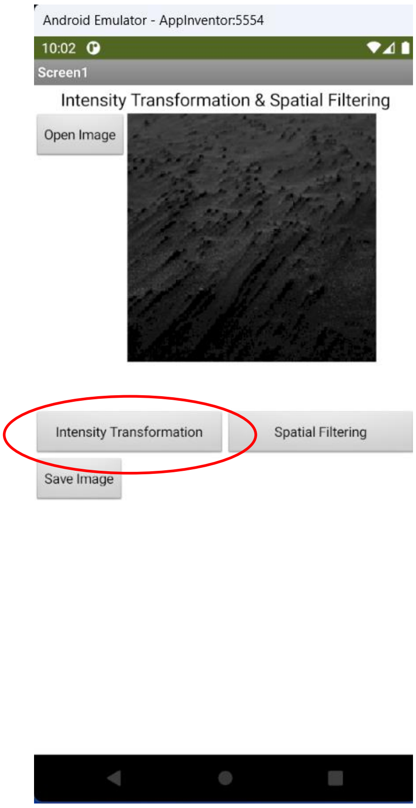
```
when Image1 .Click
do  open another screen with start value screenName Screen1
                                         startValue A-SOUR~1.JPG
when Image3 .Click
do  open another screen with start value screenName Screen1
                                         startValue THE-OR~1.JPG
```

On the home screen the input image is now displayed in the top right corner. This is shown in the next screenshot of the user interface.

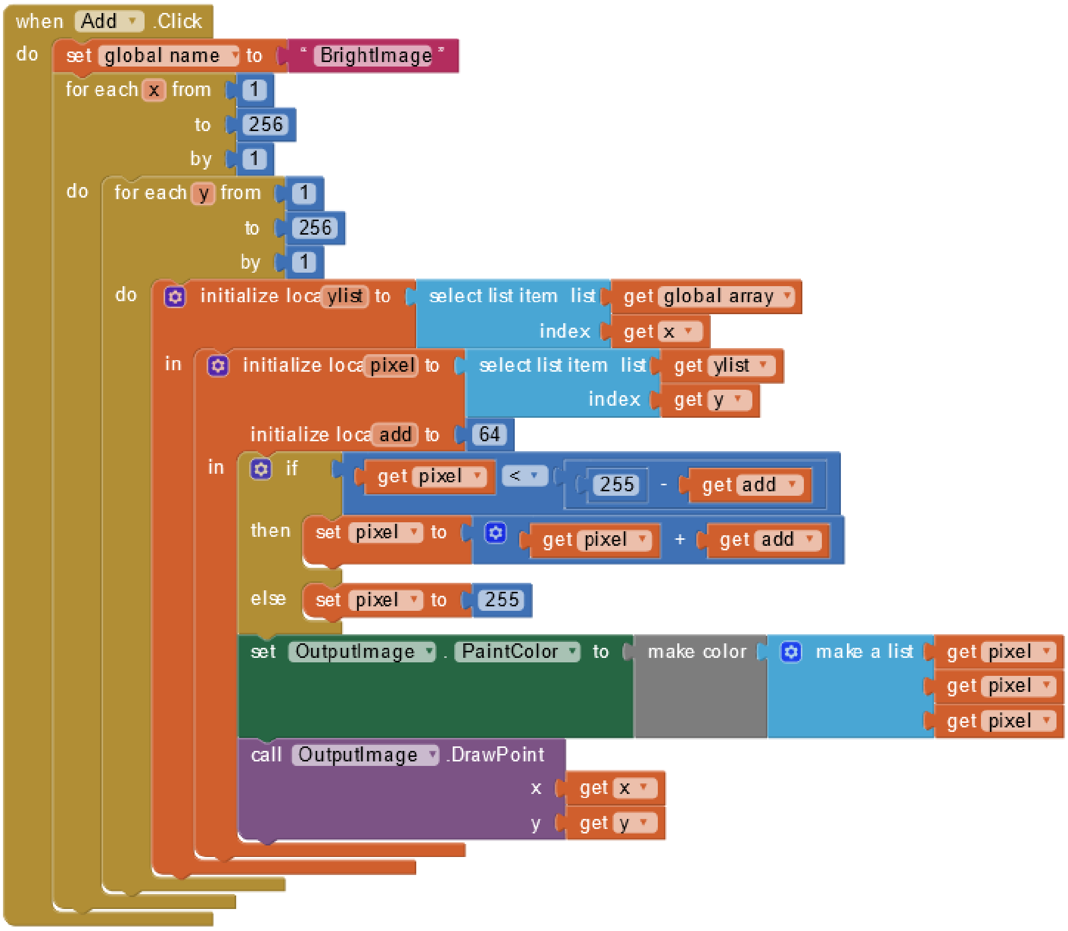
Simultaneously, the intensity of each pixel is read from the original image, with the function “GetPixelColor” and written into the predefined variable “array” by proceeding through two for-loops. After this the input image is saved as “OriginalImageN.jpg”, where N represents an ongoing number. The logic of this can be seen in the block snippet below.



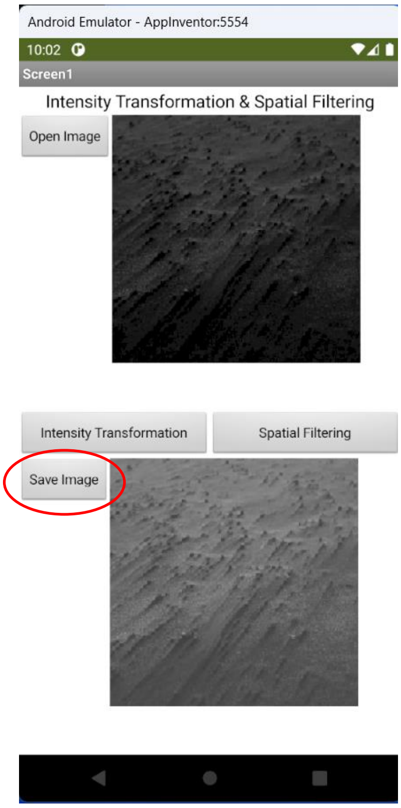
```
when Screen1 .Initialize
do  set global cnt to ( get global cnt + 1 )
    call InputImage .Clear
    set ImageData . Picture to get start value
    call OutputImage .Clear
    set ImageData . Visible to true
    for each x from 1
                to 256
                by 1
    do  initialize local ylist to create empty list
        in  for each y from 1
                       to 256
                       by 1
            do  initialize local color to select list item list split color call OutputImage .GetPixelColor
                                                                                                x get x
                                                                                                y get y
                                                          index 1
                in  add items to list list get ylist
                                      item get color
                    set InputImage . PaintColor to make color make a list get color
                                                                          get color
                                                                          get color
                    call InputImage .DrawPoint
                                           x get x
                                           y get y
            add items to list list get global array
                              item get ylist
    set ImageData . Visible to false
    evaluate but ignore result call InputImage .SaveAs
                                          fileName join " OrginalImage "
                                                        get global cnt
                                                        " .jpg "
```


