Basi di dati multimediali 2015-16.

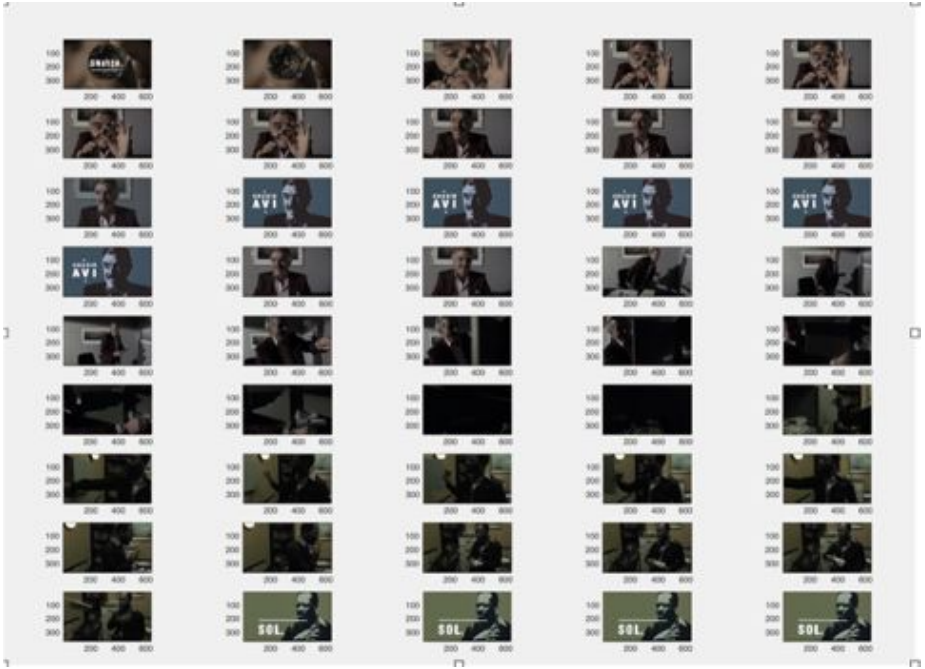
Esercitazione di approfondimento

Dato un video in input, ne estraggo i frames

```
%leggiamo da videoPath la location del video
function [frames] = extractFrames(videoPath)
   v = VideoReader(videoPath);
   vHeight = v.Height;
   vWidth = v.Width;
   frames = struct('cdata',zeros(vHeight,vWidth,3,'uint8'),'colormap',[]);

   k = 1;
   %estraiamo ogni frame e lo mettiamo all'interno della struct frames
   while hasFrame(v)
      frames(k).cdata = readFrame(v);
      k = k+1;
   end
end
```


GORGEOUS
GEORGE



Istogramma dei colori

Un istogramma consiste nella creazione di n partizioni in cui vengono suddivisi i valori di un certo dominio D e, per ogniuna di queste, trova il numero di valori che ne fanno parte.

```
H=zeros([nBins nBins nBins]);
for i=1:size(I,1)
   for j=1:size(I,2)
      p=double(reshape(I(i,j,:),[1 3]));
      p=floor(p/(256/nBins))+1;
      H(p(1),p(2),p(3))=H(p(1),p(2),p(3))+1;
   end
end
```

L’istogramma dei colori fa la stessa cosa ma con le istanze dei colori di una certa immagine: date n partizioni (bins) dello spazio dei colori, conta per ogni partizione il numero di pixel dell’immagine che, per certe istanze di colori, cadono in una data partizione.

Istogrammi delle textures

L'operatore calcola il gradiente della luminosità dell'immagine in ciascun punto, trovando la direzione lungo la quale si ha il massimo incremento possibile dal chiaro allo scuro, e la velocità con cui avviene il cambiamento lungo questa direzione. Il risultato ottenuto fornisce una misura di quanto "bruscamente" oppure "gradualmente" l'immagine cambia in quel punto, e quindi della probabilità che quella parte d' immagine rappresenti un *contorno*, e fornisce anche un'indicazione del probabile *orientamento* di quel contorno.

```
[mag, dir] = imgradient(img, 'sobel');
```

- nelle zone di immagine in cui la luminosità è costante operatore pari 0
- nei punti posti sui contorni consiste in un vettore orientato che punta nella direzione in cui si passa da valori di scuro a valori di chiaro

Punti salienti estratti dall'algoritmo SIFT

Ricaviamo i corrispettivi descrittori delle immagini i quali vengono spesso usati per trovare regioni simili all'interno di immagini differenti e che ci serviranno nell'algoritmo di mathcing.

- estraiamo i SIFT descriptors dei frames

```
[~, d1] = vl_sift(sf1);
[~, d2] = vl_sift(sf2);
```

- Per ogni descrittore d1, trovo il descrittore più vicino in db e ritorno i valori di queste distanze

```
[~, scores12] = vl_ubcmatch(d1, d2);
[~, scores21] = vl_ubcmatch(d2, d1);
```

Combinazione dei risultati

Creazione della matrice simmetrica D delle distanze *(frameSimmDistance.m)*

```
%distanza quaratica per l'istogramma dei colori
distF12c = sqrt((f1cHist-f2cHist)'*A*(f1cHist-f2cHist))/sqrt(2);
%distanza quadratica per l'istogramma delle texture
distF12t = sqrt((f1tHist-f2tHist)'*T*(f1tHist-f2tHist))/sqrt(2);
%distanza come rapporto fra #scores / #distanze
distF12s = 1 - (size(scores12,2)+size(scores21,2))/((size(d1, 2)+size(d2, 2)));

d = 0.6*distF12s + 0.2*distF12c + 0.2*distF12t;
```

