KWANGWOON UNIVERSITY

Image Upsampling

2018.9.18

Seoungjun Oh(sjoh@kw.ac.kr)
Wooju Lee (krosea@kw.ac.kr)

Multimedia LAB
VIA-Multimedia Center, Kwangwoon University

Contents

- Image Upsampling
- Peak Signal to Noise Ratio(PSNR)
- Example
- Assignment

Image Upsampling

- Upsampling
  - Process of increasing the sampling rate of a signal
  - Upsampling an image means increasing the resolution of the image

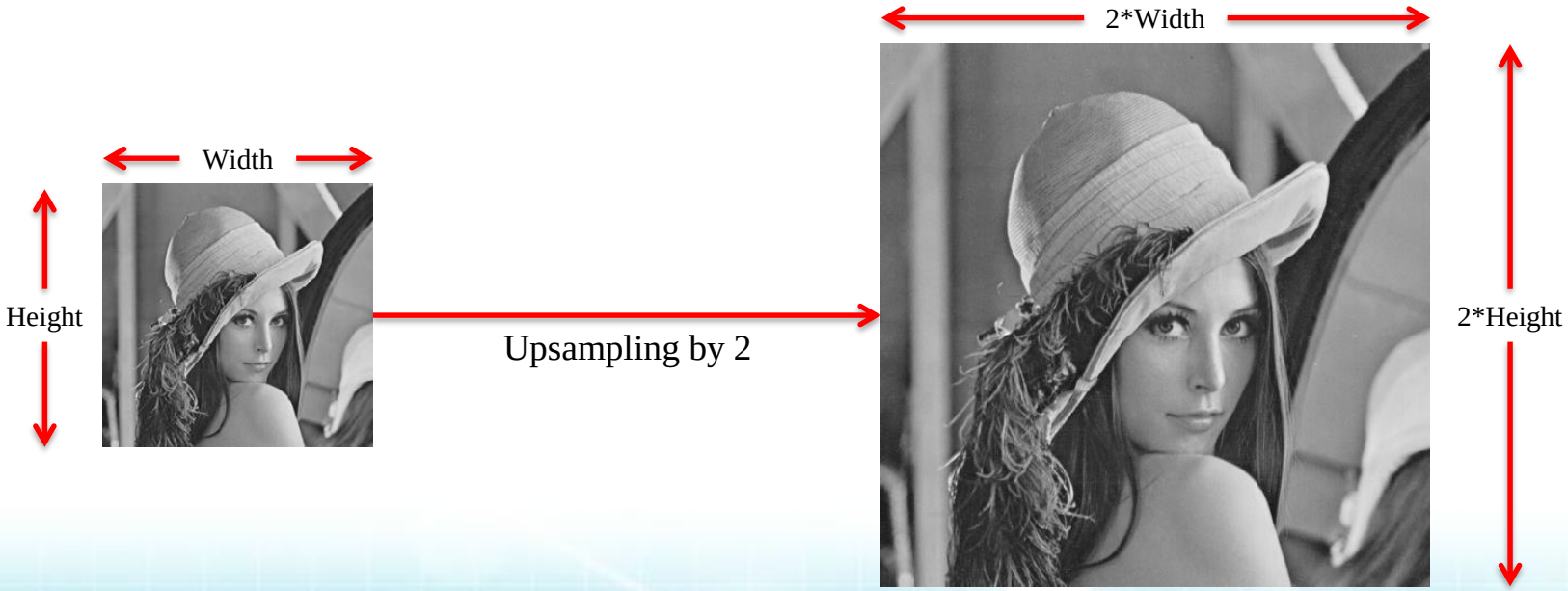


Image Upsampling

- Image upsampling
  - Interpolation method
    - Copy
    - Average
    - Interpolation filter