Android Emulator - AppInventor:5554
10:02
Screen1
Intensity Transformation & Spatial Filtering
Open Image
Intensity Transformation
Spatial Filtering
Save Image

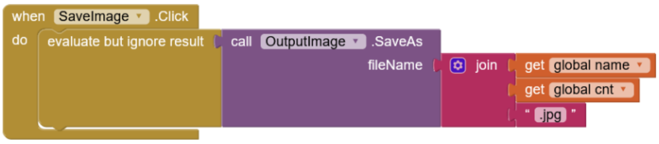
Right now, a pretty dark image is chosen and therefore the goal is to make it brighter. When clicking on the intensity transform, a brighter output image is created by adding a constant value, in this case 64, to each pixel. Again, each pixel is sampled with two for-loops, which proceed through the “array” and then 64 is added to each pixel, before the new intensity value is written into the output image. It is crucial to clip the intensity values to 255, in order to avoid errors. For example, if a pixel has the value 210 and 64 is added, resulting in 264, then the new intensity value would be wrong because the function “make color” interprets this as 264-255 = 9, which is very dark instead of white. The implementation of the clipping algorithm is achieved by a simple if-then-else condition. The name of the picture for later saving is set to “BrightImage”. The according block diagram is shown in the snippet below.



```
when Add .Click
do  set global name to " BrightImage "
    for each x from 1
                 to 256
                 by 1
    do  for each y from 1
                     to 256
                     by 1
        do  initialize loca ylist to  select list item  list  get global array
                                                        index  get x
            in  initialize loca pixel to  select list item  list  get ylist
                                                            index  get y
                initialize loca add to 64
                in  if    get pixel < 255 - get add
                    then  set pixel to  get pixel + get add
                    else  set pixel to 255
                    set OutputImage . PaintColor to  make color  make a list  get pixel
                                                                              get pixel
                                                                              get pixel
                    call OutputImage .DrawPoint
                                       x  get x
                                       y  get y
```

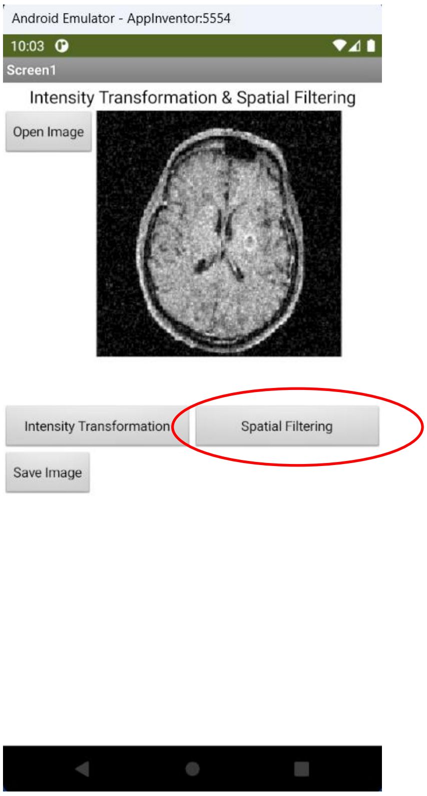

Android Emulator - AppInventor:5554
10:02
Screen1
Intensity Transformation & Spatial Filtering
Open Image
Intensity Transformation
Spatial Filtering
Save Image

The brightened output image is now displayed in the bottom right of the screen and the effects of the transformation are clearly visible. If wanted the image can be saved by clicking on the “Save Image” button. As shown in the snippet, the image is then saved as a .jpg with the previous defined name and the ongoing number.

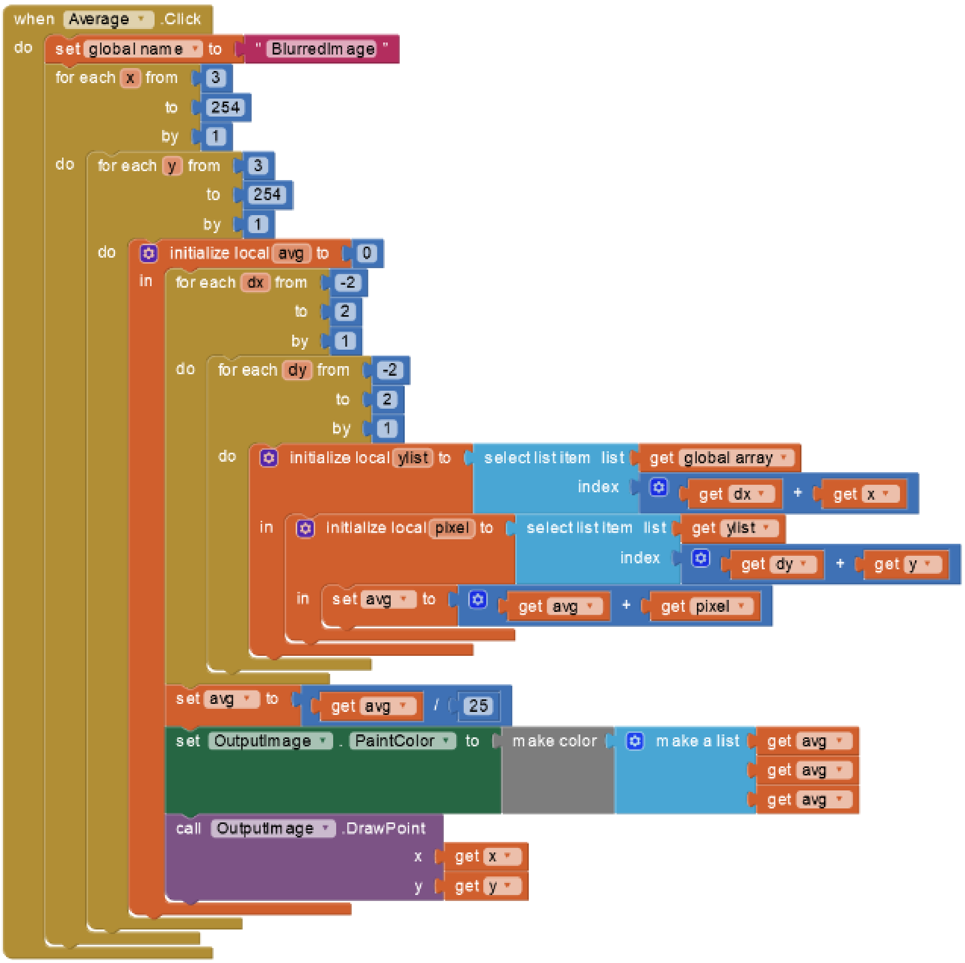


```
when SaveImage .Click
do  evaluate but ignore result  call OutputImage .SaveAs
                                     fileName  join  get global name
                                                     get global cnt
                                                     " .jpg "
```

