VEŽBA 8

Segmentacija slike upotrebom mašinskog učenja

Potrebno predznanje

- Poznavanje programskog jezika C
- 2D signali
- RGB i YUV prostori boja
- Filtriranje 2D signala u vremenskom domenu
- Računanje histograma slike

ZADATAK

1.1 Zadatak 1

Završiti implementaciju funkcije

- `static vector<vector<int>> kMeans(vector<KMeansPoint> points, int nFeatures, int K);`

Funkcija vrši grupisanje tačaka u N-dimenzionom prostoru koristeći *K-means* algoritam.

Algoritam se sastoji iz sledećih koraka:

- Za zadati broj grupa K inicijalizovati K centralnih tačaka u N-dimenzionom prostoru.
- Svaku tačku pridružiti grupi (klasteru) čija centralna tačka je najbliža zadatoj tački.
- Pronaći novu centralnu tačku za svaki klaster tako što će se izračunati aritmetička sredina svih tačaka u klasteru
- Ponavljati process dokle god barem jedna centralna tačka menja svoju poziciju.

Parametri funkcije su:

- `points` – vektor ulaznih tačaka
- `nFeatures` – broj obeležja kojima je opisana tačka
- `K` – broj grupa (klastera)

Ispravnost funkcije ispitati korišćenjem datog primera:

- `void IntensityPlusPositionBasedKMeans(uchar input[], int xSize, int ySize, int K);`

Koji vrši grupisanje piksela na osnovu intenziteta osvetljaja.

1.2 Zadatak 2

Realizovati funkciju

- `void ColorBasedKMeans(uchar input[], int xSize, int ySize, int K);`

Koja vrši grupisanje piksela u slici na osnovu boje.

Parametri funkcije su:

- `input` – Ulazna slika u RGB formatu
- `xSize` – širina slike
- `ySize` – visina slike
- `K` –Zadati broj grupa (klastera)

Unutar funkcije jepotrebno napraviti vector tačaka (tipa KMeansPoint) koji za svaku tačku sadrži opis u formi [nivo crvene, nivo zelene, nivo plave]. Potom pozvati funkciju kMeans kako bi se izvšila segmentacija. Svaki dobijeni segment obojiti različitom bojom.

1.3 Zadatak 3

Realizovati funkciju

- `void IntensityPlusPositionBasedKMeans(uchar input[], int xSize, int ySize, int K);`

Koja vrši grupisanje piksela u slici na osnovu nivoa osvetljaja i položaja.

Parametri funkcije su:

- `input` – Ulazna slika u RGB formatu
- `xSize` – širina slike
- `ySize` – visina slike
- `K` –Zadati broj grupa (klastera)

Unutar funkcije potrebno je izvršiti konverziju slike iz RGB u YUV prostor boja. Nakon toga filtrirati sliku gausovim filterom kako bi se uklonio šum. Potom je potrebno napraviti vektor tačaka (tipa KMeansPoint) koji za svaku tačku sadrži opis u formi [osvetljaj, horizontalni položaj, verzikalni položaj]. Sva tri obeležja je potrebno skalirati tako da se vrednosti nalaze u istom opsegu. Potom pozvati funkciju kMeans kako bi se izvšila segmentacija. Svaki dobijeni segment obojiti različitom bojom.

(Dodatno) Modifikovati rešenje tako što ćete skalirati obeležja koja predstavljaju položaj piksela kako bi umanjili njihov značaj u odnosu na nivo osvetljaja. Isprobajte za različite faktore.

1.4 Zadatak 4

Realizovati funkciju

- `void histogramBasedKMeans(uchar input[], int xSize, int ySize, int blockSize, int K)`

Koja vrši računanje grupisanje piksela na osnovu vrednosti lokalnog histograma osvetljaja okoline zadatog piksela.

Parametri funkcije su:

- `input` – Ulazna slika u RGB formatu
- `xSize` – širina slike
- `ySize` – visina slike
- blockSize – dimenzija bloka za koji se računa histogram
- `K` –Zadati broj grupa (klastera)

Unutar funkcije potrebno je izvršiti konverziju slike iz RGB u YUV prostor boja. Nakon toga filtrirati sliku gausovim filterom kako bi se uklonio šum. Potom je potrebno za svaki piksel (odnosi se na Y komponentu slike) izdvojiti blok veličine blockSize*blockSize oko njega, kvantizovati vrednosti u 8 kvantova, i potom izračunati histogram.

Za računanje histograma možete koristiti pomoćnu funkciju:

- `vector<double> calculateHistogram(vector<uchar> values, int quants)`

koja kao parametre prima vektor koji sadrži sve vrednosti piksela iz bloka i broj kvantova. Povratna vrednost je vektor dužine *quants* koji sadrži vrednosti histograma. Ovaj vektor ujedno predstavlja i vektor obeležja koji se koristi za grupisanje piksela. Napraviti novi objekat tipa *KMeansPoint*, postaviti odgovarajuće koordinate tačke i izračunati vektor obeležja.

Kada su izračunata obeležija za sve piksele, pozvati funkciju kMeans kako bi se izvšila segmentacija. Svaki dobijeni segment obojiti različitom bojom.

1.5 Dodatni zadatak (za napredne)

Realizovati funkciju

- `void SIFTBasedKMeans(uchar input[], int xSize, int ySize, int blockSize, int K);`

Slično kao u prethodnom zadatku, svakom pikselu u slici potrebno je pridružiti vektor obeležja koji zavisi od njegove okoline. Vektor obeležja je potrebno izračunati po uzoru na SIFT algoritam, tako što će svaki piksel biti opisan histogramom orijentacije ivica u njegovoj okolini.

Potrebno je izvršiti konverziju slike u YUV domen. Potom izračunati horizontalni i vertikalni gradijent koristeći Sobel operatore. Nakon toga, za svaku tačku potrebno je izračunati orijentaciju ivica u datoj tački (koristeći atan2 funkciju po uzoru na SIFT). Svaku tačku u slici opisati histogramom uglova koji se nalaze u okolini blockSize*blockSize oko te tačke. Izračunati histogram potrebno je zarotirati za orijentaciju zadate tačke (kao kod SIFT).

Kada su izračunata obeležija za sve piksele, pozvati funkciju kMeans kako bi se izvšila segmentacija. Svaki dobijeni segment obojiti različitom bojom.

Tokom izrade dozvoljeno je ponovno korišćenje delova koda koji se deo implementacije SIFT algoritma.