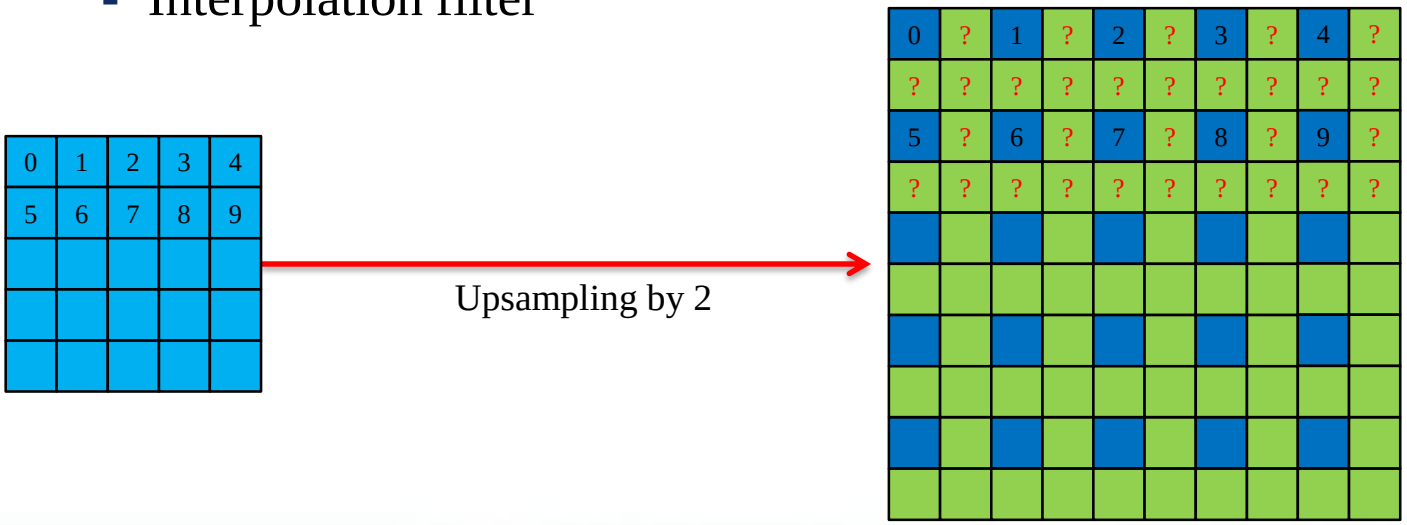


Image Upsampling

- Image upsampling
  - Interpolation method
    - Copy

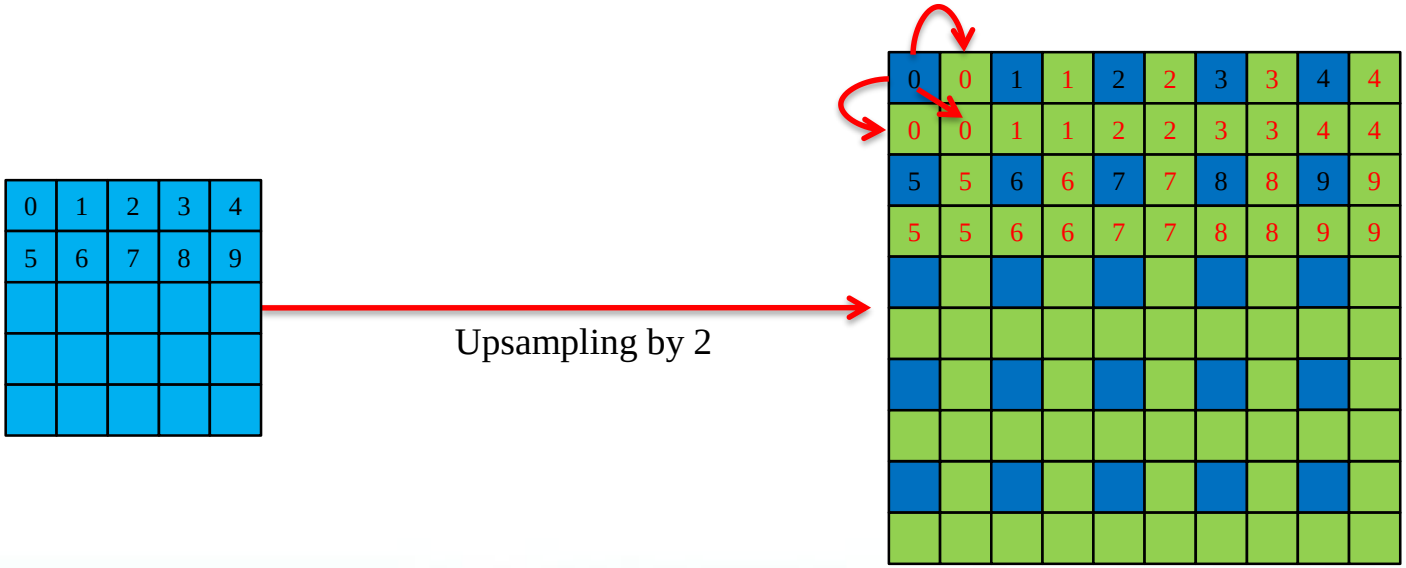


Image Upsampling

- Image upsampling
  - Interpolation method
    - Copy

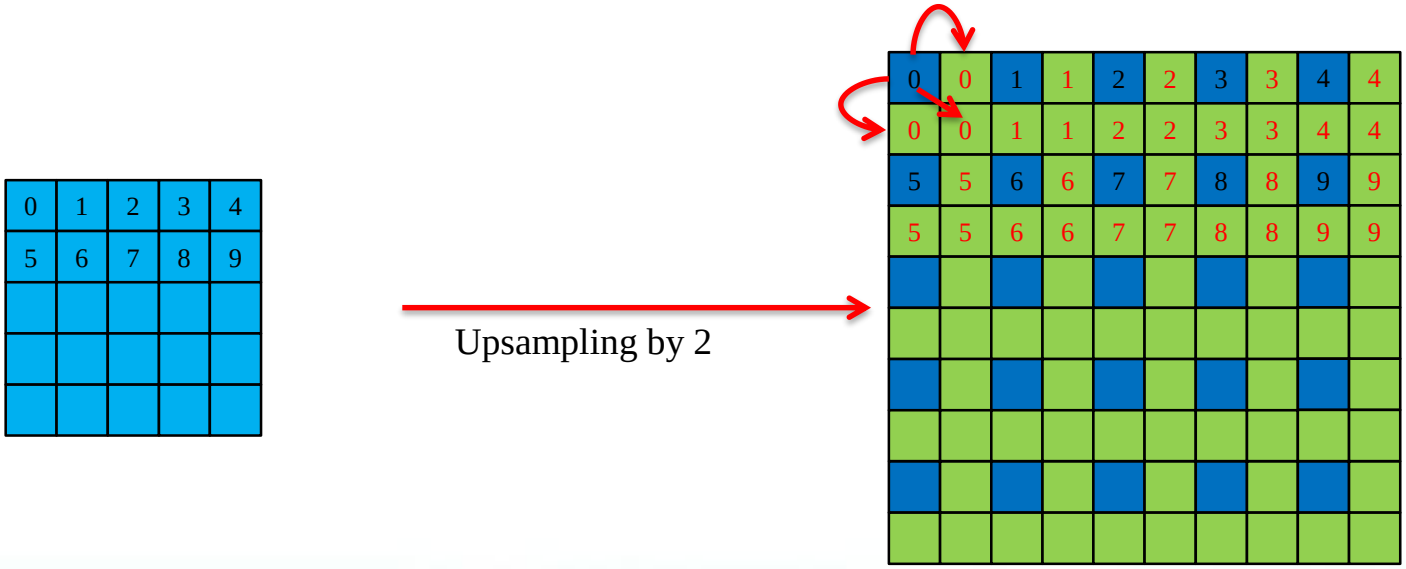


Image Upsampling

- Image upsampling
  - Interpolation method
    - Average

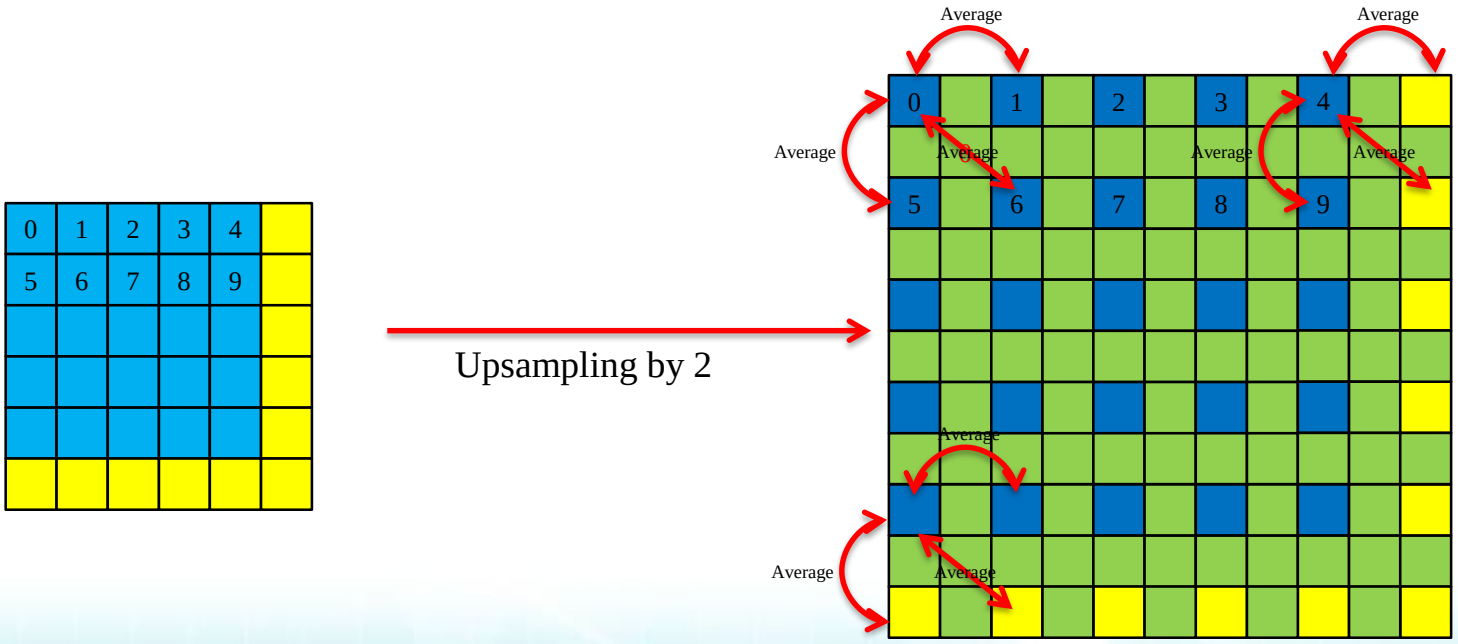


Image Upsampling

- Image upsampling
  - Interpolation method
    - Average

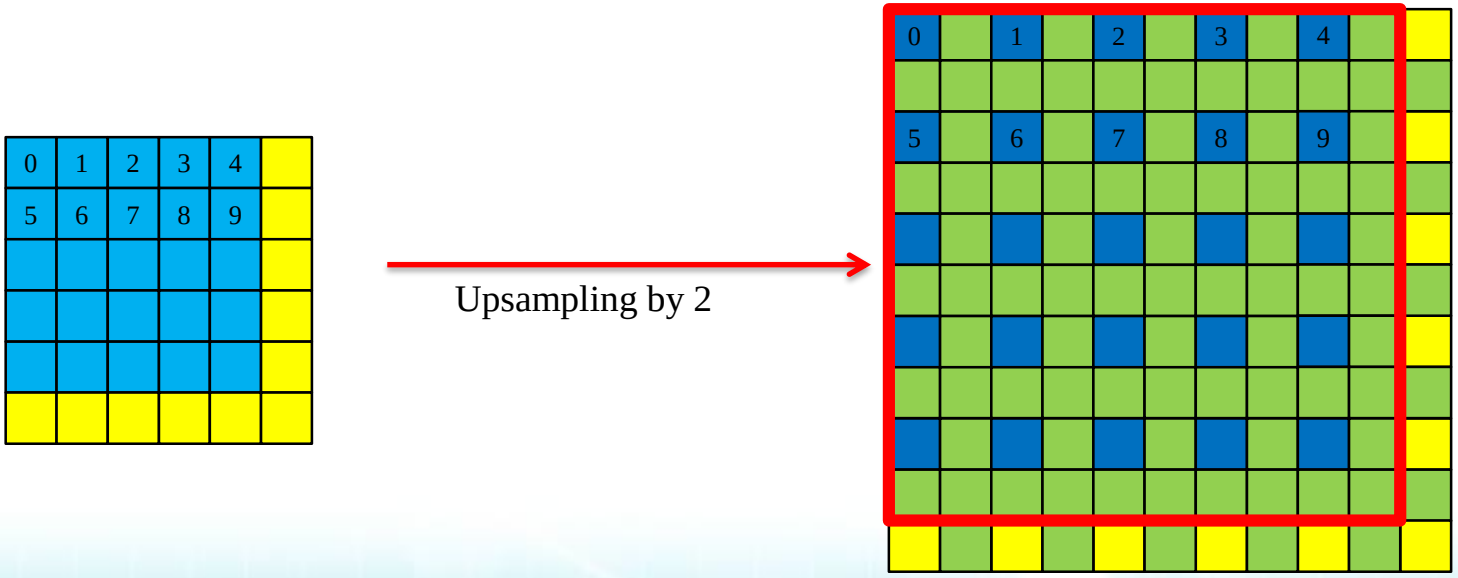


Image Upsampling

- Image upsampling
  - Interpolation method
    - Interpolation filter

Image Upsampling

- Image upsampling
  - Interpolation method
    - 8-tap interpolation filter
      - {-1, 4, -11, 40, 40, -11, 4, -1}
      - $\frac{1}{64}\sum$

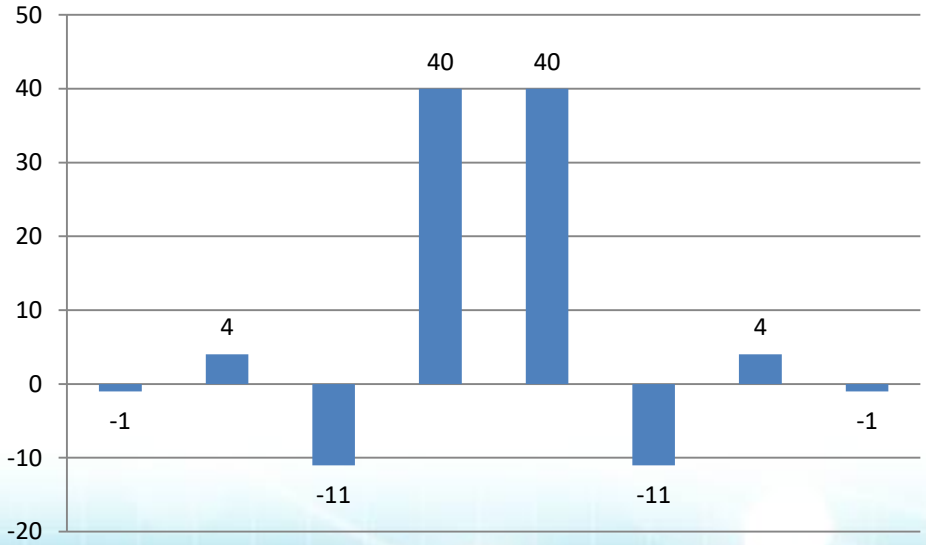
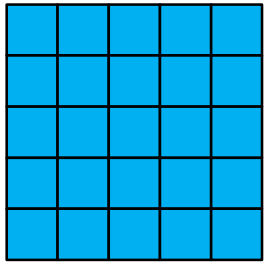


Image Upsampling

- Image upsampling
  - Interpolation method
    - Padding



< Original image >

Upsampling by 2

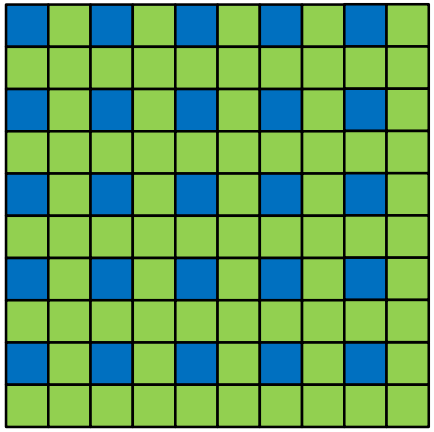
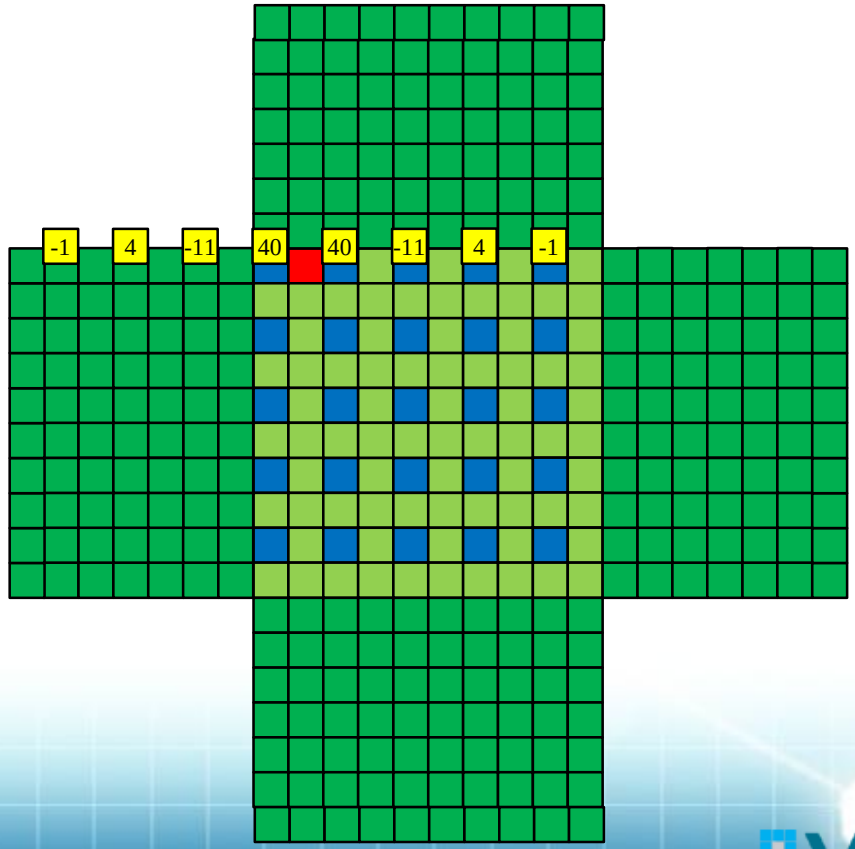


