====== Tema Metode Numerice =====

Autor : Ciocan-Merauta Mihai

Grupa : 314CC

Timp estimat de rezolvare : 8 ore

---

1. Interpolare Nearest Neighbour

- nn_2x2
  - Stiind faptul ca matricea de pixeli are valori in intervalul [1, 2] am verificat folosind 4 if-uri unde se afla cel mai apropiat colt, ajutandu-ma de punctul de mijloc.
- nn_2x2_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia nn_2x2 pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.
- nn_resize
  - Am calculat factorii de scalare, matricea de transformare si inversa acesteia.
  - In parcurgerea imaginii (cele doua for-uri), am aflat pixelul din imaginea initiala folosindu-ma de matricea inversa de transformare. Acest pixel are coordonatele shiftate cu 1 la stanga asa ca am adunat 1 la acestea pentru a corespunde cu coordonatele corespunzatoare.
  - Am calculat pixelul din imaginea interpolata folosindu-ma de cel mai apropiat colt al punctului (xp, yp) si functia round.
  - Am facut cast la uint8 matricii R.
- nn_resize_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia nn_resize pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.

---

2. Interpolare Bilineara

- bilinear_coef
  - Am facut cast matricii de imagine la double
  - Am calculat matricea A din sistem folosind formula corespondenta rescrierii functiei cu sistem
  - Am calculat matricea termenilor liberi, corespondenta pixelilor din colturile imaginii
  - Am calculat coeficientii, rezolvand sistemul folosind linsolve fara vreun argument (rezolva sistemul cu factorizare LU cu pivotare partiala, pentru matrici A patratice).

- bilinear_2x2
  - Am calculat coeficientii de interpolare folosind functia bilinear_coef, apoi am aflat valoarea interpolata a imaginii finale folosindu-ma de scrierea alternativa a functiei cu ajutorul coeficientilor si pixelilor imaginii initiale.
  - Am facut cast imaginii la uint8.
- bilinear_2x2_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia bilinear_2x2 pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.
- bilinear_resize
  - Am calculat factorii de scalare, matricea de transformare si inversa acesteia.
  - In parcurgerea imaginii (cele doua for-uri), am aflat pixelul din imaginea initiala folosindu-ma de matricea inversa de transformare. Acest pixel are coordonatele shiftate cu 1 la stanga asa ca am adunat 1 la acestea pentru a corespunde cu coordonatele corespunzatoare.
  - Folosindu-ma de functia surrounding_points am calculat punctele ce inconjoara pixelul (xp, yp).
  - Am calculat valoarea interpolata a pixelului imaginii finale folosind formula scrierii functiei cu ajutorul coeficientilor.
  - Am facut cast imaginii la uint8.
- bilinear_resize_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia bilinear_resize pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.
- bilinear_rotate
  - Am calculat sinus si cosinus de rotation_angle si am scris matricea de transformare si inversa acesteia.
  - In parcurgerea imaginii (cele doua for-uri), am aflat pixelul din imaginea initiala folosindu-ma de matricea inversa de transformare. Acest pixel are coordonatele shiftate cu 1 la stanga asa ca am adunat 1 la acestea pentru a corespunde cu coordonatele corespunzatoare.
  - Am verificat daca punctele xp si yp sunt in interiorul imaginii, iar daca acestea nu se afla pun un pixel negru in imaginea noua. In cazul in care se afla, calculez punctele ce inconjoara pixelul (xp, yp) folosind surrounding_points, dar acesta poate sa returneze si valori negative care nu sunt dorite in imaginea finala, iar in acest caz pun un pixel negru in imaginea finala. Daca valorile sunt pozitive calculez coeficientii de interpolare si valoarea noua din matricea imaginii finale.
  - Am facut cast imaginii la uint8.
- bilinear_rotate_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia bilinear_rotate pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.
- surrouding_points
  - Calculez cele 4 puncte care inconjoara pixelul (x, y) folosind functia floor care ia partea intreaga a numarului dat. In cazul in care pixelul (x, y) se afla pe frontiera imaginii, valorile care

inconjoara punctul vor fi fix frontiera si frontiera - 1.

---

3. Interpolare Bicubica

- fx
  - Calculez derivata functiei in raport cu x
- fy
  - Calculez derivata functiei in raport cu y
- fxy
  - Calculez derivata functiei in raport cu xy
- bicubic_coef
  - Calculez cele 3 matrice componente ale matricei de coeficienti.
  - Convertesc matricele la double.
  - Calculez matricea de coeficienti ca fiind produsul celor 3 matrice de mai sus.
- surrounding_points
  - Calculez cele 4 puncte care inconjoara pixelul (x, y) folosind functia floor care ia partea intreaga a numarului dat. In cazul in care pixelul (x, y) se afla pe frontiera imaginii, valorile care inconjoara punctul vor fi fix frontiera si frontiera - 1.
- precalc_d
  - Initializez matricele finale cu zero.
  - Fiecare matrice va fi derivata imaginii initiale in raport cu x, y, xy. In calcularea acestora tin cont de faptul ca Ix are 0-uri pe prima si ultima coloana, Iy pe prima si ultima linie, Ixy pe toate marginile.
- bicubic_resize
  - Am calculat factorii de scalare, matricea de transformare si inversa acesteia.
  - In parcurgerea imaginii (cele doua for-uri), am aflat pixelul din imaginea initiala folosindu-ma de matricea inversa de transformare. Acest pixel are coordonatele shiftate cu 1 la stanga asa ca am adunat 1 la acestea pentru a corespunde cu coordonatele corespunzatoare.
  - Folosindu-ma de functia surrounding_points am calculat punctele ce inconjoara pixelul (xp, yp).
  - Calculez coeficientii de interpolare bicubica folosind functia bicubic_coef.
  - Trec (xp, yp) in patratul unitate scazand x1, y1 care reies din functia surrounding_points.
  - Calculez valoarea interpolata a pixelului folosind formula de calcul cu ajutorul matricei de coeficienti.
  - Am facut cast imaginii la uint8.
- bicubic_resize_RGB
  - Am extras cele 3 canale ale imaginii, apoi am aplicat functia bicubic_resize pe fiecare dintre canale pe care le-am combinat ulterior in imaginea finala folosind functia cat.

4. Probleme intalnite in rezolvarea task-urilor

- Principalele probleme intalnite in rezolvarea task-urilor au fost legate de debugarea codului. Am avut multe situatii in care apelam functii incepand cu index-ul 0, ceea ce nu este corect.
- De asemenea am mai avut probleme cu testele, in care sunt inversate anumite valori, spre exemplu in cazul calcularii derivatelor in precalc. Ix foloseste derivata in functie de y si Iy in functie de x pentru ca asa sunt facute testele.
- In plus, alte probleme au mai aparut din cauza checker-ului care dureaza foarte mult sa testeze (ceea ce este normal pentru teste foarte mari), dar care in anumite situatii a blocat command line-ul Octave-ului si nu am mai putut face altceva decat sa repornesc aplicatia.

---

5. Alte ganduri

- Aceasta tema a fost una care m-a ajutat foarte mult sa inteleg interpolarea si domeniile sale de aplicabilitate. Nu a fost o tema foarte mare, ceea ce este un plus, mai ales ca a fost si una foarte interesanta si din care am putut invata multe lucruri. Pentru acest lucru si pentru timpul mare disponibil in care putem rezolva tema, a big thumbs up pentru echipa de teme.