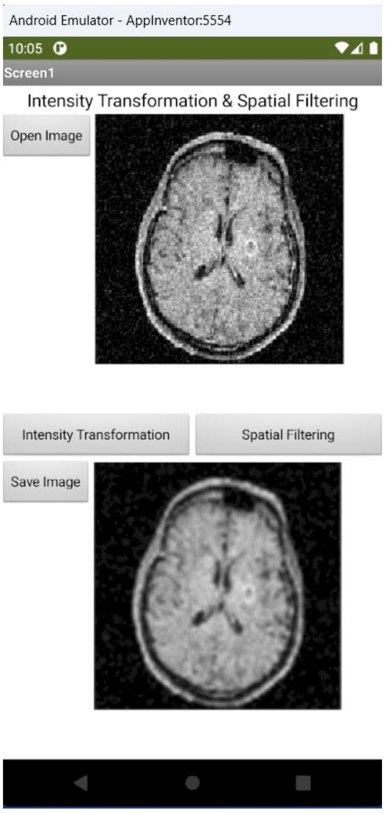
By clicking on the “Open Image” image button, it is possible to choose another image to process. This is described next. Alternatively, the spatial filter can be applied to the already selected image.


Android Emulator - AppInventor:5554
10:03
Screen1
Intensity Transformation & Spatial Filtering
Open Image
Intensity Transformation
Spatial Filtering
Save Image

Now, a MRI of the brain is selected, which is corrupted with gaussian noise. To get rid of this, an average filter is applied on the image. When clicking on the “Spatial Filtering” button the image will get smoothed with a 5x5 average filter. Notice that no padding is applied and therefore the two edge pixels will be ignored. Once again, the array is sampled with two for-loops, but this time for each pixel the neighbors are also sampled by proceeding through two more for-loops from -2 to +2. Then the intensity value of all these pixels is added together and divided by 25 to receive the average intensity. Here, the name of the image is set to “BlurredImage” for saving it later. The block sippet is shown below.

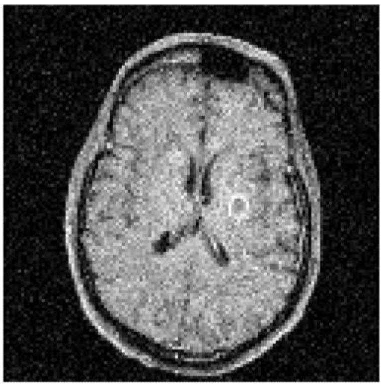
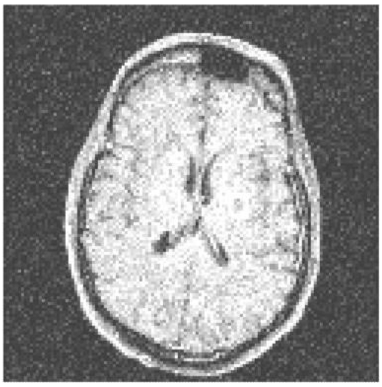
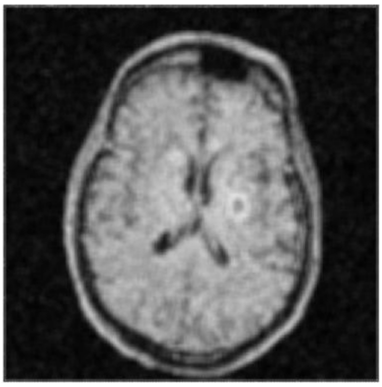
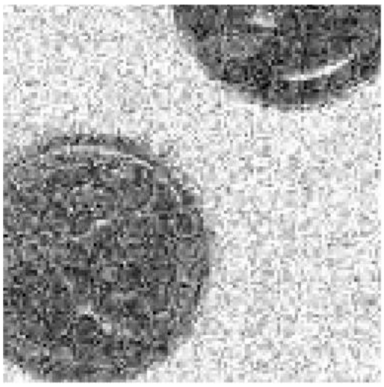
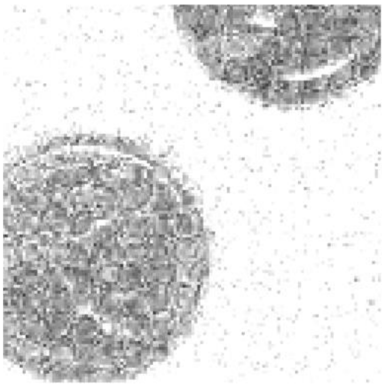
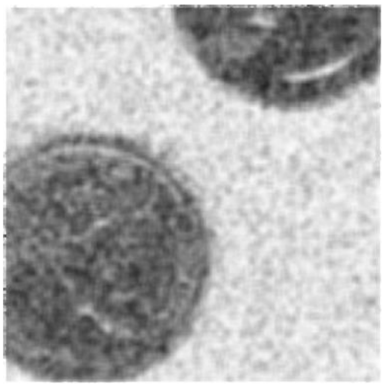