Image Upsampling

- Image upsampling
  - Interpolation method



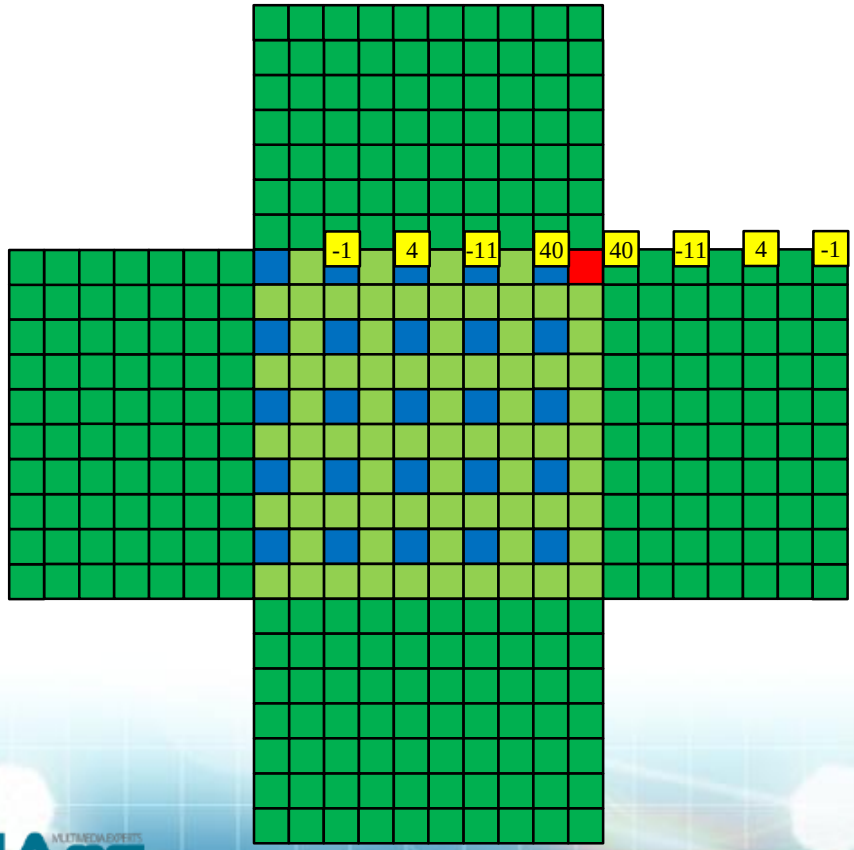
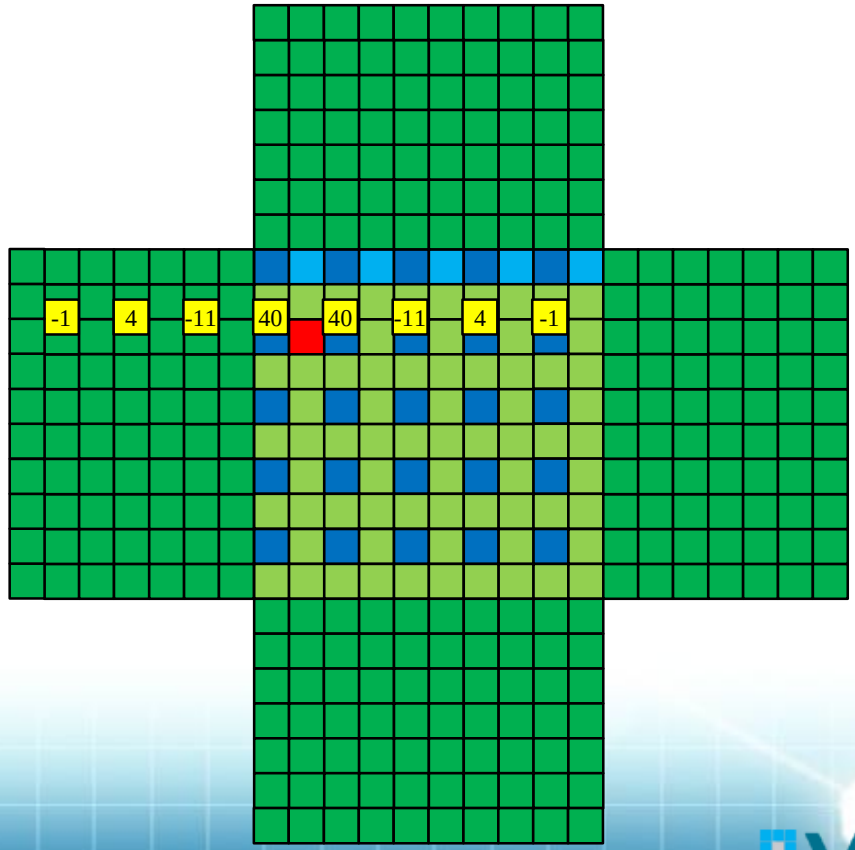


Image Upsampling

- Image upsampling
  - Interpolation method



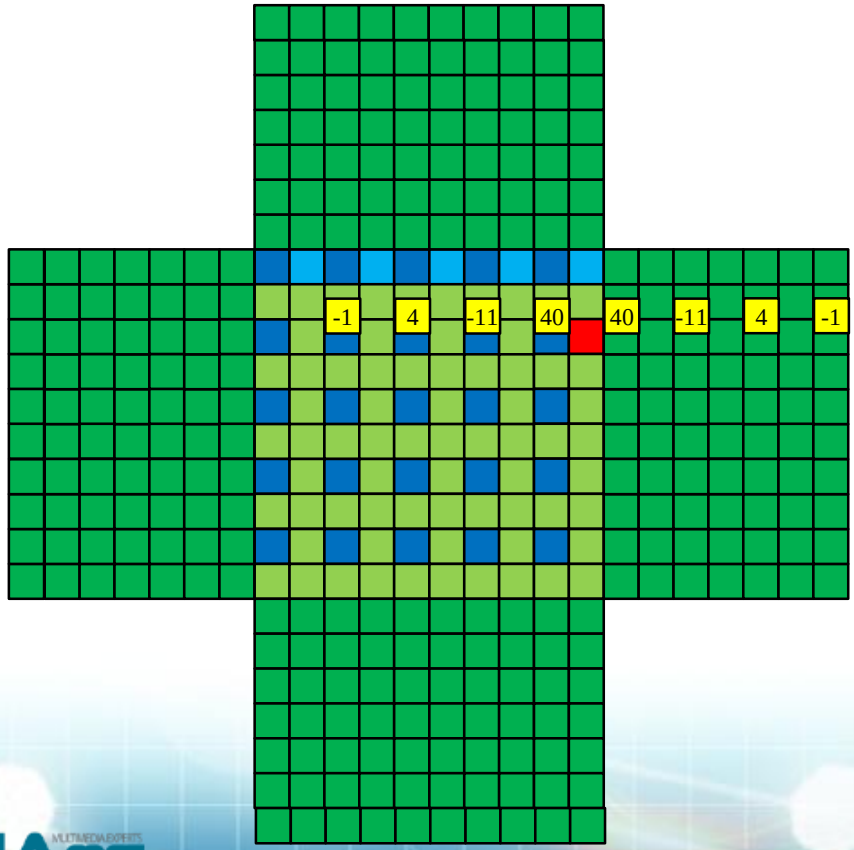
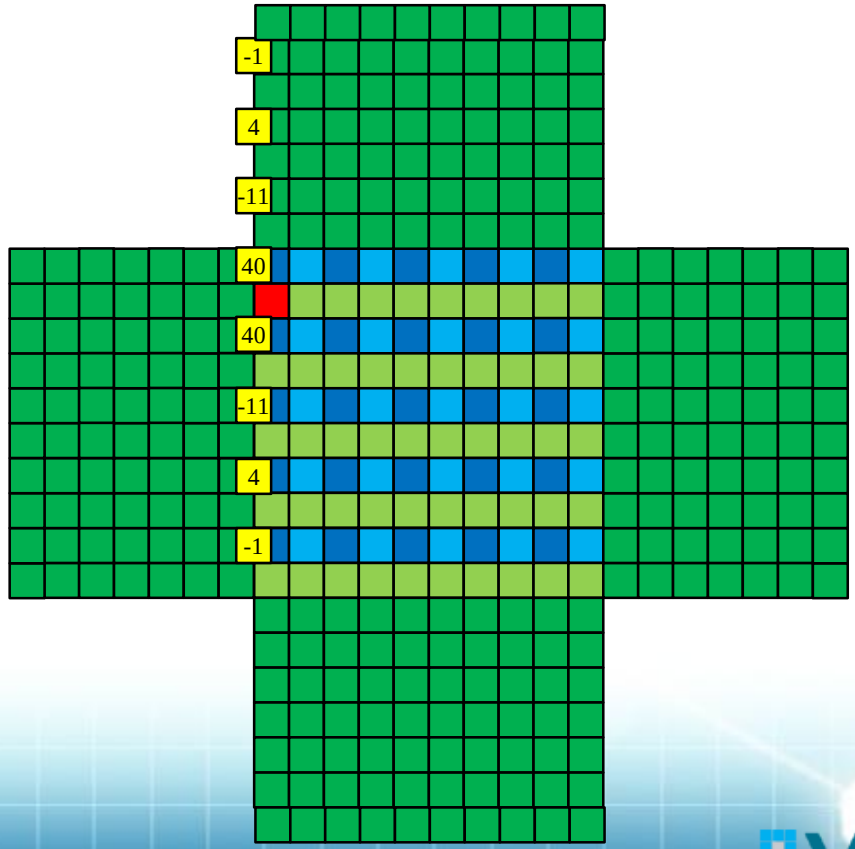


