Univerza v Ljubljani, Fakulteta za elektrotehniko
Podiplomski magistrski študijski program 2. stopnje Elektrotehnika
Informacijsko komunikacijske tehnologije - Obdelava slik in videa (64238)

NAVODILA

Vaja 11: Vektor premika

Pripravili: Luka Škrlj, Gašper Podobnik & Tomaž Vrtovec

Navodila

Naslednje naloge se opravi med laboratorijsko vajo.

Algoritem *bločnega ujemanja* predstavlja enega najosnovnejših pristopov za določanje vektorjev premika na področju obdelave videa.

1. Dan je testni video simple-video.avi, ki je sestavljen iz $K = 153$ zaporednih sivinskih slik velikosti $X \times Y = 160 \times 100$ slikovnih elementov in zapisan v nezgoščeni obliki s frekvenco 25 Hz znotraj zalogovnika AVI.

   Napišite funkcijo, ki naloži točno določeno sliko iz videa:

   ```
   def loadFrame(iVideo, iK):
       # ...
       return oFrame
   ```

   kjer vhodni argument `iVideo` predstavlja video posnetek, ki ga predhodno naložite s funkcijo `VideoCapture` iz knjižnice `cv2` (https://pypi.org/project/opencv-python/), `iK` pa zaporedno številko slike, ki jo želimo naložiti.
   Naložite dani video, iz njega izluščite sliki pri $k = 30$ in $k = 31$ ter sliki prikažite. Pomagajte si z metodama `.set()` in `.read()` objekta `cv2.VideoCapture`.

2. Napišite funkcijo za določanje polja vektorjev premika z algoritmom bločnega ujemanja:

   ```
   def blockMatching(iF1, iF2, iSize, iSearchSize):
       # ...
     return oMF, oCP
   ```

   kjer vhodni argument `iF1` predstavlja sliko videa pri času t_1, `iF2` sliko videa pri času $t_2 > t_1$, `iSize` $= [B_x, B_y]$ vektor dimenzije $B_x \times B_y$ (v slikovnih elementih) posameznega bloka slike, `iSearchSize` $= 2P + 1$ pa velikost (v slikovnih elementih) kvadratnega področja iskanja vektorjev premika. Izhodni argument `oMF` predstavlja polje vektorjev premika, `oCP` pa koordinate središča posameznega bloka slike (v slikovnih elementih). Oba izhodna argumenta sta tridimenzionalni matriki, kjer število vrstic predstavlja število blokov vzdolž vrstic slike, število stolpcev število blokov vzdolž stolpcev slike, število plasti pa x in y komponenti vektorjev premika oz. koordinati središč blokov.

   Bločno ujemanje temelji na razdelitvi slike na M blokov $\mathcal{B}_m$; $m = 1, 2, \ldots, M$, za katere je potrebno določiti pripadajoče vektorje premika $\boldsymbol{d}_m = [d_{m,x}, d_{m,y}]^T$ med sliko $I_k(\boldsymbol{x})$ ob času k in sliko $I_{k-1}(\boldsymbol{x})$ ob času $k-1$ tako, da je napaka napovedovanja $\varepsilon(\boldsymbol{d}_m)$ čim manjša:

$$\boldsymbol{d}_m^* = \arg\min_{\boldsymbol{d}_m \in \mathcal{P}} \varepsilon(\boldsymbol{d}_m); \qquad \varepsilon(\boldsymbol{d}_m) = \frac{1}{|\mathcal{B}_m|} \sum_{\boldsymbol{x} \in \mathcal{B}_m} \left| I_k(\boldsymbol{x}) - I_{k-1}\left(\boldsymbol{x} - \boldsymbol{d}_m\right) \right|,$$