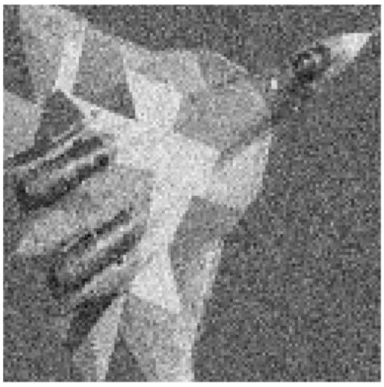
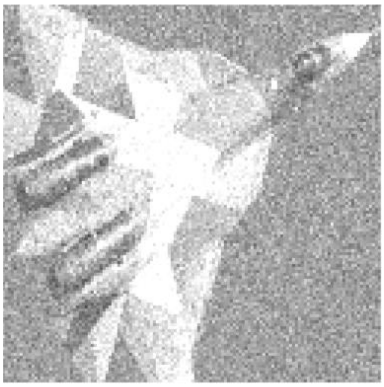
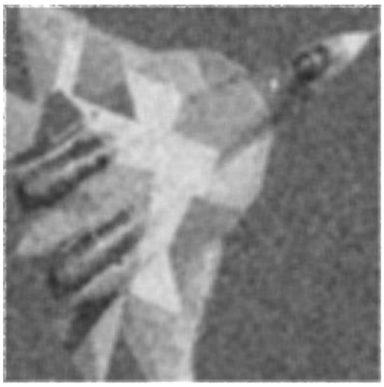
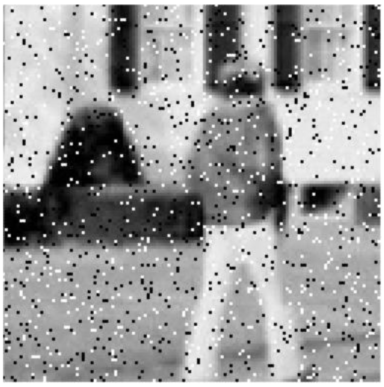
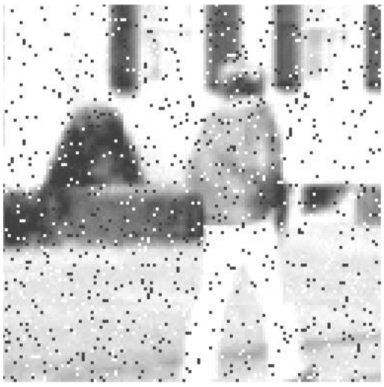
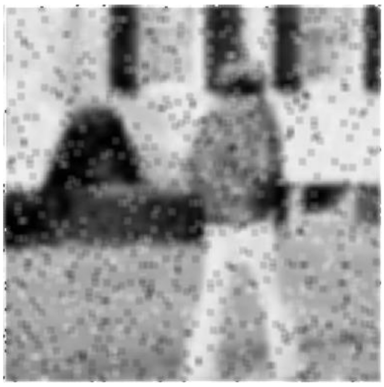
```
when Average .Click
do  set global name to " BlurredImage "
    for each x from 3
                 to 254
                 by 1
    do  for each y from 3
                     to 254
                     by 1
        do  initialize local avg to 0
            in  for each dx from -2
                             to 2
                             by 1
                do  for each dy from -2
                                 to 2
                                 by 1
                    do  initialize local ylist to select list item  list  get global array
                                                                   index  get dx + get x
                        in  initialize local pixel to select list item  list  get ylist
                                                                       index  get dy + get y
                            in  set avg to get avg + get pixel
                set avg to get avg / 25
                set OutputImage . PaintColor to make color  make a list  get avg
                                                                          get avg
                                                                          get avg
                call OutputImage .DrawPoint
                                         x  get x
                                         y  get y
```

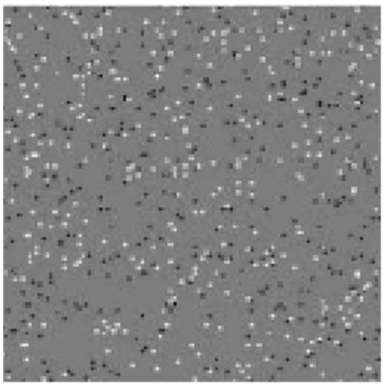
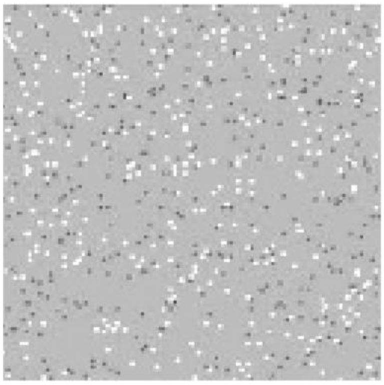
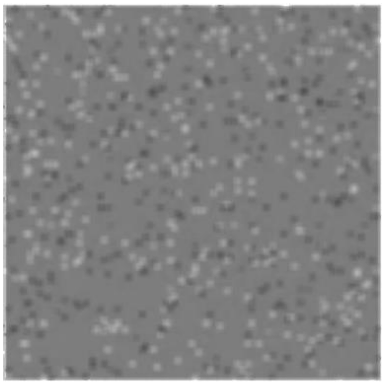
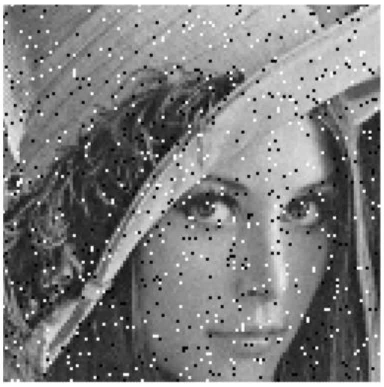
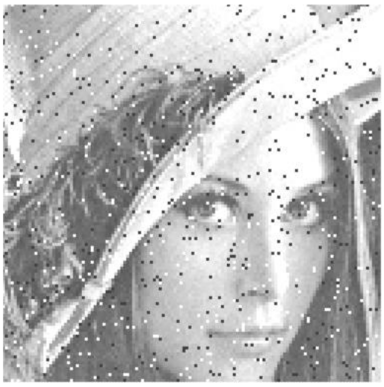
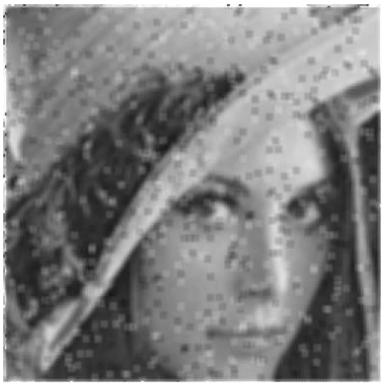
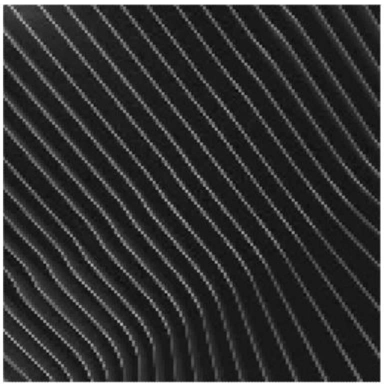
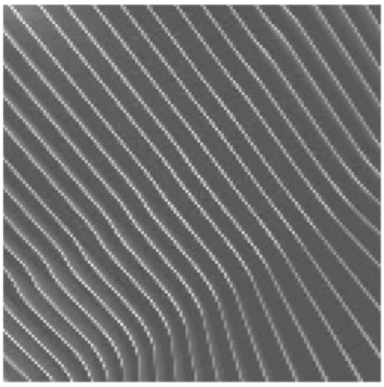
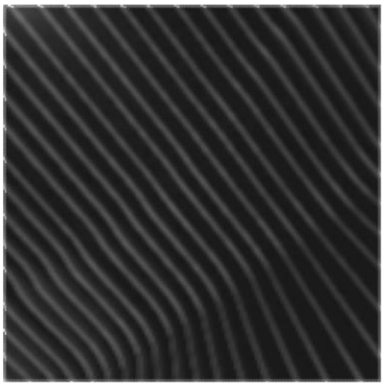