Image Upsampling

- Image upsampling
  - Interpolation method



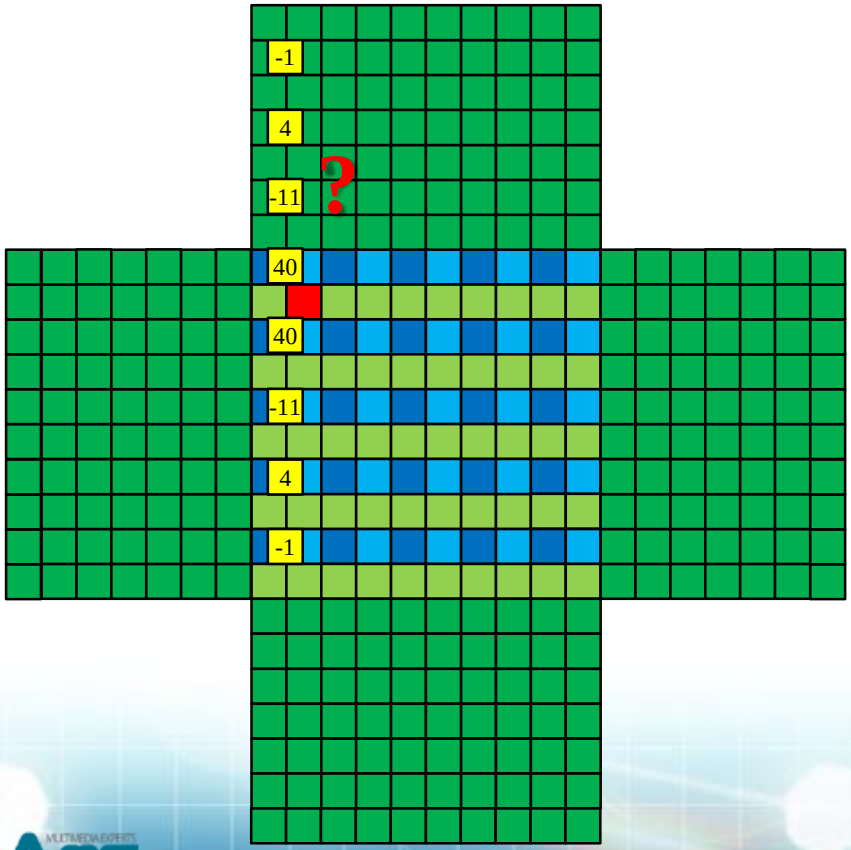
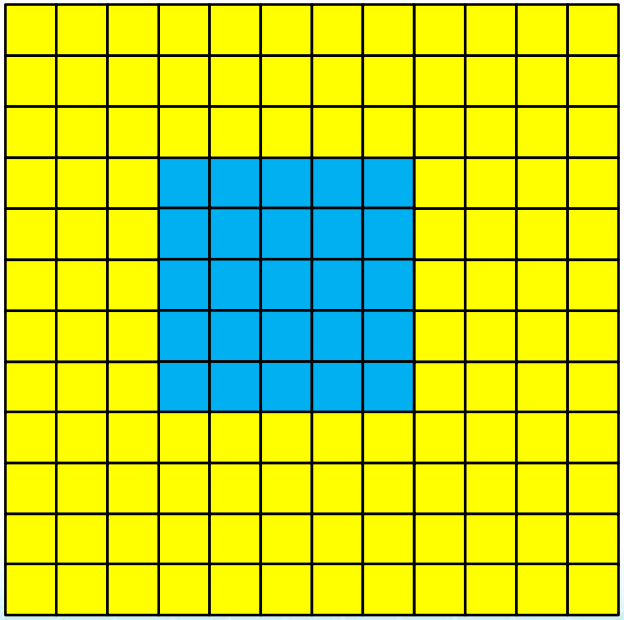


