Θόλωση εικόνας με φίλτρο μέσου όρου

Για κάθε κελί της συνιστάμενης y αντικαθιστούμε την τιμή του,

```
for (int x = 0; x < 240; x++)
{
    for (int y = 0; y < 416; y++)
    {
        newyplane[x][y] = find_avg(x, y);
    }
}
```

με τον μέσο όρο των τιμών των 9 κοντινότερων κελίων με εκείνο, μέσω της συνάρτησης find_avg

```
int find_avg(int x, int y)
{
    int sum = 0;
    for (int i = (x - 1); i < (x + 2); i++)
        for (int j = (y - 1); j < (y + 2); j++)
            if (i >= 0 && i < 240 && j >= 0 && i < 416)
            {
                sum = sum + yplane[i][j];
            }
    return sum / 9;
}
```

Το ίδιο κάνουμε και για τις μάσκες των 9x9 και 15x15 pixel, αντικαθιστώντας τις τιμές του κάθε pixel με τον μέσο όρο των 81 και 225 κοντινότερων του pixel αντίστοιχα.

Το αποτέλεσμα είναι η όλο και πιο έντονη θόλωση της εικόνας.

Από την αρχική της μορφή

Στη μορφή της για μάσκα 3x3,

9x9

Και 15x15

Θόλωση εικόνας με φίλτρο μέσης τιμής

Για κάθε κελί της συνιστάμενης y αντικαθιστούμε την τιμή του,

```
for (int x = 0; x < 240; x++)
{
    for (int y = 0; y < 416; y++)
    {
        newyplane[x][y] = median(x, y);
    }
}
```

με τον μέσο όρο των τιμών των 9 κοντινότερων κελίων με εκείνο, μέσω της συνάρτησης median,

```
int median(int x, int y)
{
    int result = 0, counter = 0;
    int array[9] = {0, 0, 0, 0, 0, 0, 0, 0, 0};
    for (int i = (x - 1); i < (x + 2); i++)
        for (int j = (y - 1); j < (y + 2); j++)
            if (i >= 0 && i < 240 && j >= 0 && i < 416)
            {
                array[counter] = yplane[i][j];
                counter++;
            }
            else
            {
                array[counter] = 0;
                counter++;
            }
    qsort(array,9,sizeof(int),cmpfunc);
    return array[4];    //return value at the middle of the sorted array
}
```

Η οποία δημιουργεί και αρχικοποιεί με μηδενικά ένα πίνακα και για κάθε pixel της y συντεταγμένης που δεν ξεφεύγει των ορίων της αρχικής εικόνας και έχει απόσταση +-1, τοποθετεί την τιμή του μέσα σε αυτόν τον πίνακα. Τότε τον ταξινομεί με την συνάρτηση qsort και επιστρέφει την μέση τιμή του ή αλλιώς το στοιχείο στη θέση 4.

Παρόμοια διαδικασία ακολουθείται και για τις δυο άλλες περιπτώσεις των 9x9 και 15x15 μασκών.

Το αποτέλεσμα είναι η σταδιακή θόλωση των εικόνων για κάθε αύξηση του μεγέθους της μάσκας.

Αρχική εικόνα

Μάσκα 3x3

Μάσκα 9x9

Μάσκα 15x15

Όξυνση εικόνας με Λαπλασιανη μάσκα

Για την λαπλασιανη μασκα χρησιμοποιουμε δυο συναρτησεις, την laplacian και την bound.Η laplacian κανει χρηση της μασκας c

0	−1	0
−1	4	−1
0	−1	0

Από το Figure 3.37 του βιβλίου Digital Image Processing, (3rd ed.) για κάθε pixel .

```
int laplacian(int x, int y)
{
    int up = 0, down = 0, left = 0, right = 0, center = 0;
    center = (+4) * yplane[x][y];
    if (y - 1  >= 0)        left  = (-1) * yplane[x][y - 1];
    if (y + 1 < 416)        right = (-1) * yplane[x][y + 1];
    if (x - 1  >= 0)        up    = (-1) * yplane[x - 1][y];
    if (x + 1 < 240)        down  = (-1) * yplane[x + 1][y];
    return up+down+left+right+center;
}
```

Τότε για κάθε pixel της αρχικής εικόνας υπολογίζουμε μέσω της laplacian την τιμή του με τη μάσκα και το προσθέτουμε στην τιμή του αρχικού pixel, φροντίζοντας μέσω της συνάρτησης bound οι

τιμές που επιστρέφονται από την παράσταση να μην υπερβαίνουν το 255 και να μην είναι μικρότερες από το 0.

```
for (int x = 0; x < 240; x++)
{
    for (int y = 0; y < 416; y++)
    {
        newyplane[x][y] = bound(yplane[x][y] + laplacian(x, y));
    }
}
```

Η εικόνα που δημιουργείται ως αποτέλεσμα έχει σε σχέση με την αρχική εικόνα εντονότερες άκρες.

Αρχική εικόνα

Αποτέλεσμα