Koda

{.octave include=/home/arun/videoprocess/skripte/korelacija.m}

Imamo tri primere videjev. Ta glavni je sample.mp4, zraven so še napisani pravilno izračunani zamiki. Če se vzame sample.mp4 in zamakne za omenjen zamik, dobimo istoležne slike. , drugi trije so:

- cutsample.mp4 zamik 100
- drugisample.mp4 zamik 275
- tretjisample.mp4 zamik 150

Testiranje.m se uporablja za testiranje, najdi.m je pa glavno vozlišče kode, notri damo dva video posnetka in nam vrne zamik. V planu je narediti malo lepšo prezentacijo zamika oziroma rezultata.

Posnetek.m

To je naš struct, je bila ustvarjena posebi funkcija/struct, katera bo hranila vse podatke o našem video posnetku. Struct se podaja skozi celotni proces in se dopolnjuje.

```
function pos = posnetek(video)
    pos.datoteka=video;
    pos.vektor_sprememb = 0;
    pos.podvzorcene_slike = 0;
    pos.dolzina = 0;
    pos.polozaj_v_normxcorr = "";
    pos.vecji = true;
    pos.pot_do_posnetka= "";
endfunction


```

Generacija.m

Prvi postanek, v začetku nastavimo par parametrov in spremenljivk. V naslednem while loopu beremo video in ga pretvarjamo sproti. Nato zračunamo vektor sprememb in nakoncu dodamo parametre v naš struct. Sprejme path do videa in vrne naš struct.

```
function [video_struct]= generacija (video_posnetek)
  pkg load video
  pkg load image

  #kreiramo objekt
```

```
  video_struct = posnetek(video_posnetek);

  pot_do_posnetka = strcat(pwd,"/",video_posnetek);
  captureObjekt = cv.VideoCapture(video_posnetek);
  podvzorcene_slike = zeros(54 ,96,captureObjekt.FrameCount); #3D matrika init

  #beremo video
  i=1;
  while(captureObjekt.isOpened())
    frame = captureObjekt.read('FlipChannels',true); #flipchannel da gre v RGB namesto BRG
    podvzorceno = imresize(frame,[54 96]);
    sivinska_slika= rgb2gray(podvzorceno);
    podvzorcene_slike(:,:,i) = sivinska_slika;
    i++;
    if(captureObjekt.FrameCount < i )
      break
    endif
  endwhile
  captureObjekt.release()

  #tukaj se zacne MAD
  matrike_video_posnetka = podvzorcene_slike;
  dolzina_3D_matrike = size(matrike_video_posnetka,3) -1; #-1 ker bo vedno en manj rezultat
  printf("stevilo elementov v 3d matriki: %d \n", dolzina_3D_matrike);
  vektor_sprememb(dolzina_3D_matrike)=0; #inicializiramo vektor z niclami
  for i = 1:dolzina_3D_matrike
    #filamo vektor, razlika med i-tem in i+1--tem.
    vektor_sprememb(i)= mad(matrike_video_posnetka(:,:,i), matrike_video_posnetka(:,:,i+1) )
  end
  podvzorcene_slike=uint8(podvzorcene_slike); # spreminjamo v pravi zapis, drugače imshow ne

  video_struct.pot_do_posnetka = pot_do_posnetka;
  video_struct.podvzorcene_slike = podvzorcene_slike;
  video_struct.vektor_sprememb = vektor_sprememb;
endfunction

#-----------samo za preglednost je tukaj tole---------
%function pos = posnetek(video)
%    pos.datoteka=video;
%    pos.vektor_sprememb = 0;
%    pos.podvzorcene_slike = 0;
%    pos.dolzina = 0;
%    pos.polozaj_v_normxcorr = "";
%    pos.vecji = true;
%    pos.pot_do_posnetka= "";
%endfunction
```