Image Upsampling

- Image upsampling
  - Interpolation method
    - Padding



< Original image >

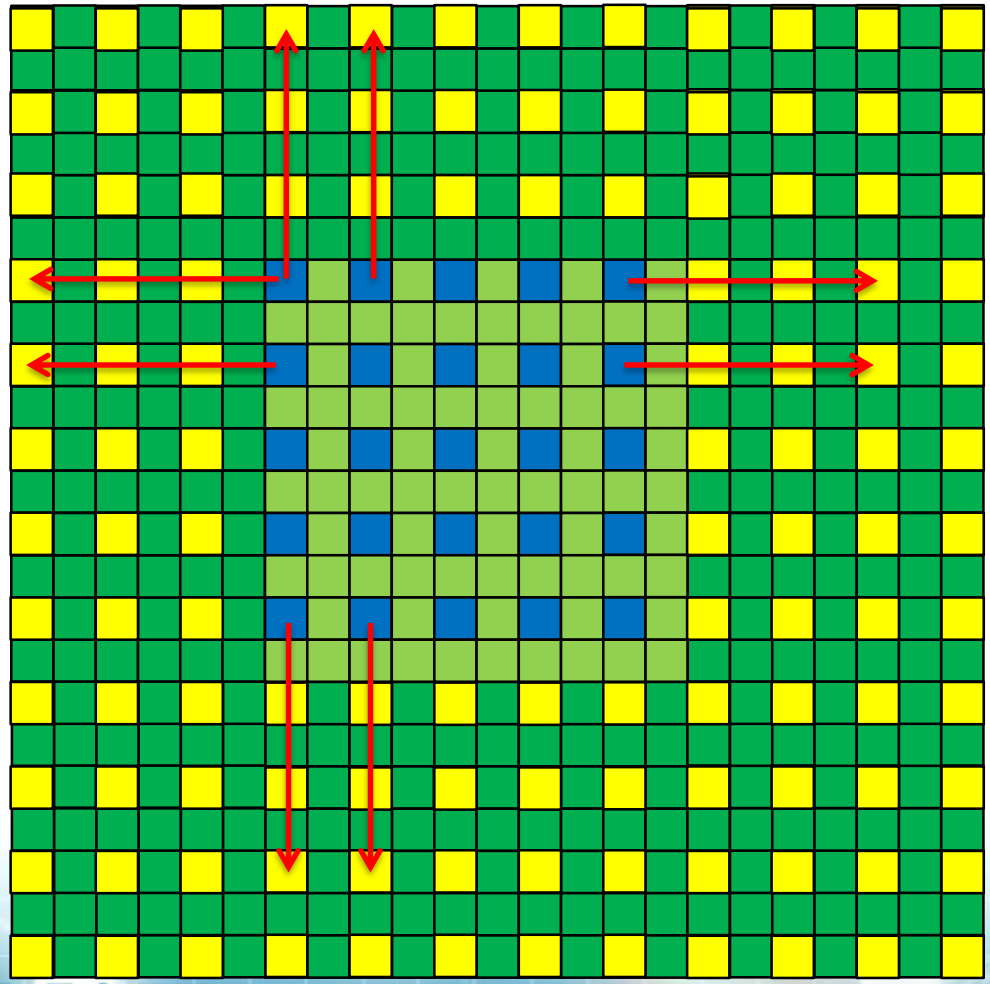


Image Upsampling

- Image upsampling
  - Interpolation method

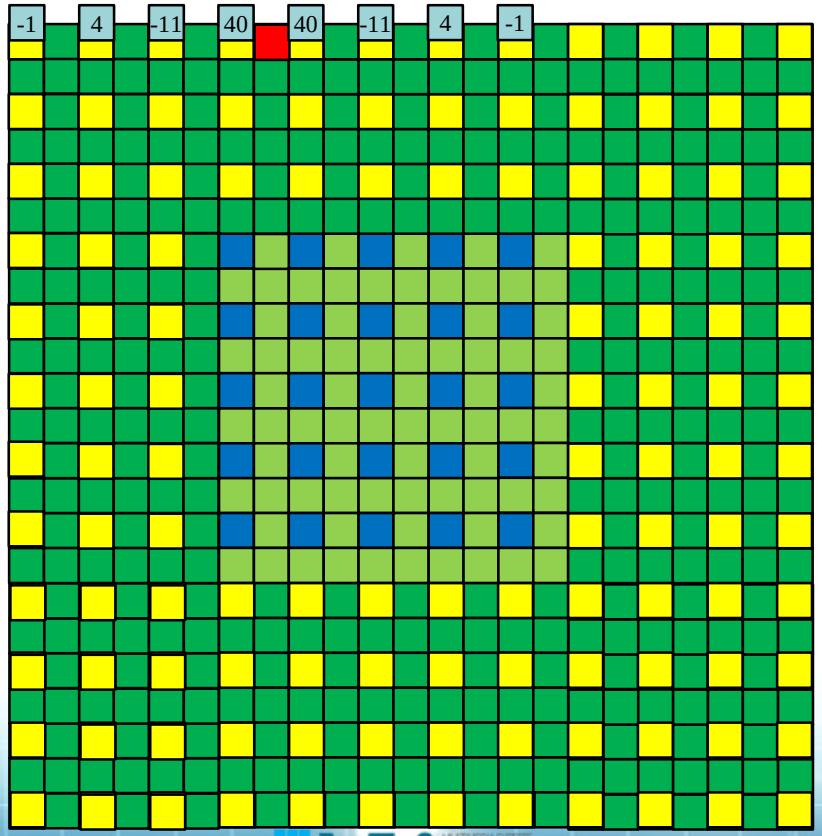


Image Upsampling

- Image upsampling
  - Interpolation method

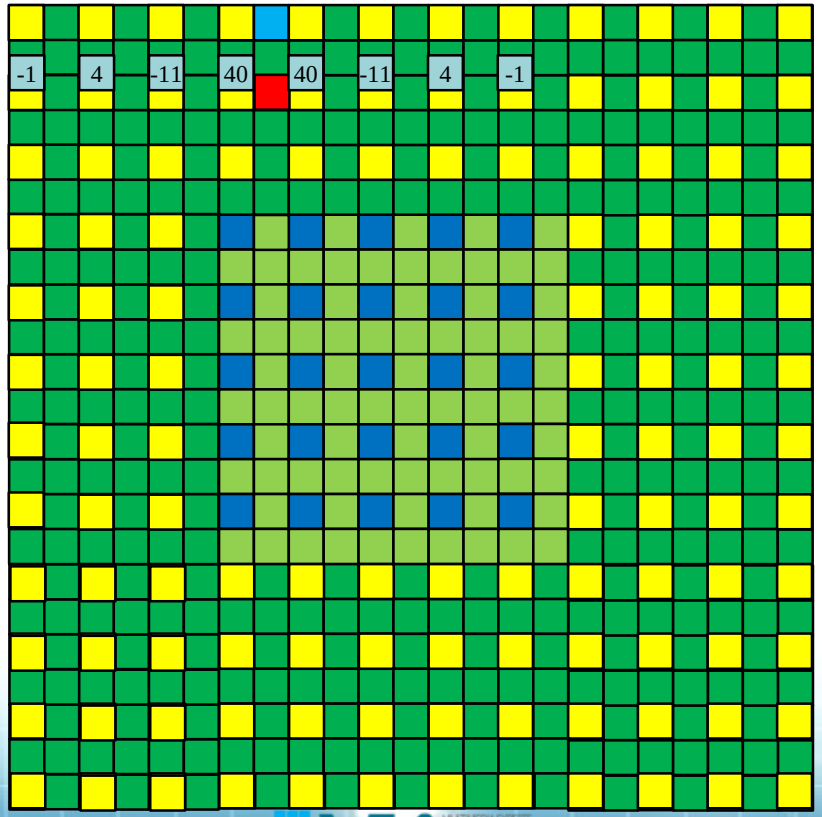


Image Upsampling

- Image upsampling
  - Interpolation method

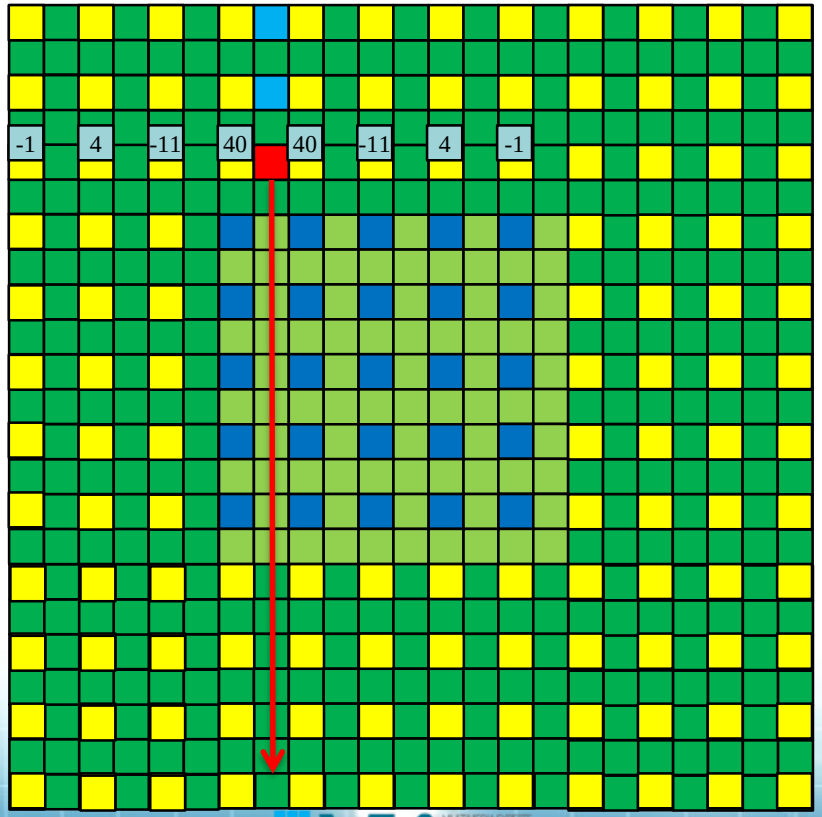


Image Upsampling

- Image upsampling
  - Interpolation method

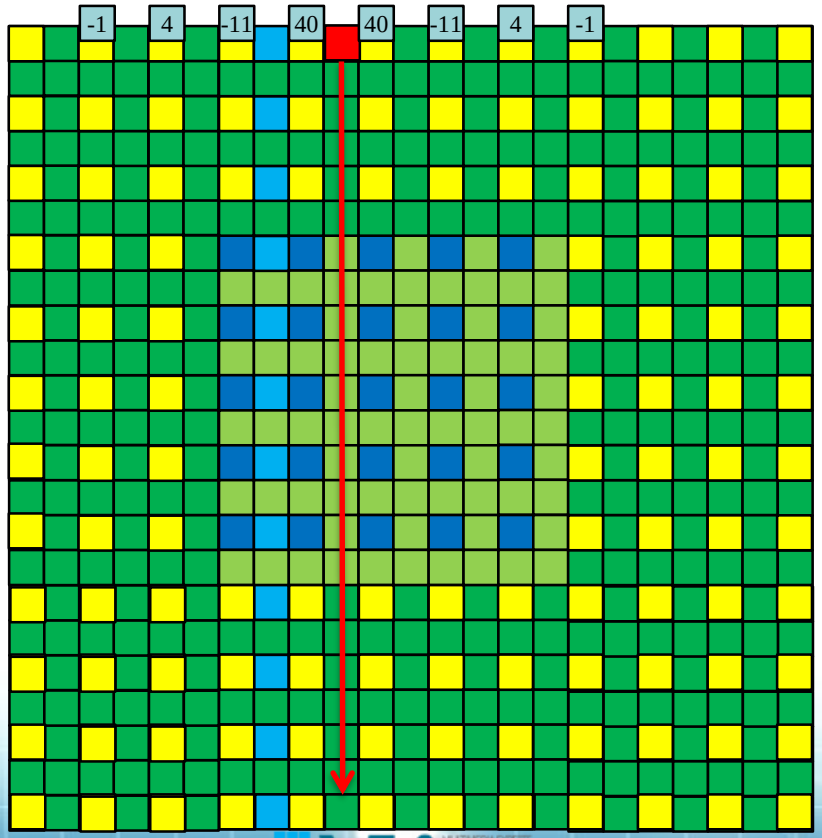


Image Upsampling

- Image upsampling
  - Interpolation method

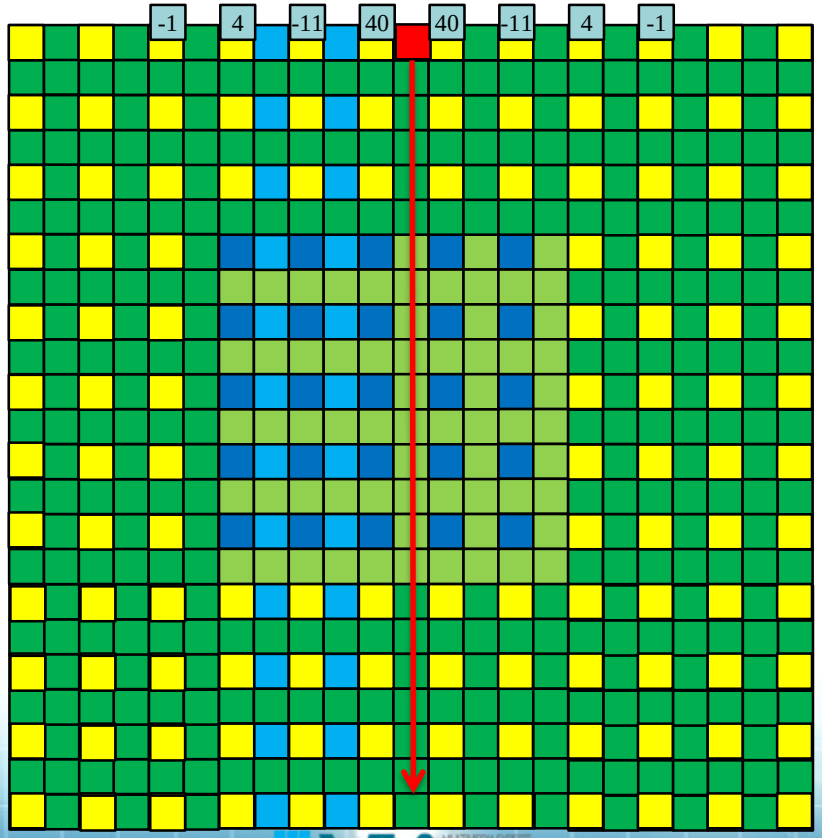


Image Upsampling

- Image upsampling
  - Interpolation method

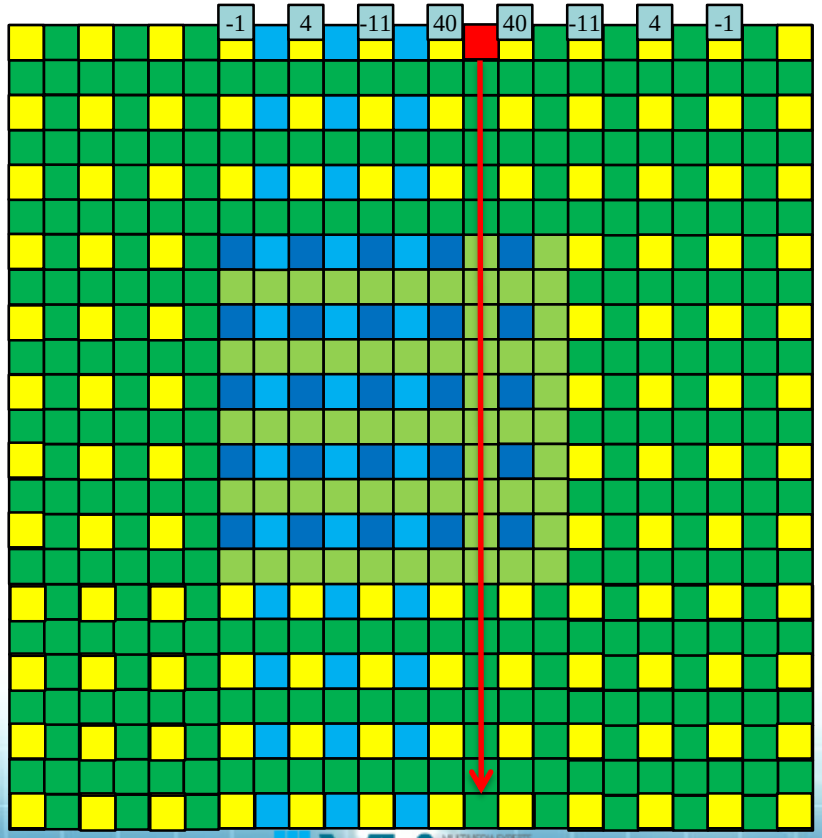


Image Upsampling

- Image upsampling
  - Interpolation method

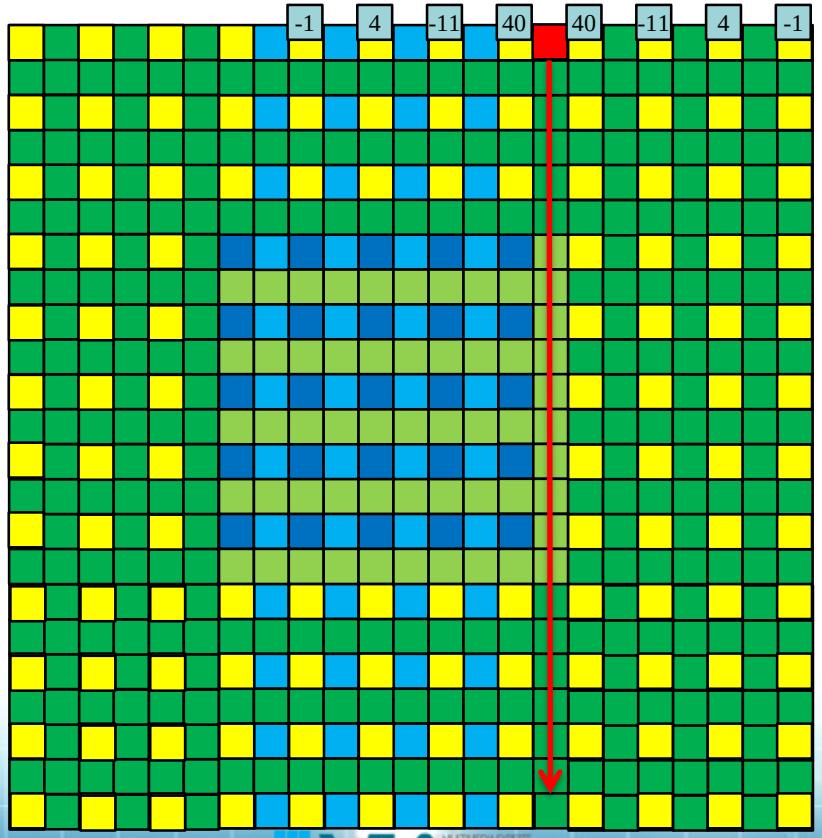


Image Upsampling

- Image upsampling
  - Interpolation method

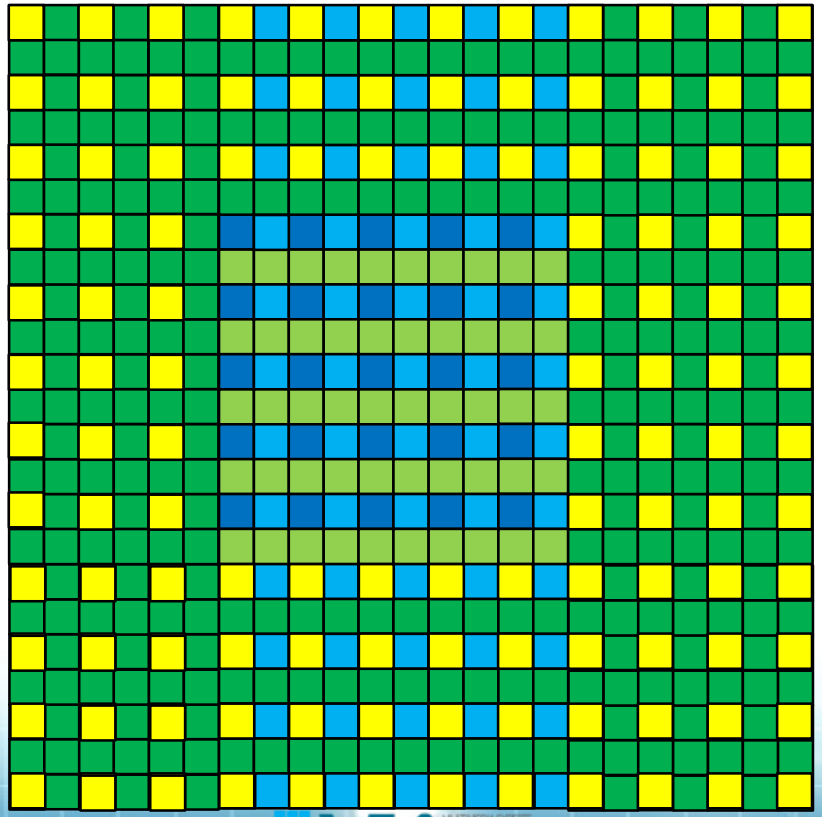


Image Upsampling

- Image upsampling
  - Interpolation method

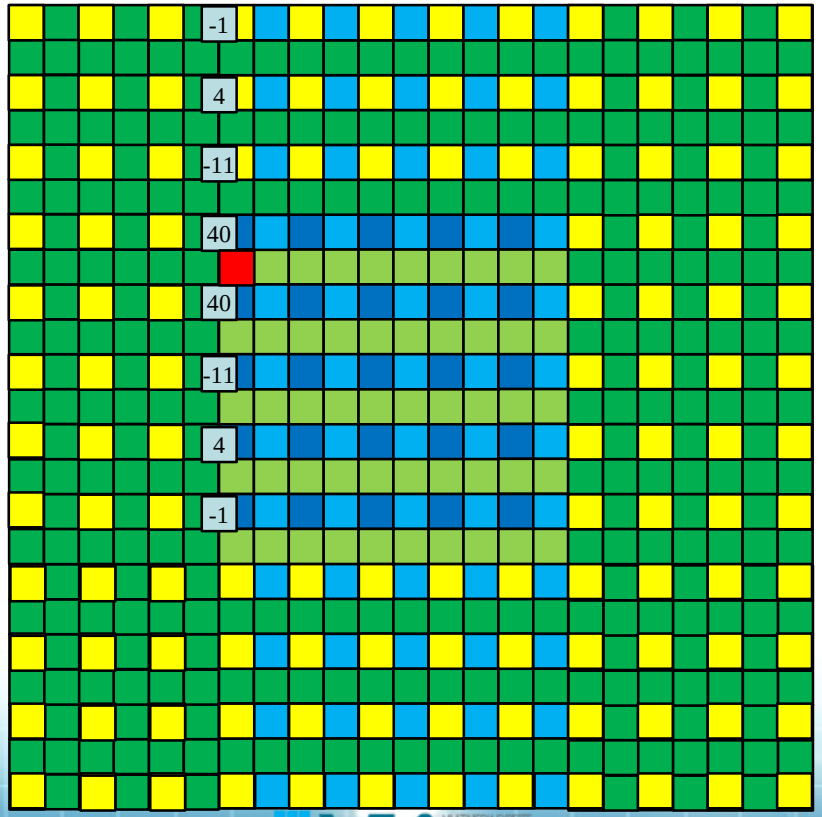


Image Upsampling

- Image upsampling
  - Interpolation method

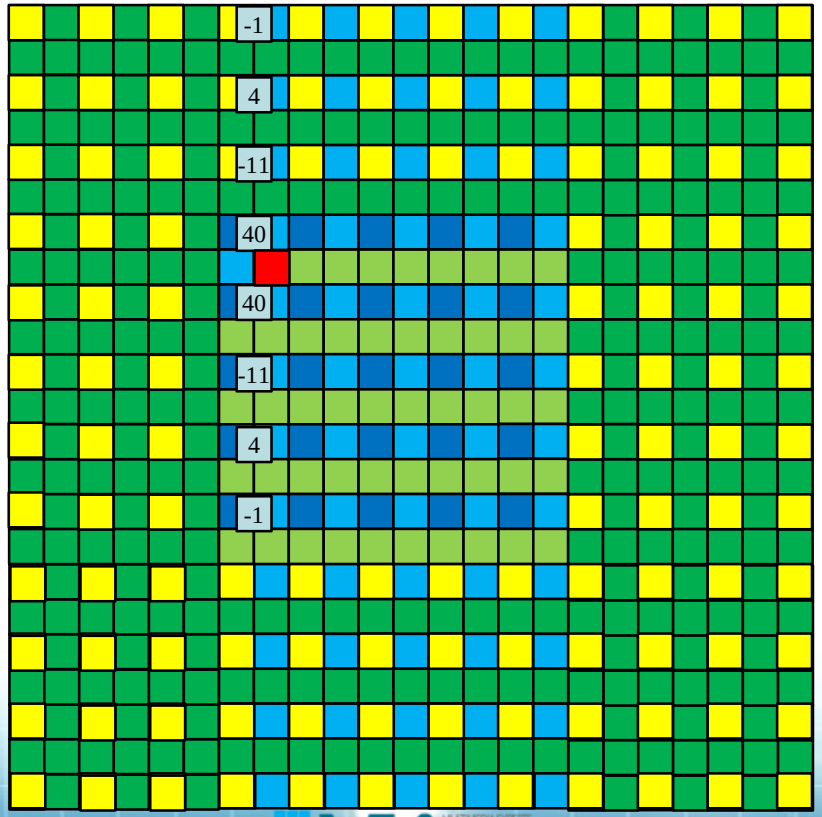


Image Upsampling

- Image upsampling
  - Interpolation method

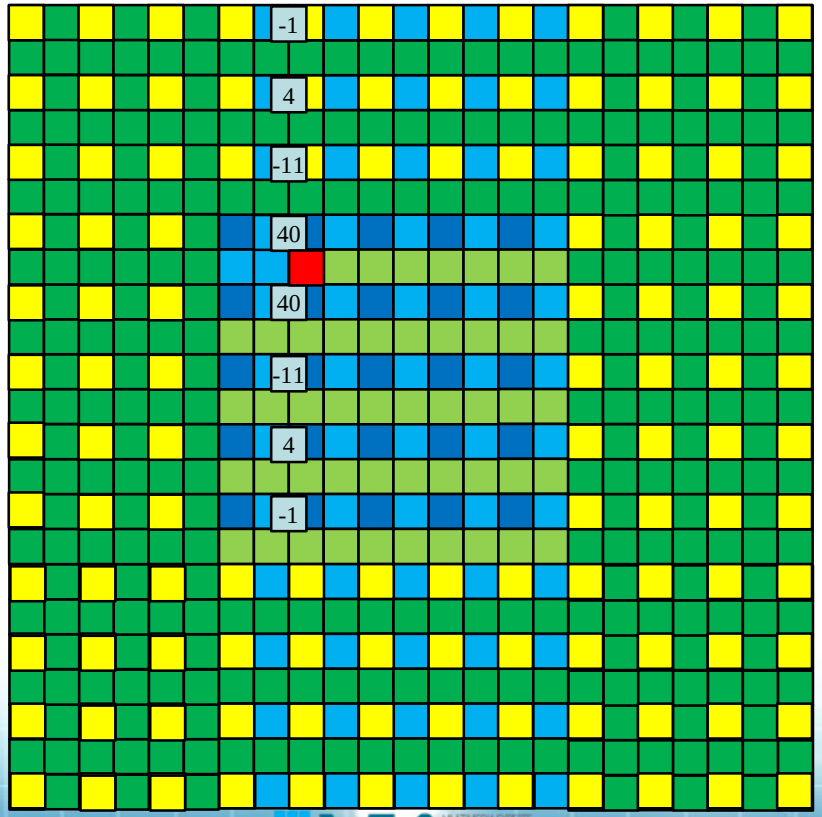


Image Upsampling

- Image upsampling
  - Interpolation method

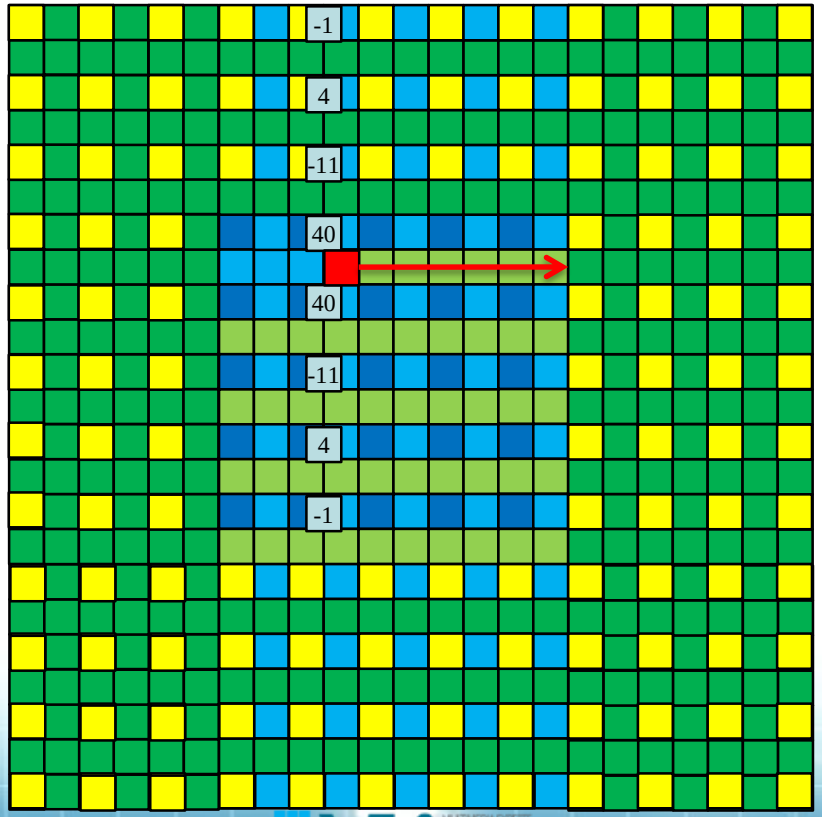


Image Upsampling

- Image upsampling
  - Interpolation method

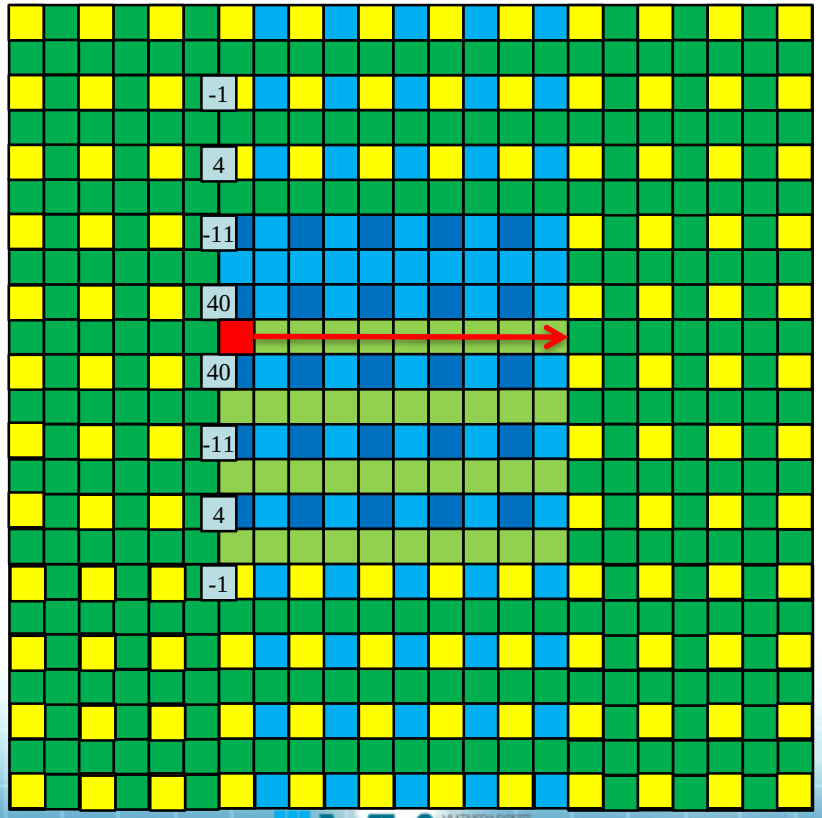


Image Upsampling

- Image upsampling
  - Interpolation method

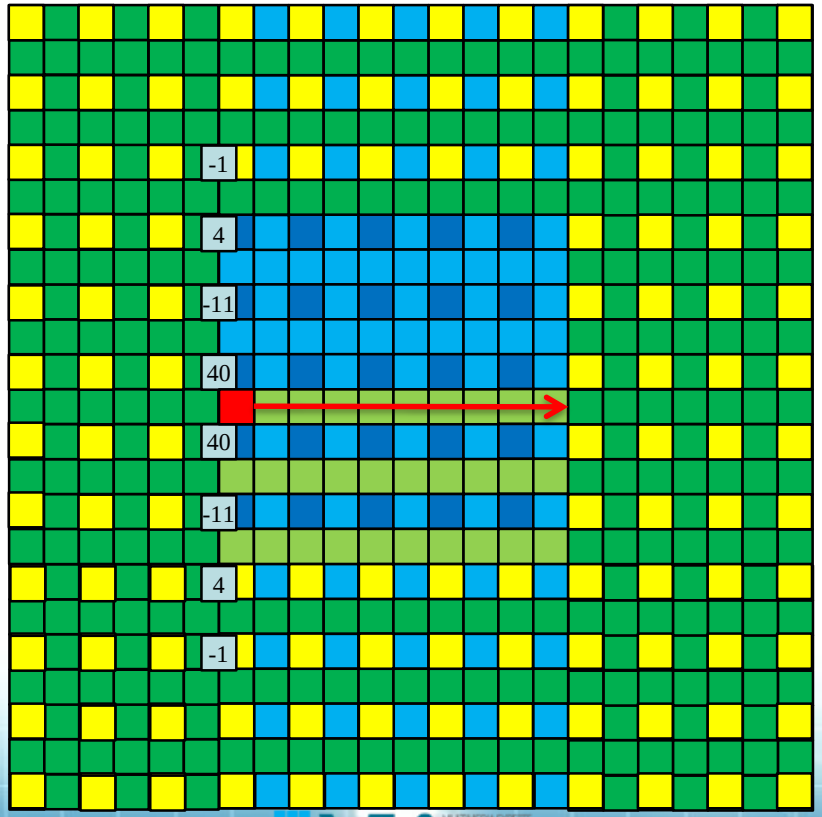


Image Upsampling

- Image upsampling
  - Interpolation method

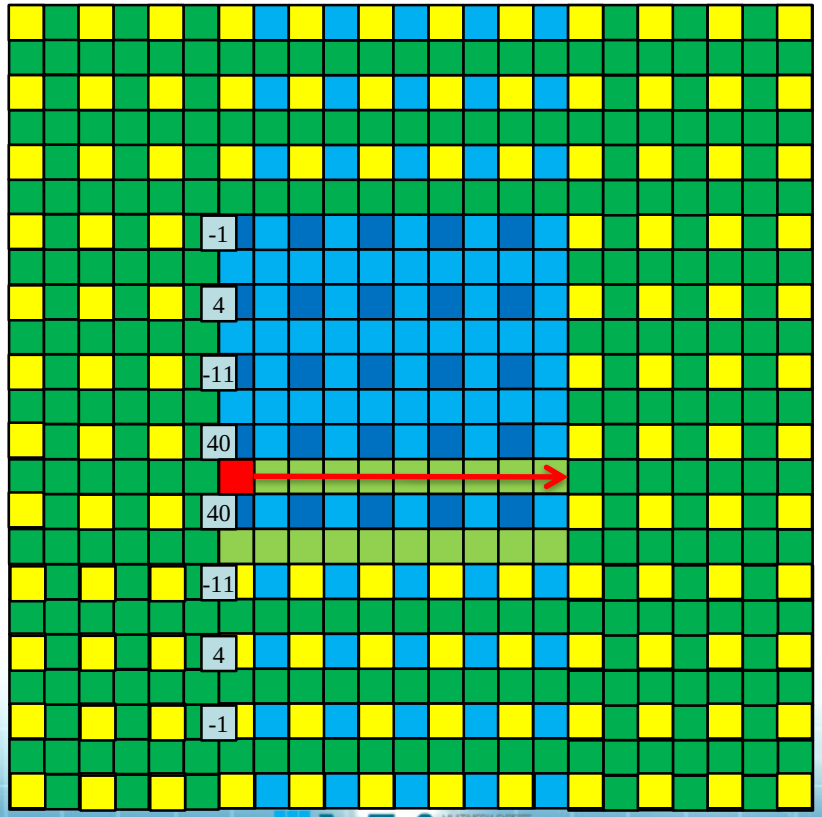


Image Upsampling

- Image upsampling
  - Interpolation method

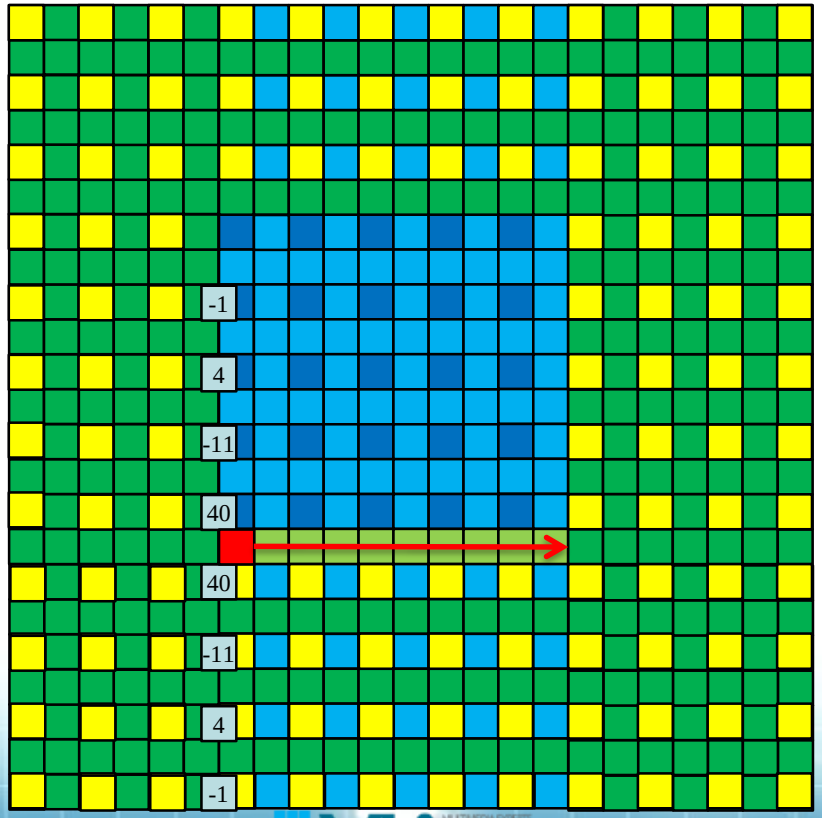


Image Upsampling

- Image upsampling
  - Interpolation method

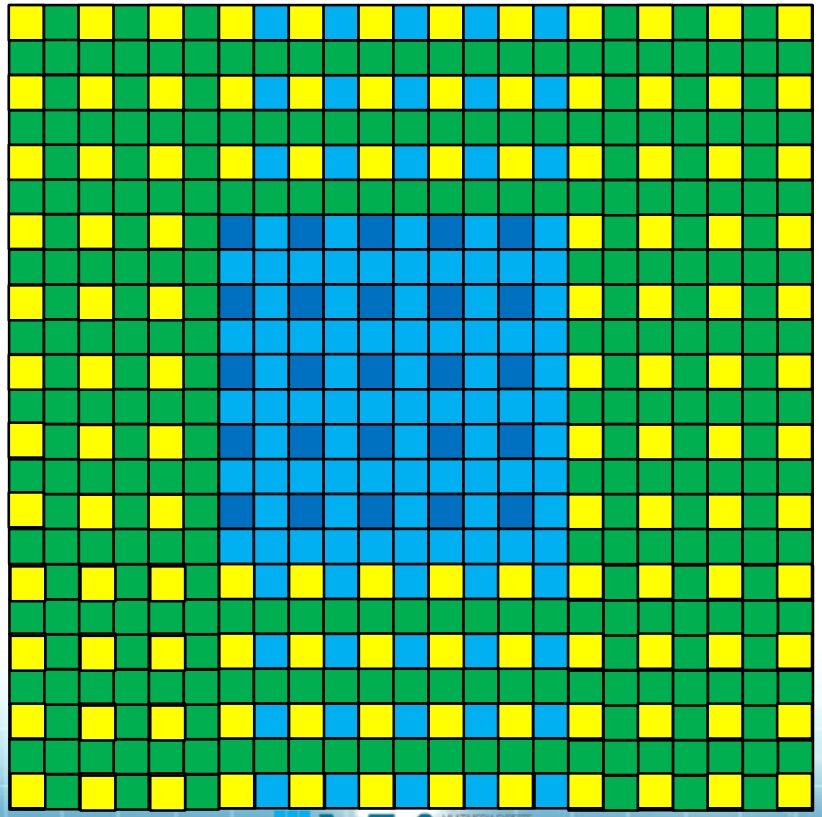
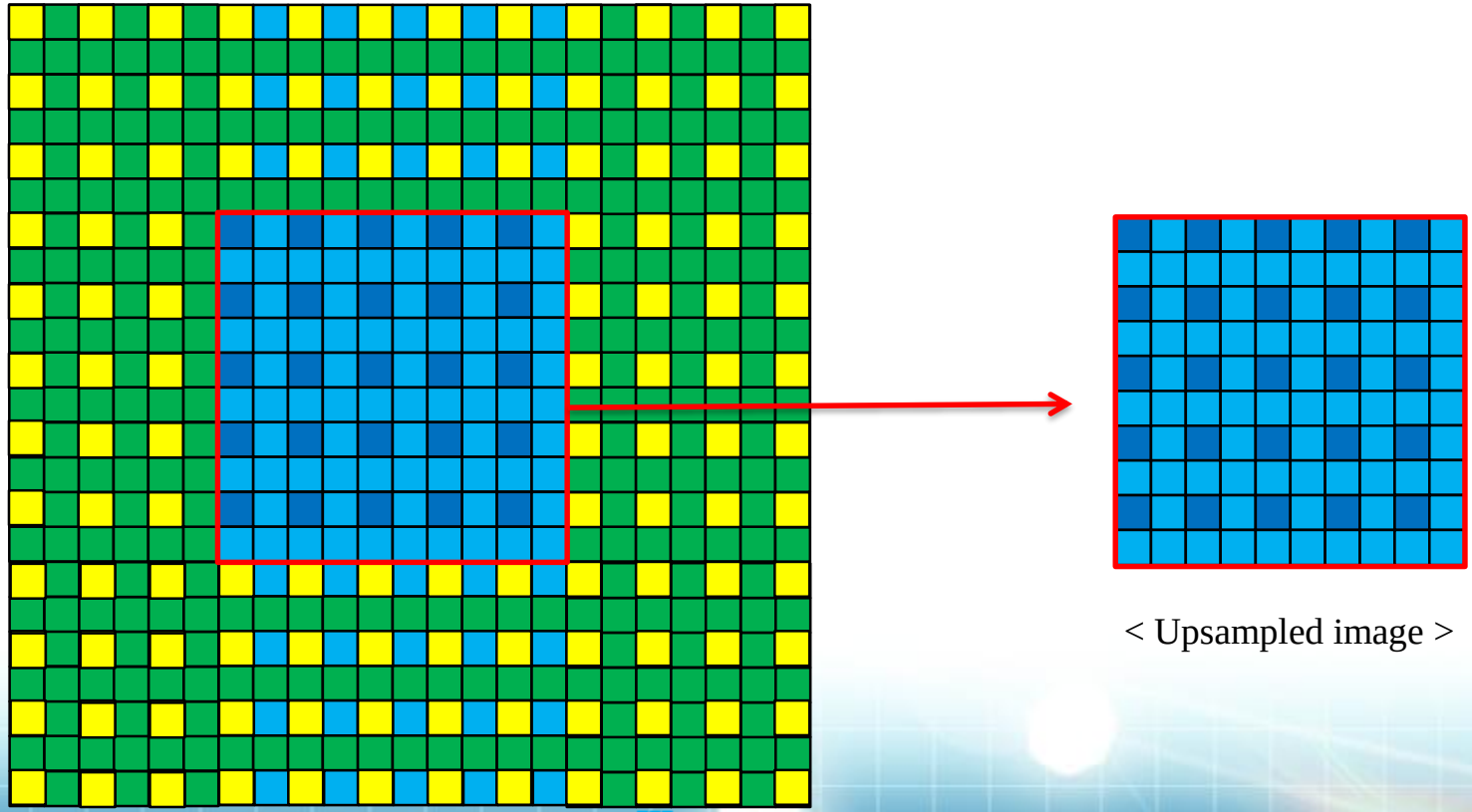


Image Upsampling

- Image upsampling
  - Interpolation method



< Upsampled image >

Peak Signal to Noise Ratio(PSNR)

