Stilizacija slike koristeći evolucijsko računanje

Matej Krehula

Evolucijsko računanje

Jedan od načina rješavanja optimizacijskih problema je koristeći evolucijsko računanje. Postoji mnogo varijanti evolucijskih algoritama, a najpoznatiji je genetski algoritam. On kreće od populacije koja ima određeni broj jedinki na koji se zatim primjenjuju određeni operatori, a cilj je postići što veću vrijednost funkcije dobrote. Maksimizacija funkcije dobrote provodi se iterativno. Operatori koji se ističu su:

- elitizam - odabir određenog broja jedinki koje postižu najveću vrijednost funkcije dobrote
- selekcija - odabir jedinki koje ne postižu najveću vrijednost funkcije dobrote (npr. uz nasumičnost)
- križanje - kombiniranje postojećih jedinki
- mutacija - dodavanje nasumičnih promjena postojećem rješenju

Opis razvijenih funkcionalnosti

Program na ulazu prima niz argumenata o kojima ovisi konačan rezultat. Neki od ovih argumenata su:

- width - određuje širinu rezultantne slike
- height - određuje visinu rezultantne slike
- original_path - putanja do slike koju je potrebno stilizirati
- recreated_path - putanja gdje se sprema konačna slika
- angle_f - string koji opisuje funkciju za nagib elipsi koje se iscrtavaju
- size_f - string koji opisuje funkciju za širinu i visinu elipsi koje se iscrtavaju
- fitness_f - string koji opisuje funkcije dobrote
- color_strat - određuje strategiju generiranja boje ("kmeans" ili "random")
- num_colors - broj boja za elipse koje se generiraju
- num_iter - broj iteracija algoritma

- num_mutations - broj mutacija u svakoj iteraciji algoritma
- verbose - određuje obaviještava li se korisnik o vrijednosti funkcije dobrote
- seed - određuje vrijednost seeda

Opis programskog rješenja je sljedeći. Program započinje čitanjem argumenata iz komandne linije. Zatim se čita slika koju želimo stilizirati i generira se prazna slika.

Algoritam je najlakše opisati pseudokodom:

```
za broj iteracija:
    - generiraj boje

    za broj mutacija:
        - napravi operator mutacije
        - izracunaj dobrotu

        ako je dobrota trenutnog rjesenja veca od najvece dobrote:
            - najbolja dobrota postaje trenutna dobrota
            - zapamti najbolju mutaciju

    ako najbolja mutacija povecava najbolju dobrotu:
        - zapamti trenutnu dobrotu kao najbolju dobrotu
        - nacrtaj novo rjesenje
```

Razlika u odnosu na genetski algoritam je da ovaj algoritam ne koristi operatore elitizma, selekcije i križanja, međutim uspijeva odraditi početni zadatak.

Programsko rješenje je ostvareno koristeći jezik python i biblioteke numpy, opencv. Numpy je korišten za računanje potrebno za operator mutacije, opencv je korišten za iterativni prikaz rješenja. U nastavku je nekolicina primjera rada programa.