Korelacija.m

Sprejme posnetka, vrne vektor korelacije, seznam tau-ov, in dopolnjene posnetke. Prvo kliče našo funkcijo za normirano korelacijo, ki smo jo ustvarili sami, zato ker funkcija zahteva argumente v določenem vrstnem redu. Vektor korelacije zgladimo z gaussovim filtrom z skalrjem vrednosti 30. Nato izločimo vse vrednosti ki so nad 0.6 in tako dobimo možne vrhove. Vrednost 0.6 je variabilna, nebi smela biti fiksna, mogoče jo lahko damo kot parameter funckije.

```
#sprejme dva vektorja
#vrne vrhove
function [a seznam_tau prvi drugi] = korelacija (prvi_posnetek,drugi_posnetek)
pkg load signal
pkg load image



[vektor_korelacije prvi drugi]= normirana_korelacija(prvi_posnetek,drugi_posnetek);
vrhovi= zeros(numel(vektor_korelacije));
vektor_korelacije(end+5)=0;  #kaj pa zacetek , ce je korelacija na zacetku
a = vektor_korelacije; # vracamo za radovednost
v1 = imsmooth(vektor_korelacije, "Gaussian", 30);
v2 = (vektor_korelacije-v1)>0.6;
seznam_tau = find(v2);

#-------------mojo staro primerjanje
%for i = 1:numel(vektor_korelacije)-5
%   #seznam_tau = 10; # ne pride v prvi if.
%  #ce je lokalna manjsa od globalne, spremenu obratno, zakaj ta 0.4 deluje?
%  if ( std(vektor_korelacije(i:i+5))*0.5 > std(vektor_korelacije) )
%    #računamo sproti taue
%     if(vektor_korelacije(i)>0)
%        vrhovi(i) = vektor_korelacije(i);
%        seznam_tau(end +1)  =  i;
%     endif
%  endif
%endfor

endfunction
```

Normirana_korelacija.m

Obracamo vektorje tako, da imamo manjsi/vecji, ker tako sprejme arguemnte normxcorr2.

```
function [korelacija_rezultat manjsi vecji] = normirana_korelacija(A,B)

#obracamo
if length(A.vektor_sprememb) >= length(B.vektor_sprememb)
vecji = A;
manjsi = B;
else
manjsi = A;
vecji = B;
endif

#oznacimo in damo v korelacijo
manjsi.vecji=false;
vecji.vecji=true;
korelacija_rezultat = normxcorr2 (manjsi.vektor_sprememb,vecji.vektor_sprememb);
end
```

Najdi_ujemanja.m

Spet rabimo najsi/vecji. To zaradi računanja zamikov. Tau dobimo tako da odštejemo tau iz seznama vrhov od korelacije , dolzino manjsega vektorja sprememb. nato za en določen zamik naredimo MAD in nato pogledamo če je maksimalna vrednost večja od nekega tresholda ki ga sami določimo, če je, to velja za zamik ki je lahko uporabljen.

```
function[vsivektorji,zamik] = najdi_ujemanja (vrhovi, prvivektor,drugivektor)
  pkg load signal
  pkg load image

  zamik = [];
  primerjalni_vektor =[];
  # nastavimo ker je vecji in ker manjsi
  if(prvivektor.vecji == false )
    manjsivektor = prvivektor;
    vecjivektor = drugivektor;
  else
    manjsivektor = drugivektor;
    vecjivektor = prvivektor;
  endif

  # glavni loop
```

```
  for i =1:length(vrhovi)
    tau = vrhovi(i) - numel(manjsivektor.vektor_sprememb);
    if(tau + length(manjsivektor.vektor_sprememb ) > length(vecjivektor.vektor_sprememb)  )
      continue # če zamik ne ustreza, preskočimo
    endif
    # za določen tau, oziroma zamik, gremo z MAD primerjat, če najdemo zamike.
    for j = 1:length(manjsivektor.vektor_sprememb)
      primerjalni_vektor(j)= mad( vecjivektor.vektor_sprememb(tau+j) , manjsivektor.vektor_s
    endfor

    vsivektorji{i}=primerjalni_vektor; # hranimo, samo ker smo radovedni

    tempsmooth= (imsmooth(primerjalni_vektor, "Gaussian", 30));
    if(max(tempsmooth) < 0.5) # naš "threshold", razlika med piksli manj kot 1, potem je nek
      zamik(end+1) = tau;
    endif
  endfor
endfunction

```