- PSNR
  - Ratio between the maximum possible power of a signal and the power of corrupting noise

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

where
- MSE : mean squared error
- $I(i,j)$: original image
- $K(i,j)$: distorted image

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

where MAX_I^2 is the maximum possible pixel value of the image

Example

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>           //header file

#define WIDTH 256
#define HEIGHT 256          //image size

#define Clip(x) x<0?0:(x>255?255:x)

typedef unsigned char BYTE;

BYTE** MemAlloc_2D(int width, int height);          //2D memory allocation
void MemFree_2D(BYTE** arr, int height);            //2D memory free

void FileRead(char* filename, BYTE** img_in, int width, int height);    //read data from a file
void FileWrite(char* filename, BYTE** img_out, int width, int height);  //write data to a file

void UpSampling_Copy(BYTE** img_in, BYTE** img_out, int width, int height);     //up-sampling by 2 using copy method
void UpSampling_Average(BYTE** img_in, BYTE** img_out, int width, int height);  //up-sampling by 2 using average method
void UpSampling_Filter(BYTE** img_in, BYTE** img_out, int width, int height);   //up-sampling by 2 using interpolation method

float GetPSNR(BYTE** img_ori, BYTE** img_dist, int width, int height);  //PSNR calculation

int main()
{
    BYTE **img_in, **img_out_copy, **img_out_average, **img_out_filter;

    img_in = MemAlloc_2D(WIDTH, HEIGHT);
    FileRead("Lena(256x256).raw", img_in, WIDTH, HEIGHT);

    img_out_copy = MemAlloc_2D(WIDTH * 2, HEIGHT * 2);
    img_out_average = MemAlloc_2D(WIDTH * 2, HEIGHT * 2);
    img_out_filter = MemAlloc_2D(WIDTH * 2, HEIGHT * 2);

    UpSampling_Copy(img_in, img_out_copy, WIDTH, HEIGHT);
    UpSampling_Average(img_in, img_out_average, WIDTH, HEIGHT);
    UpSampling_Filter(img_in, img_out_filter, WIDTH, HEIGHT);

    FileWrite("[UpSampling_Copy]Lena(512x512).raw", img_out_copy, WIDTH * 2, HEIGHT * 2);
    FileWrite("[UpSampling_Average]Lena(512x512).raw", img_out_average, WIDTH * 2, HEIGHT * 2);