MDS

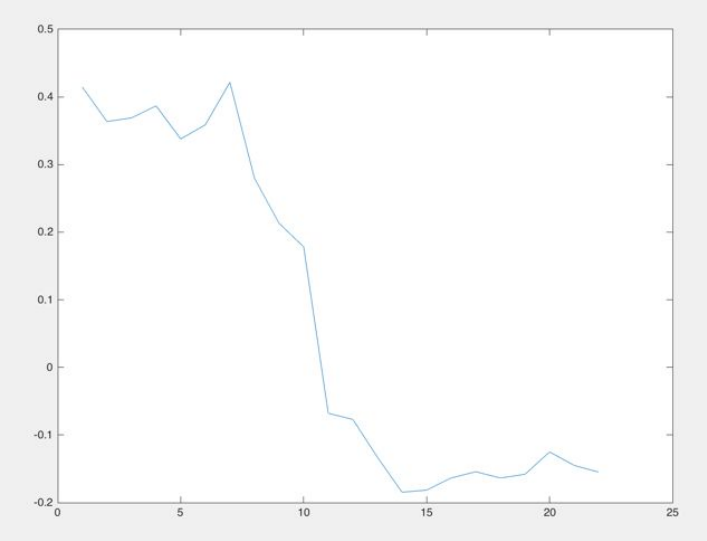
Una volta convertite opportunamente le similarità delle distanze, applichiamo l'algoritmo di MultiDimensional Scaling che associa ad ogni immagine del video un numero reale, riducendo così il video ad una serie temporale. *(MDS.m)*

```
y = mdscale(D,1);
```

- D è la matrice simmetrica delle distanze combinate precedentemente
- p = 1 è il parametro che specifica il numero delle dimensioni in cui stiamo portando il nostro sistema

CUTS

1. Mappa le entries in uno spazio monodimensionale tramite l'algoritmo di MultiDimensional Scaling
2. Estende lo spazio in funzione del tempo



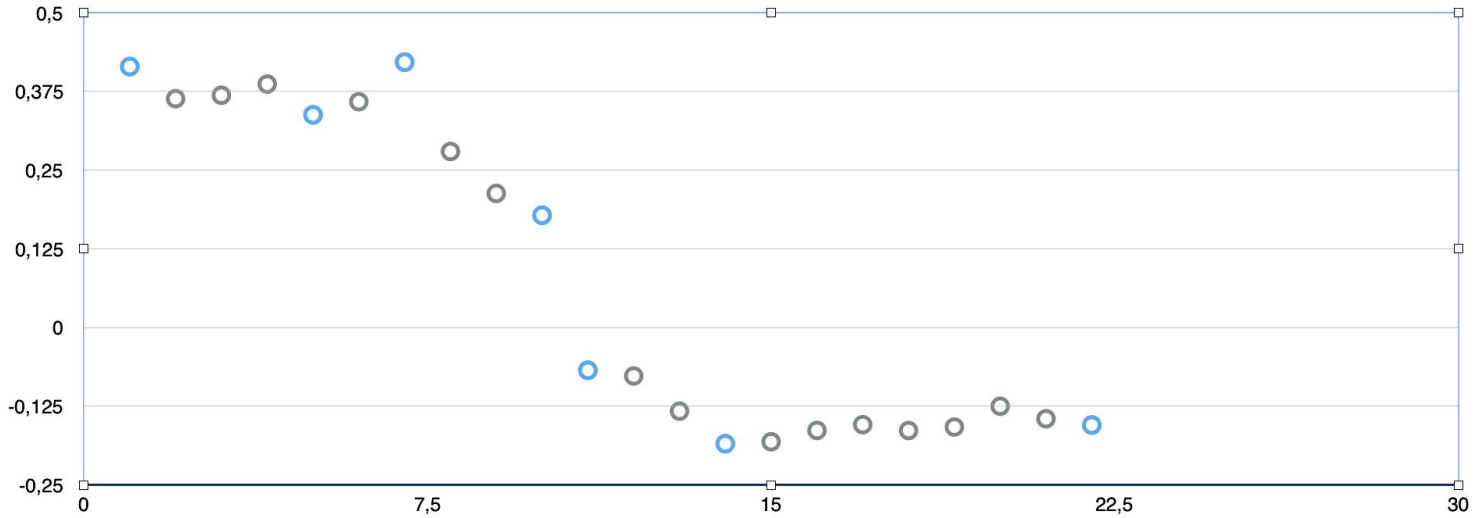
Curve Segmentation

Curva approssimata e quindi segmentata in una serie di rette tramite divisione ricorsiva (*CUTS.m)*

- Considera il primo e l'ultimo punto della curva e li salva: start e end
- Trova il punto, breakpoint, della curva più distante dal segmento individuato dai punti precedenti (start,end)
- Se il punto è più vicino di una data distanza, viene automaticamente scartato
- L'algoritmo chiama ricorsivamente se stesso sui segmenti individuati dalle coppie di punti (start, breakpoint) e (breakpoint, end)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0,4145674960421600	0,3635420599337500	0,3689788716259444	0,3867332969008955	0,3378528354730211	0,3586879333026366	0,4215558833415487	0,2796685630677975	0,2130183582295155	0,1783122176229844	-0,0679607567670531	-0,0769012498141745	-0,13288159454446767	-0,18458289799737474	-0,18148892040697147	-0,16349949747704747	-0,15430817573493636	-0,16361154484864141	-0,15804929704491919	-0,12505670513511818	-0,14491516104757777	-0,15493231634134141

1	5	7	10	11	14	22
0,414567496042160	0,337852835473021	0,421555883341548	0,178312217622984	-0,0679607567670531	-0,184582897997374	-0,154932316341341



Prova su secondo video