Android Emulator - AppInventor:5554
10:05
Screen1
Intensity Transformation & Spatial Filtering
Open Image
Intensity Transformation
Spatial Filtering
Save Image

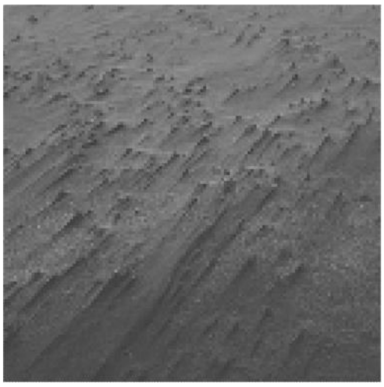
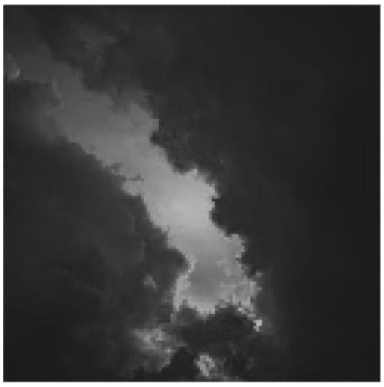
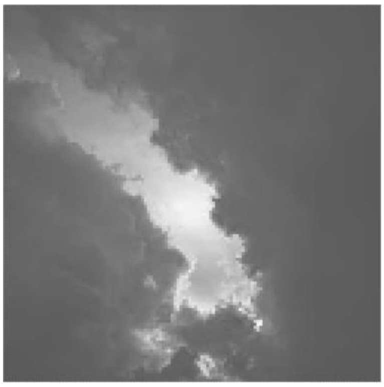
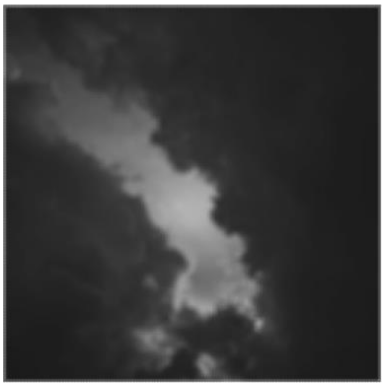
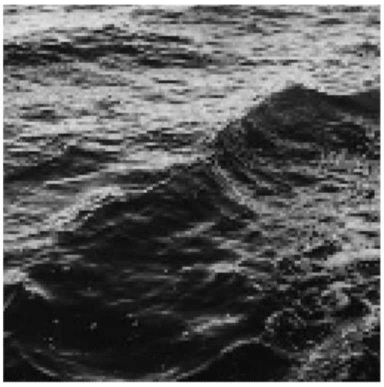
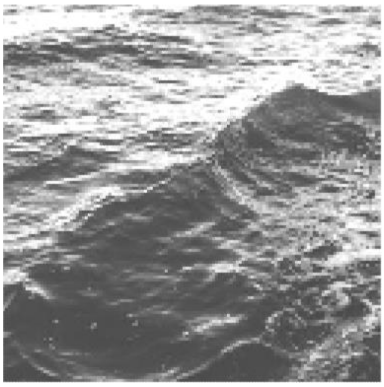
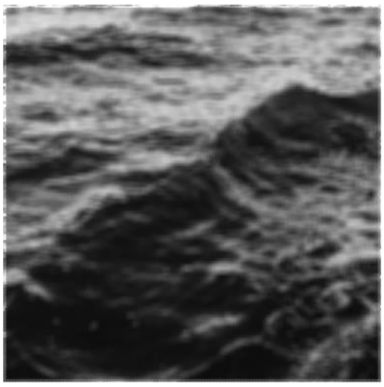
The result of the spatial filtering is displayed in the bottom right of the screen. The effects of applying an average filter can be seen. If wanted the image can be saved or another image can be opened.