    FileWrite("[UpSampling_Filter]Lena(512x512).raw", img_out_filter, WIDTH * 2, HEIGHT * 2);

    printf("Copy method PSNR : %.2f dB \n\n", GetPSNR(img_out_filter, img_out_copy, WIDTH * 2, HEIGHT * 2));
    printf("Average method PSNR : %.2f dB \n\n", GetPSNR(img_out_filter, img_out_average, WIDTH * 2, HEIGHT * 2));

    MemFree_2D(img_in, HEIGHT);
    MemFree_2D(img_out_copy, HEIGHT * 2);
    MemFree_2D(img_out_average, HEIGHT * 2);
    MemFree_2D(img_out_filter, HEIGHT * 2);
    return 0;
}
```

Example

```
BYTE** MemAlloc_2D(int width, int height)
{
    BYTE** arr;
    arr = (BYTE**)malloc(sizeof(BYTE*)*height);
    for (int i = 0; i < height; i++)
        arr[i] = (BYTE*)malloc(sizeof(BYTE)*width);

    return arr;
}


void MemFree_2D(BYTE** arr, int height)
{
    for (int i = 0; i < height; i++)
        free(arr[i]);
    free(arr);
}

void FileRead(char* filename, BYTE** img_in, int width, int height)
{
    FILE *fp_in;
    int i;

    fp_in = fopen(filename, "rb");
    for (i = 0; i < height; i++)
        fread(img_in[i], sizeof(BYTE), width, fp_in);
    fclose(fp_in);
}
void FileWrite(char* filename, BYTE** img_out, int width, int height)
{
    FILE *fp_out;
    int i;

    fp_out = fopen(filename, "wb");
    for (i = 0; i < height; i++)
        fwrite(img_out[i], sizeof(BYTE), width, fp_out);
    fclose(fp_out);
}
```

Image Upsampling

- Example

```
void UpSampling_Copy(BYTE** img_in, BYTE** img_out, int width, int height)
{
    int i, j, m, n;
    for (i = 0; i < height; i++){
        for (j = 0; j < width; j++){
            for (m = 0; m < 2; m++){
                for (n = 0; n < 2; n++){
                    img_out[2 * i + m][2 * j + n] = img_in[i][j];
                }
            }
        }
    }
}

void UpSampling_Average(BYTE** img_in, BYTE** img_out, int width, int height)
{
    BYTE** img_in_padding = MemAlloc_2D(width + 1, height + 1);
    BYTE** img_out_padding = MemAlloc_2D((width + 1) * 2, (height + 1) * 2);
    int i, j;
    for (i = 0; i < height; i++){
```

?

```
    MemFree_2D(img_in_padding, height + 1);
    MemFree_2D(img_out_padding, (height + 1) * 2);
