Practical block III: Human Behavior Analysis

Interacting with computers using human motion is commonly employed in human-computer interaction (HCI applications). One way to incorporate human motion in HCI application is to use a predefined set of human joint motions i.e., gestures. Gesture recognition has been an active research area.

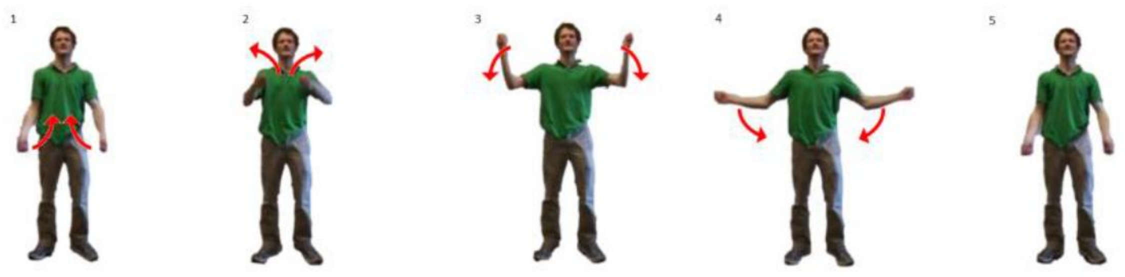
A variety of methods have been proposed for gesture recognition, ranging from the user of Dynamic Time Warping to Hidden Markov Models. DTW measures similarity between two time sequences which might be obtained by sampling a source with varying sampling rates or by recording the same phenomenon occurring with varying speeds. For example, DTW is used in speech recognition to warp speech in time to be able to cope with different speaking speeds. DTW is also used in data mining and information retrieval to deal with time-dependent data. In gesture recognition, DTW time-warps and observed motion sequence of body joints to pre-stored gesture sequences (or some other defined human pose features).

The conventional DTW algorithm is basically a dynamic programming algorithm, which uses a recursive update of DTW cost by adding the distance between mapped elements of the two sequences at each recursion step. The distance between two elements is oftentimes the Euclidian distance, which gives equal weights to all dimensions of a sequences sample. However, depending on the problem a weighted distance might perform better in assessing the similarity between a test sequences and a reference sequence.

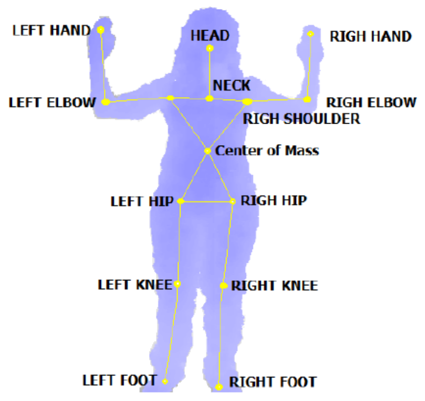
In this practical work we are going to use DTW as a template matching algorithm, in order to find the best match for a test pattern out of the reference patterns, where the patterns are represented as a time sequence of measurements or features obtained from measurements.

Dataset:

The Microsoft Research Cambridge-12 Kinect^{TM} gesture data set consists of sequences of human movements, represented as body-part locations, and the associated gesture to be recognized by the system. The data set includes 594 sequences and 719,359 frames-approximately six hours and 40 minutes-collected from 30 people performing 12 gestures. In total, there are 6,244 gesture instances. The motion files contain tracks of 20 joints estimated using the Kinect^{TM} Pose Estimation pipeline. The body poses are captured at a sample rate of 30Hz with an accuracy of about two centimeters in joint positions (http://research.microsoft.com/en-us/um/cambridge/projects/msrc12/):



The articulated human model is defined by the set of 20 reference points show in the next image:



The gestures can be categorized in the following categories:

1. Crouch or hide (G2)
2. Shoot a pistol (G6)
3. Throw an object (G8)
4. Change weapon (G10)
5. Kick (G12)
6. Put on night vision goggles (G4)
7. Start Music/Raise Volume (of music) (G1)
8. Navigate to next menu (G3)
9. Wind up the music (G5)
10. Take a Bow to end music session (G7)
11. Protest the music (G9)
12. Move up the tempo of the song (G11)

Matlab files:

1. LOAD_FILE -- Load gesture recognition sequence

```
% Input
% file_basename: sequence name such as 'P1_1_1A_01'.
% discard_zero_frames: (optional), if >0, no-skeleton frames at the
% beginning of the sequence are discarded. Default: 1.
%
% Output
% X: (T,80) skeletal frames.
% Y: (T,GN) 0/1 encoding of gesture presence.
% tagset: (1,GN) cellarray of gesture names.
```

X contains all the information about the joints position (x,y,z world coordinates) during *N* frames making different gestures recorded in that sequence. If we want to obtain the length of sequence in frames we can use *T=size(X,1)*. Each. The x,y,z,v values (*v* is not used in this practical work) of the joints is stored in the following order:

1. HipCenter
2. Spine
3. ShoulderCenter
4. Head
5. ShoulderLeft
6. ElbowLeft
7. WristLeft
8. HandLeft
9. ShoulderRight
10. ElbowRight
11. WristRight
12. HandRight
13. HipLeft
14. KneeLeft
15. AnkleLeft
16. FootLeft
17. HipRight
18. KneeRight
19. AnkleRight
20. FootRight

For example if we want to extract the information relative of the Spine in the frame 10 we will use: X(10,(5:8)), where 1:4 are the XYZV variables of the HipCenter, while 5:8 are the XYZV variables corresponding to the Spine.

Y contains the beginning and end frame of each gesture. There could be more than one gesture in a sequence.