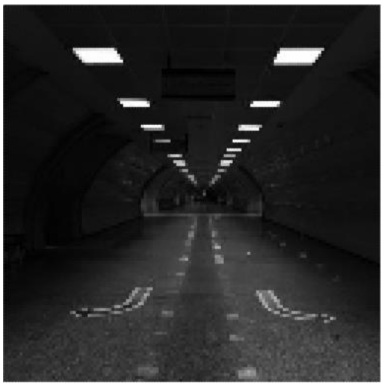
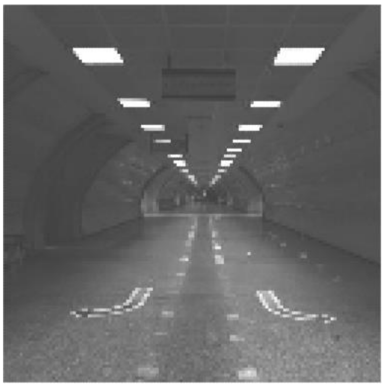
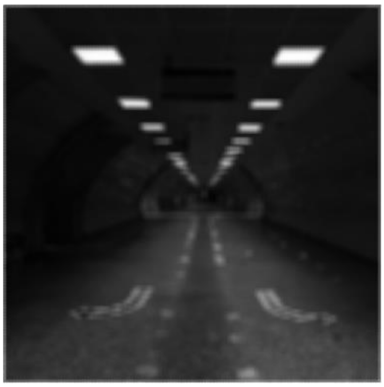
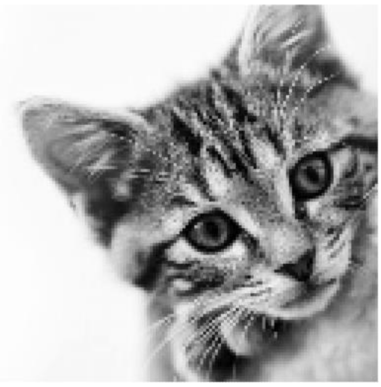
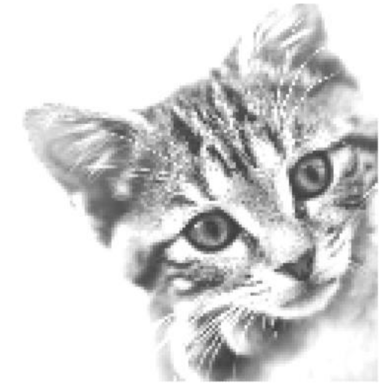
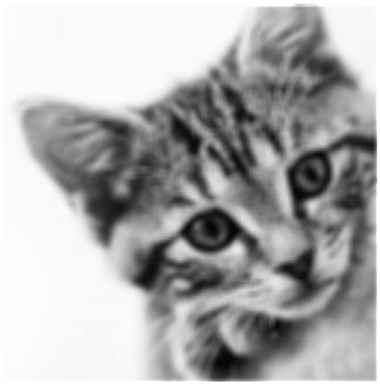
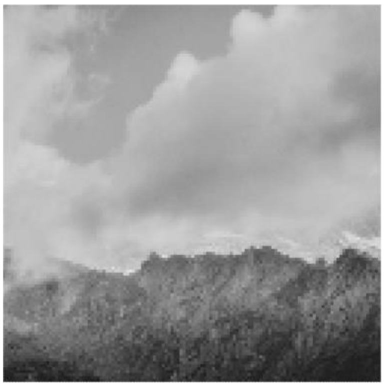
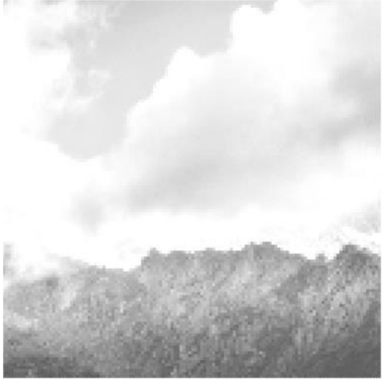
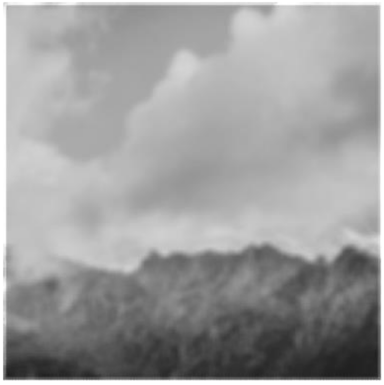
5. Picture Analysis

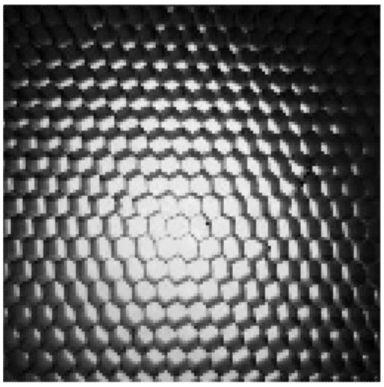
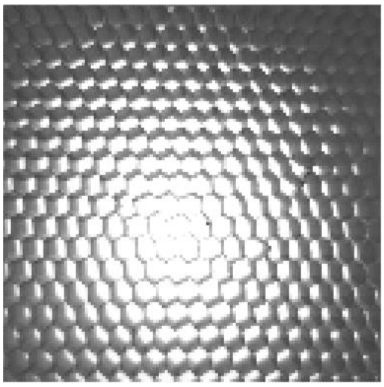
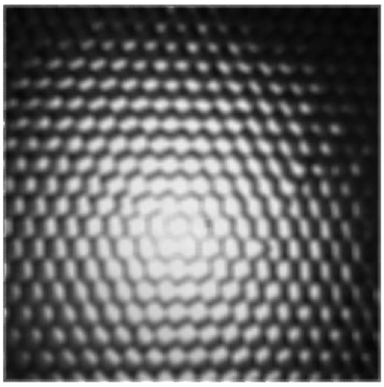
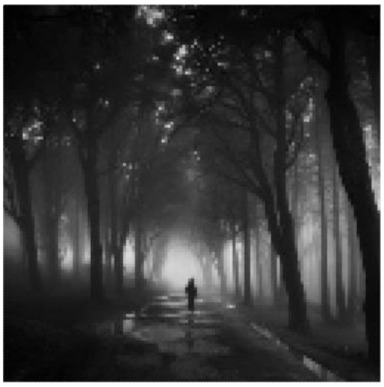
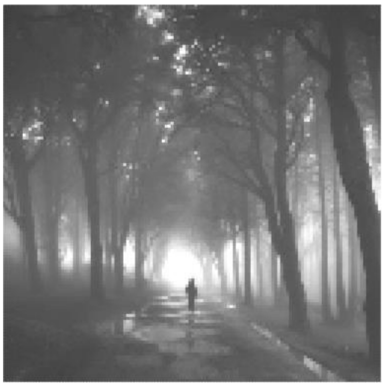
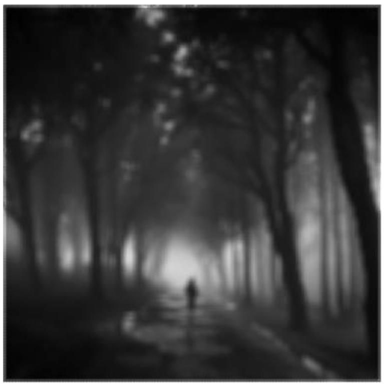
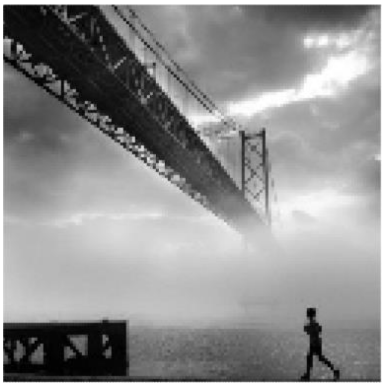
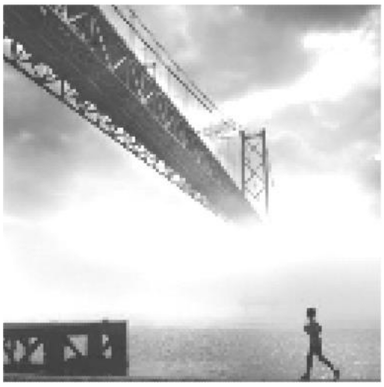
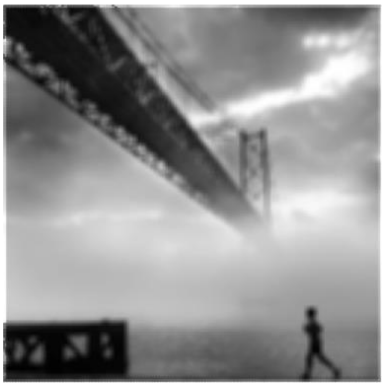
Original Image	Intensity Transformation	Spatial Filtering
		
