# Algorithm and Design Considerations

## Filtering & Down-sampling

Processor is designed to down sample an image. Down sampling is the reduction in spatial resolution while keeping the same two-dimensional (2D) representation. To down sample following steps are to be followed.



### Low Pass Filter

Prior to down sampling an image, it should be low pass filtered to reduce the effects of aliasing caused by the high frequency components in the image. For the filtering purpose, there are mainly two types of low pass filtering kernels.



Although it is easy to perform addition using normalizing kernel, as it should be divisible by 9, we are not going to use this kernel for low pass filtering.

LightboxIn Gaussian Kernel, shifting is needed to be done before addition. 16 is a number which is a power of 2, (16=24). So, it can be easily shifted when we use Gaussian Kernel. The filtering algorithm is based on the 3x3 Gaussian kernel where the middle pixel location is considered as (0,0) and the standard deviation is 1 in the below equation and then approximating it to discrete numbers.

Table

Description automatically generated

The Gaussian filtering can be used to blur the images and to remove the noise present. Here a 3×3 matrix is used mainly because 99% of the distribution fall within 3 standard deviations. The central pixel in Gaussian kernel has a higher weighting than the others on the sides.

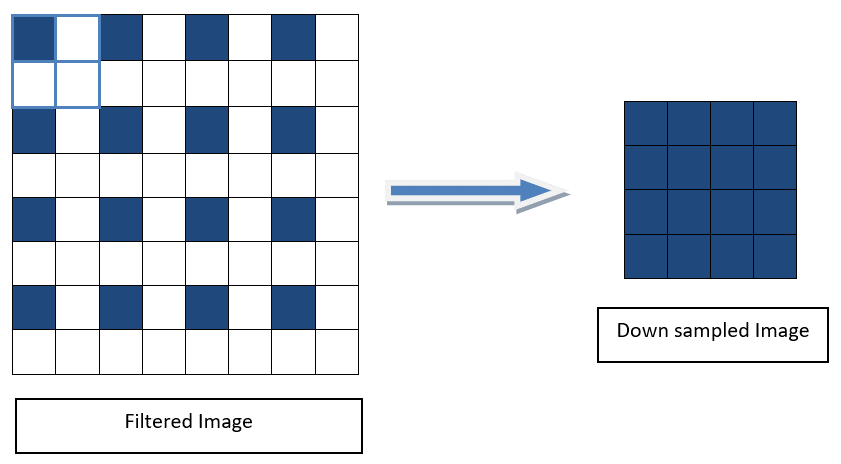
Now this center pixel is overlapped with the image pixel and a weighted summation value is calculated and it is divided by 16, which is the total weight of the kernel.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | .. |  |  |
|  |  |  |  |  | .. |  |  |
|  |  |  |  |  | .. |  |  |
|  |  |  |  |  | .. |  |  |
|  |  |  |  |  | .. |  |  |
| .. | .. | .. | .. | .. |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

The kernel will calculate the average value first and stores it in the red colored pixel box. Then, it will move to the right to calculate the next average value. As per the figure it will move to the blue colored pixel box. After calculating all the average values in the right and storing it in that pixel, the kernel will move to the next row, which is marked in the yellow color. Same as earlier it will move right and will come to next row to filter all the pixel values. Finally, it will end up in the green colored pixel box. The gray colored pixel boxes remain unfiltered as it will not a big concern for down sampling.

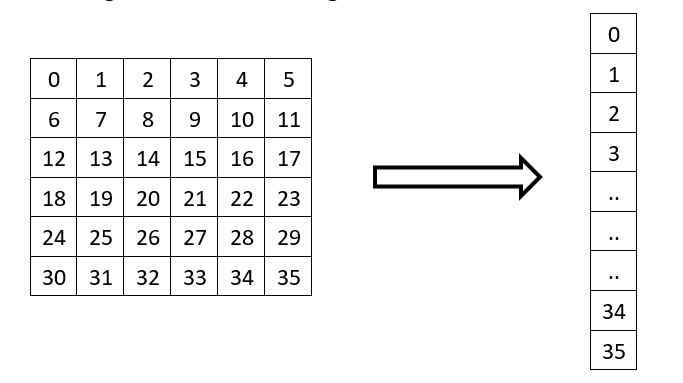
### Down-sampling

Down sampling reduces the number of pixels in image by a given down sampling factor. Here, the objective is to down sample an image by a down sampling factor 2. We are going to consider an image with the size 256×256. So, the resultant image size is 128×128. For that we are going to consider one value per four pixels from the above filtered image.



## Data Processing Techniques

The very first step of our task is to store the data included in the pixels of the 256x256 image into a text file. This can be done using the python code ‘input\_create.py’. All the pixel values of the image are transformed to an array and the binary values of them are written in a text file. Data arrangement in the image and the text file is given below.



Then for the simulation of filtering and down sampling, the binary values in the text file are used as the input. From the simulation we can obtain the filtered output text file or the down sampled output text file, from considering the operation.

By using this filtered and down sampled text files as data files and by executing the ‘show.py’ python code, the binary values in the text files are stored in an array. After that this array is transformed into a matrix and then the filtered and down sampled images are obtained.

For the evaluation of the performance of our processor, a python code is used for filtering and down sampling. Here a comparison is made between the image data arrays of down sampled images from the designed processor and the python code execution by taking their difference. The error calculated from the above procedure is zero, which concludes that the designed processor is as per our expectations. Below images show the results from above mentioned methods.

A screenshot of a person in a hat

Description automatically generated with low confidence

Original, filtered and down sampled images from the processor

A person wearing a hat

Description automatically generated with low confidence

Original, filtered and down sampled images from the python code implementation