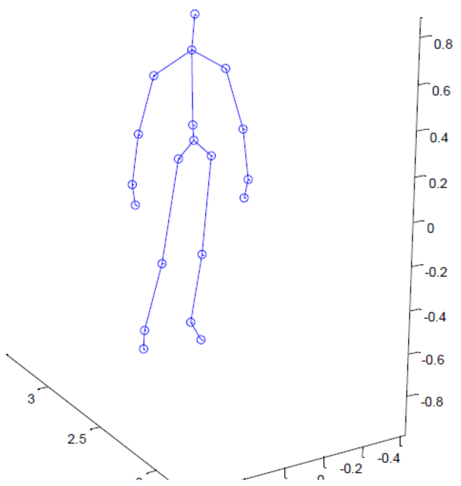
2. GESTURE_CUTS –It extracts the start and finish time (in frames) of each gesture recorded in a sequence.

```
% Input
% X: (T,80) skeletal frames.
% Y: (T,GN) 0/1 encoding of gesture presence.
%
% Output
% gestures: type and action time of each gesture divided
% ngestures: Number of gestures in the sequence
```

Gestures is a matrix composed by ngestures rows and 3 columns. In the first column there is the start time of the gesture (in frames), in the second column there is the finish time, and in the third column there is the gesture type.

3.SKEL_VIS-We can show the position of each joint in a specific frame position.

```
% SKEL_VIS -- Visualize a skeleton in 3D coordinates.
%
% Input
% X: (T,4*NUI_SKELETON_POSITION_COUNT) matrix from load_file.
% tidx: time index >=1, <=T.
% h: (optional) axes handle to draw in.
```



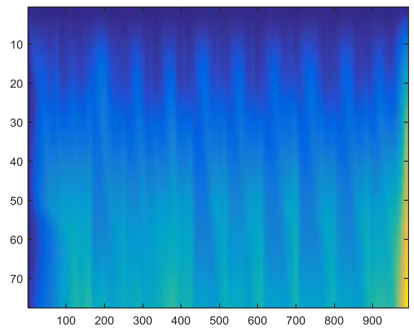
Suggestion: In order to perform the alignment between two patterns, we can employ the command:

```
imresize
B = imresize(A, [numrows numcols])
```

Tasks:

1. LOAD DATA
   Read all the **Microsoft Research Cambridge-12 Kinect^{TM} gesture dataset**.
   Write a function "[data, tagset] = loadAll();" using the provided function "LOAD_FILE" (provided with the data-set) extracting the dataset and the tags related to the actions. data will be a structure containing 594 sequences described by the vectors X and Y.
   for instance plot(data(13).X(:,1)), illustrate the variation of the X axis (all the frames) of the sequence 13
2. PREPARE THE GROUND TRUTH
   Code a new function called [gt] = getSamples(data, chosen_class); For each sequence the function
   a. separates the beginning and end of each gesture (remember, there could be zero or more than one gesture in a single sequence). For this purpose code a function called "[gest, ngest] = gestureCuts(data(i).X, data(i).Y);" where "ngest" is the number of gesture and "gest" is a list of gestures, of size (ngest,3), consisting of a gesture label, and a beginning and end frame
   b. The function then select and concatenate only the gestures of a chosen label (for instance the gesture 12), use the beginning and end frame to select only the subsequence of the signal contained between beg-end.
   c. the resulting variable "gt" is a structure composed of the following vectors:
      subSeq = sequence cuts size(n_ gestures,80) where 80 is 20xXYZV
      indices = beg-end frames of each sequence size(n_ gestures)
3. Code a multi-dimensional version of the DTW using Euclidian distance as cost function.
   a. The original version of the algorithm is designed for two mono-dimensional signal (a sample and a model (https://it.wikipedia.org/wiki/Dynamic_time_warping). Extend the original version, in such way that a multi-dimensional input (80 rows) can be used. TIP: the weight given to each cell of the DTW matrix can be computed as distance between each pair of signals.
   b. Use as "model" one single sequence extracted from the ground truth (you may want to make several tests by changing the "model" sequence in order to test the robustness of the approach. Use as sample all the remaining sequences of the dataset.

   c. The DTW computes a table. Compute the minimum path (backtrack) corresponding to the optimal position of the model within the sample. The minimum path and DTW costs indicates where the "model" fits within the "sample". The DTW function returns the minimum cost and the position of the "model" within the "sample", given by the extremes of the minimum path (backtrack).



   Compare the result against the ground truth, in order to validate if the sequence has been found correctly or not.

4. Please note that, when the DTW is applied to the whole data set, it is possible that the specific gesture is not contained within the sequence. In such case the DTW will return a result having a cost very high. In order to identify if the gesture is contained, a threshold of the cost of the DTW should be identified.
   a. Define a cross-validation strategy in order to find the threshold required for discarding the incorrect detection of the gesture.
5. Code: Write a demo matlab file for loading a model of a gesture category and showing its correct recognition This means defining a model of a category and then test this model against samples of the different categories, showing how DTW recognize the correct gesture for a given threshold value.
6. Report: short report just describing implementations and the tests.

Deliverable

A single report in PDF format of a maximum of three pages must be delivered for the first practical block. The matlab code of the session should be delivered as well

- **IN CASE YOU CAN ACCESS TO THE CAMPUS VIRTUAL**

please upload in the corresponding task your file before the dead line

- **IN CASE YOU HAVE NO ACCCESS TO THE CAMPUS VIRTUAL**

CREATE A DROPBOX link with the zip containing the whole deliverable session and send to simone.balocco@ub.edu for evaluation Subject: MAI PR3).
IMPORTANT, do not send the ZIP file directly.