This is a MRI scan of a brain. The problem with the pictures is, that it has noise.	The noise is still there. In the background, the noise even is more visible. The part of the barin is too bright and it is hard to recognize details.	The noise of the backgroud is almost gone. The brain is now smoother. Despite not having big noise anymore, it is harder to make up details.
		
This is a microscope picture. It is having problems with noise.	The picture did not need brightening. But in the background the noise got better because of the clipping, only salt and pepper remain. The objects did not improve.	The picture got smoother. The picture got improved. The issue is not gone completely.

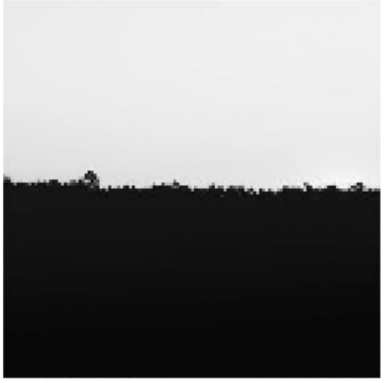
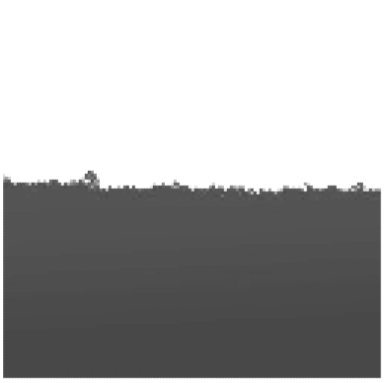
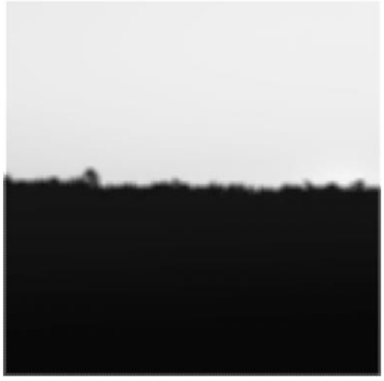
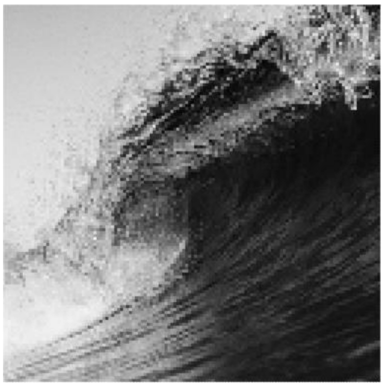
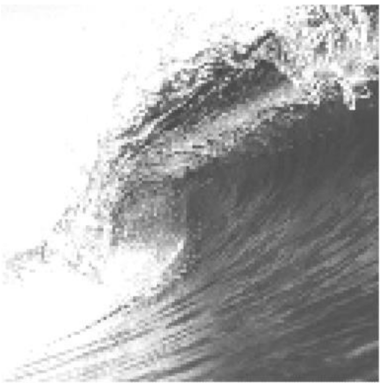
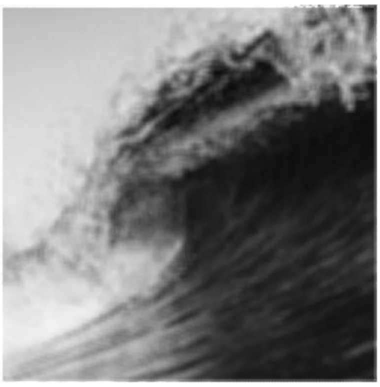
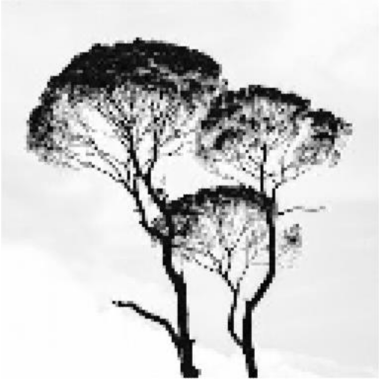
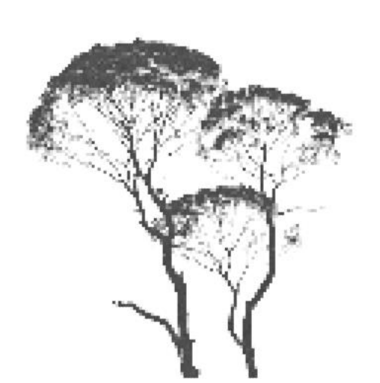
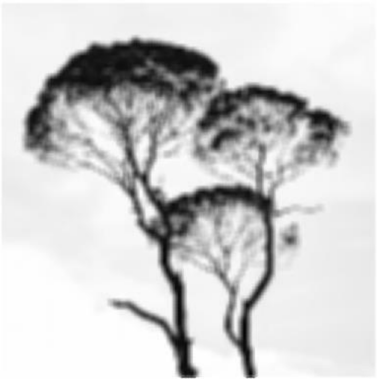
Original Image	Intensity Transformation	Spatial Filtering
		
This is a image of a fighter jet with noise.	The brightening has different effects on different parts of the picture. In darker areas, like the jet propulsion, it is now easier to make up details. In brighter parts, like the cross in the middle, its harder to make up details.	The image is smoother than the original one. Especially the background looks more like an entity. The smoothing comes at cost of some details like at the jet propulsion.
		