}
```

Example

```
void UpSampling_Filter(BYTE** img_in, BYTE** img_out, int width, int height)
{
    int interpolFilter[8] = { -1,4,-11,40,40,-11,4,-1 };
    int sum = 64;       //sum of interpolation filter's coefficients
    float temp;

    int i, j, m;

    BYTE** img_in_padding = MemAlloc_2D(width + 7, height + 7);
    BYTE** img_out_padding = MemAlloc_2D((width + 7) * 2, (height + 7) * 2);
```

?

```
    MemFree_2D(img_in_padding, height + 7);
    MemFree_2D(img_out_padding, (height + 7) * 2);
}

float GetPSNR(BYTE** img_ori, BYTE** img_dist, int width, int height)
{
```

?

```
}
```

Assignment

Assignment

- Example code completion
  - Function implementation
    - void UpSampling_Average(unsigned char** img_in, unsigned char** img_out, int width, int height);
      - Upsampling by 2 using average method
        - img_in : input image
        - img_out : output image(upsampled image)
        - width : input image width
        - height : input image height

Assignment

- Example code completion
  - Function implementation
    - void UpSampling_Filter(unsigned char** img_in, unsigned char** img_out, int width, int height)
      - Upsampling by 2 using interpolation filter
        - img_in : input image
        - img_out : output image to be written
        - width : input image width
        - height : input image height
    - float GetPSNR(unsigned char** img_ori, unsigned char** img_dist, int width, int height);
      - Return the PSNR value
        - img_ori : original image
        - img_dist : distorted image
        - width : image width
        - height : image height

Image Upsampling

- Example
  - Result

< Original >

< Upsampled image using copy method >

< Upsampled image using average method >

Image Upsampling

- Example
  - Result

< Original image >

< Upsampled image using interpolation filter>

Image Upsampling

- Example
  - Result

< Copy method >

< Average method >

< Interpolation filter >

Peak Signal to Noise Ratio(PSNR)

- PSNR

< Original : Interpolation filter >

< Copy method PSNR : 30.4 dB >

< Average method PSNR : 37.9 dB >