   kjer $\boldsymbol{x} = [x, y]^T$ predstavlja vektor prostorskih koordinat slikovnega elementa slike, $\tilde{I}_k(\boldsymbol{x}) = I_{k-1}\left(\boldsymbol{x} - \boldsymbol{d}_m\right)$ napoved slike $I_k(\boldsymbol{x})$ s kompenzacijo premika, $\boldsymbol{d}_m^*$ pa pripadajoči optimalni vektor premika za blok $\mathcal{B}_m$, ki sestoji iz $|\mathcal{B}_m|$ slikovnih elementov. Kandidate za optimalni vektor premika $\boldsymbol{d}_m^*$ se izbira na področju iskanja $\mathcal{P}$ ob predpostavkah, da je velikost $\mathcal{P}$ enaka $(2P+1) \times (2P+1)$ ter da velja $P = 2^a - 1$; $a \in \mathbb{N}$, in sicer se $\boldsymbol{d}_m^*$ določi s t. i. logaritemskim iskanjem v treh korakih:

$$P_i = \frac{P+1}{2^i}; \qquad \mathcal{P}_i = \left\{ \boldsymbol{x} : (0,0); (0, \pm P_i); (\pm P_i, 0); (\pm P_i, \pm P_i) \right\}; \qquad i = 1, 2, 3.$$

Določite polje vektorjev premika za izluščeni sliki videa pri velikosti bloka 8×8 slikovnih elementov in velikosti področja iskanja vektorjev premika 15 slikovnih elementov.

3. Napišite funkcijo za prikaz polja vektorjev premika:

```
def displayMotionField(iMF, iCP, iTitle, iImage=None):
    # ...
  return fig
```

kjer vhodni argument `iMF` predstavlja polje vektorjev premika, `iCP` koordinate središča posameznega bloka (v slikovnih elementih), `iTitle` naslov prikaznega okna, `iImage` pa je parameter, ki predstavlja sliko iz videa. Če uporabnik ne poda vrednosti za ta argument, se izriše samo polje vektorjev, v nasprotnem primeru pa se polje vektorjev izriše superponirano na podano sliko iz videa. Za prikaz vektorjev si pomagajte s funkcijo `quiver()` iz knjižnice `matplotlib`. Izhodni argument `fig` naj predstavlja objekt prikaznega okna (`fig = plt.figure()`).

Prikažite polje vektorjev premika, ki ste jih pridobili pot točko 2, ter jih superponirajte na pripadajočo sliko videa.

Gradivo

Naslednje naloge so neobvezne in namenjene boljšemu razumevanju vsebine.

1. Izračunajte napoved slike pri času $t_2 > t_1$ na podlagi slike pri času t_1 ter pripadajočo sliko razlik.

2. Sestavite video polja vektorjev premika za vsak par zaporednih slik danega videa in ga shranite v datoteko tipa `gif`, tako da bo frekvenca novega videa enaka frekvenci originalnega video posnetka. Za pridobitev slik videa si pomagajte s Pythonovo funkcijo `fig2img()`:

```
from PIL import Image
import io
def fig2img(fig):
    """Convert a Matplotlib figure to a PIL Image and return it"""
    buf = io.BytesIO()
    fig.savefig(buf)
    buf.seek(0)
    img = Image.open(buf)
    return img
```

   ki celotno sliko prikaznega okna `fig` pretvori v spremenljivko tipa `PIL.Image`. Zapis videa v `gif` naredite tako, da si posazmezne slike tipa `PIL.Image` shranite v seznam (npr. z imenom `frames`), nato pa z metodo `.save()` shranite `gif` na želeno lokacijo. Pomagate si lahko s spodnjim ukazom in dokumentacijo na spletni strani: https://pillow.readthedocs.io/en/stable/handbook/image-file-formats.html#saving.

```
frames[0].save(
        "<ime-posnetka>.gif",
        duration= # potrebno nastaviti,
        loop=0,
        save_all=True,
        optimize=False,
        append_images=frames[1:],
    )
```

3. Preizkusite delovanje algoritma bločnega ujemanja na realnem videu real-video.avi, ki je sestavljen iz $K = 138$ zaporednih sivinskih slik velikosti $X \times Y = 256 \times 144$ slikovnih elementov in zapisan v nezgoščeni obliki s frekvenco 25 Hz znotraj zalogovnika AVI. Izrišite slike (polje vektorjev premika ter polje vektorjev premika, superponirano na sliko) za poljubno izbrani primer slik iz tega videa.