This image shows a man and a house. The issue of the picture is salt and pepper noise.	The brightening results in details loss. It is brightened too much. Only a little bit would have been enough.	The picture is quite blurry. The average filter trying to smooth the noise. But the salt and pepper noise is only weakened.

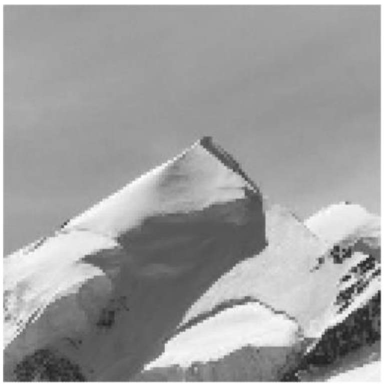
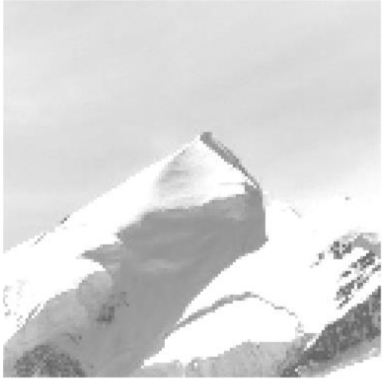
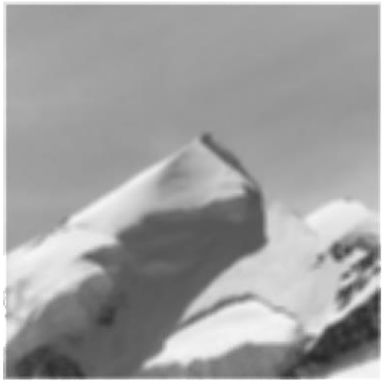
Original Image	Intensity Transformation	Spatial Filtering
		
This is a grey image with salt and pepper noise.	The brightening makes the image brighter.	The salt and pepper noise was hardly removed, but therefore the picture looks blurry.
		
This is a picture of a woman with salt and pepper noise.	More details are visible in the dark parts. Especially the bottom left corner it is easier to make up details.	The hard noise is smoothened. The salt and pepper effect is still visible. But the picture is smoother to look at.
		
This picture is a image of high constrast and high frequency components.	The black parts of this picture have been lightened. The white parts are in clipping. So the contrast could be lowered.	This makes the edges smoother. The picture does not look as pixeled any more.

Original Image	Intensity Transformation	Spatial Filtering
		
This is a close taken picture of sand. It is lacking detail, because it is to dark.	It is now possible to see the details of the picture. It is visible the some sand grains form little mountains which have their own shadows.	The picture is still dark and details are too blurry.
		
This is a picture of the sky. Here one can see some white part with dark background.	After the brightening, it is visible that in the sky are clouds with a gap inbetween them.	The dark picture is now smoothened, which makes it harder to spot details.
		
This is a picture of the ocean.	This is a brighter ocean. Especially in the buttom left corner details can be seen.	This is a blurry ocean.

Original Image	Intensity Transformation	Spatial Filtering
		
This is a picture a tunnel, which makes it hard to see in the darkness. It is also quite pixeled.	Thanks to the brightening, it is possible to see the shape of the background of the tunnel. This is a good example for what the transformation was designed.	The pixeled parts are smoothened. But is also harder to make up details like the arrows on the surface.
		
This is a picture a sweet cat. This example shows the wrong use of transformation.	The picture is to bright. It is losing details in parts where the picture arlready was bright.	The soomthening lost the picutre some details.
		
This is a picture of mountains, which are quite dark and a cloudy sky.	The mountain section is gaining details. The clouds directly obvove loose some to clipping.	The picture got blurry.

Original Image	Intensity Transformation	Spatial Filtering
		
The picture shows hexagons, which have high contrasts and high frequency components.	The brightening makses the edges better visible. For the middle, contrast is lost.	The edges between the hexagons are more blurry.
		
This is a picture of a person in the woods The problem is, that it is too dark to see every detail.	It is now easier to spot details, like trunks. The left side still is a bit to blurry and dark to see.	The picture got too blurry.
		
This is a picture of a bridge. It has many intensity levels.	The birghtening leads to clipoing in the sky. Ths picture is loosing detials.	No smoothining is needed, it is less visible now.

Original Image	Intensity Transformation	Spatial Filtering
		
This is a horizon with woods on in the bottom part. It is not possbible to spot details as it is to dark and pixeled.	The brightening doses not realy unvail details as the original image they got clipped of into one black value.	The pixeling is hard to remove for this filter, as it has to deal with black and nearly white. The picture still got smoother.
		
This a picture of a wave. It has some issues with contrast.	The brighening can only attend it partially. The water itself gains details. The background looses the image some details.	The picture gets blurry.
		
This is a picture of trees. The pictures has high contrasts.	Clipping takse some of the details as the picture got brightened too much.	The high contrast lead to a too strong smoothening and blurring.

Original Image	**Intensity Transformation**	**Spatial Filtering**
		
This picture shows a moutain. As it is covered in snow, the picture is quite bright.	Brighening the picture even more up, leads to loss in constrast and details.	Edges getting smootened. The picture becomes blurry.

6. References

Gonzalez, R. and Woods, R., *Digital Image Processing*, 4th edition, 2018

Gupta, B. and Agarwal, T. K., *New contrast enhancement approach for dark images with non-uniform illumination*, Jabalpur, Elsevier, 2017

T., Wang, J. and Yao, K., *Visibility enhancement of underwater images based on active polarized illumination and average filtering technology*, Beijing, Elsevier, 2021

Hussain, K. and Others, *A histogram specification technique for dark image enhancement using a local transformation method*, Dhaka, Springer Open, 2018

H. Abelson, *University page-Prof. Hal Abelson*: https://www.csail.mit.edu/person/hal-abelson, 03.06.2023

MIT Inventor, *Reference Documentation*: http://ai2.appinventor.mit.edu/reference/, 03.06.2023