# untitled

Naziv predmeta:  
**Osnovi Algoritama i Struktura DSP 2**

**Projektni zadatak 2**

**Interpolacija slike**

|  |  |
| --- | --- |
| Profesor: Željko Lukač | Student:  Nikola Malenčić, RA122/2015 |
|  |  |
| Novi Sad, Mart 2018. | |

# Izveštaj

# Zadatak 1):

void sampleAndHold(uchar input[], int xSize, int ySize, uchar output[], int newXSize, int newYSize);

Kratak opis: Funkcija koja vrši skaliranje slike upotrebom Sample and Hold algoritma. Algoritam je jako jednostavan i ogleda se u tome da se vrednosti novih piksela uzimaju od poznate vrednost najbližeg piksela u osnovnom rasteru.

Resursi: Minimalno iskorišćenje procesora

Parametri: 1. Input – ulazna slika u obliku niza unsigned karaktera

2. xSize – širina osnovne slike

3. ySize – dužina osnovne slike

4. output – niz gde ce biti smestena nova slika

5. newXSize – širina skalirane slike

6. newYSize – dužina skalirane slike

Prednosti: Jednostavnost, malo resursa

Mane: Pikselizovana slika, grube ivice

Primer:

 1.1 Originalna slika



1.2. Slika nakon primene Sample and Hold algoritma za faktor skaliranja V: 1.3, H: 1.5

# Zadatak 2):

void bilinearInterpolate(uchar input[], int xSize, int ySize, uchar output[], int newXSize, int newYSize);

Kratak opis: Funkcija koja vrši skaliranje slike upotrebom bilinearne interpolacije. Algoritam je kompleksniji od SH algoritma i ogleda se u tome da se vrednost piksela racuna na osnovu 4 najbliže tačke, i to po sledećoj formuli: 

Y – novi piksel, X – stari piksel, (a, b) – faktori udaljenosti, (m, n) – koordinate starog piksela

Resursi: Osrednje korišćenje procesora

Parametri: 1. Input – ulazna slika u obliku niza unsigned karaktera

2. xSize – širina osnovne slike

3. ySize – dužina osnovne slike

4. output – niz gde ce biti smestena nova slika

5. newXSize – širina skalirane slike

6. newYSize – dužina skalirane slike

Prednosti: Blaže ivice

Mane: Kompleksnija realizacija

Primer:



2.1. Slika nakon SH interpolacije V: 1.3, H: 1.5



2.2. Slika nakon bilinearne interpolacije V: 1.3, H: 1.5

# Zadatak 3):

void bicubicInterpolate(uchar input[], int xSize, int ySize, uchar output[], int newXSize, int newYSize);

Kratak opis: Funkcija koja vrši skaliranje slike upotrebom bikubične interpolacije. Algoritam je kompleksan i procesorski zahtevan. Izračunavanje vrednosti interpolacione tačke se vrši tako što se nad 4 reda od po 4 tačke poziva kubična interpolacija. Dobijena 4 rezultata smestiti u niz od 4 elementa, i nad njima ponovo pozvati kubičnu interpolaciju. Kubična interpolacija je definisana sledećim formulama:

**

d – distanca novog od starog piksela

Resursi: Veliko opterećenje procesora

Parametri: 1. Input – ulazna slika u obliku niza unsigned karaktera

2. xSize – širina osnovne slike

3. ySize – dužina osnovne slike

4. output – niz gde ce biti smestena nova slika

5. newXSize – širina skalirane slike

6. newYSize – dužina skalirane slike

Prednosti: Skoro u potpunosti zadržan kvalitet slike

Mane: Neophodno mnogo resursa

Primer:



Slika 3.1. Slika nakon bilinearne interpolacije V: 1.3, H: 1.5

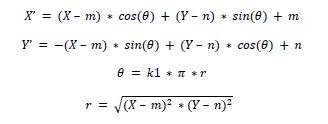


Slika 3.2. Slika nakon bikubične interpolacije V: 1.3, H:1.5

## Zadatak 4):

void imageSwirl(uchar input[], int xSize, int ySize, uchar output[], int m, int n, double angle);

Kratak opis: Funkcija koja uvrće datu sliku za faktor k1 upotrebom SH algoritma. Slika se uvrće oko odredjene tačke, u ovom projektu je realizovano da je ta tačka uvek centar slike. Novi pikseli se izračunavaju po sledećim formulama, gde su X', Y' koordinate tačke stare slike, a X i Y koordinate nove slike:



Resursi: slabo opterećenje procesora

Parametri: 1. Input – ulazna slika u obliku niza unsigned karaktera

2. xSize – širina osnovne slike

3. ySize – dužina osnovne slike

4. output – niz gde ce biti smestena nova slika

5. m – X koordinata tačke oko koje se vrši uvrtanje

6. n – Y koordinata tačke oko koje se vrši uvrtanje

7. angle – faktor uvrtanja

Prednosti: jednostavno, malo koristi resurse

Mane: loše ivice

Primer:



Slika 4.1. Originalna slika

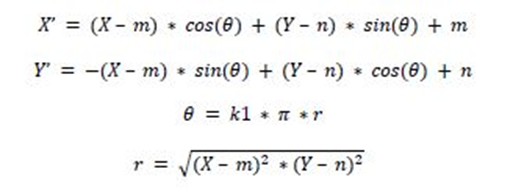


Slika 4.2. Slika nakon uvrtanja za faktor 0.004

## Zadatak 5):

void imageSwirlBilinear(uchar input[], int xSize, int ySize, uchar output[], int m, int n, double angle);

Kratak opis: Funkcija koja uvrće datu sliku za faktor k1 upotrebom bilinearne interpolacije. Slika se uvrće oko odredjene tačke, u ovom projektu je realizovano da je ta tačka uvek centar slike. Novi pikseli se izračunavaju po sledećim formulama, gde su :



(X’, Y’) – koordinate starog piksela, (X, Y) koordinate novog piksela



Y – novi piksel, X – stari piksel, (a, b) - faktori

Resursi: srednje opterećenje procesora

Parametri: 1. Input – ulazna slika u obliku niza unsigned karaktera

2. xSize – širina osnovne slike

3. ySize – dužina osnovne slike

4. output – niz gde ce biti smestena nova slika

5. m – X koordinata tačke oko koje se vrši uvrtanje

6. n – Y koordinata tačke oko koje se vrši uvrtanje

7. angle – faktor uvrtanja

Prednosti: finije obradjene ivice

Mane: kompleksniji algoritam od SH, veće iskorišćenje procesora

Primer:



Slika 5.1. Slika nakon uvrtanja za faktor 0.004 Sample and Hold algoritam



Slika 5.2. Slika nakon uvrtanja za faktor 0.004 uz pomoć bilinearne interpolacije