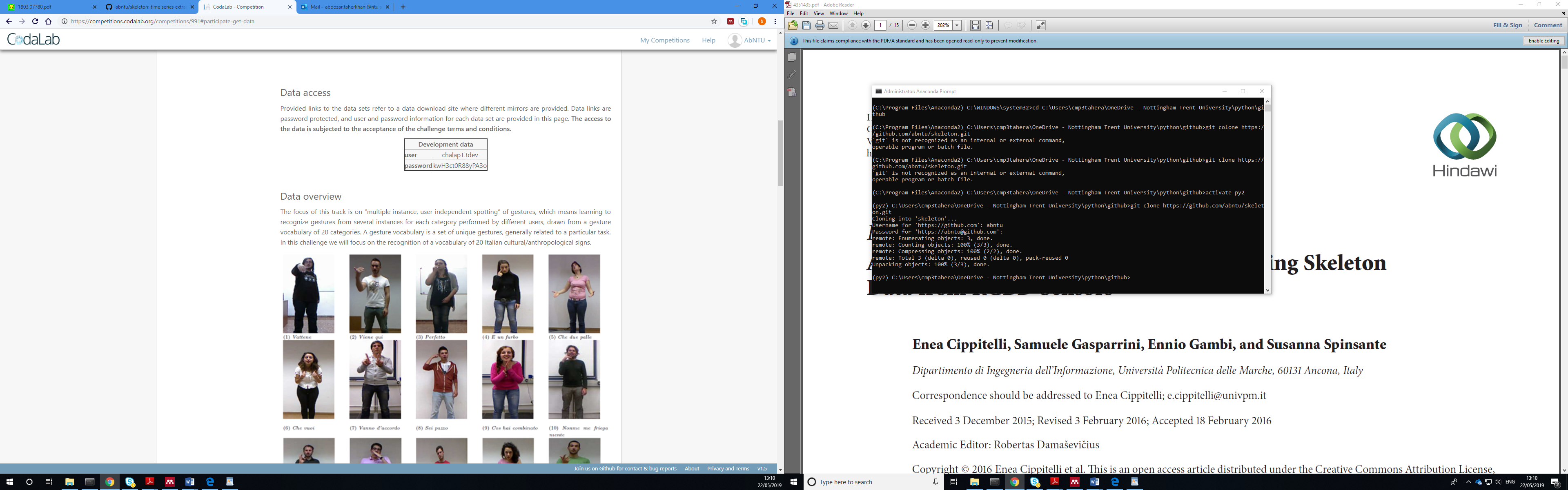
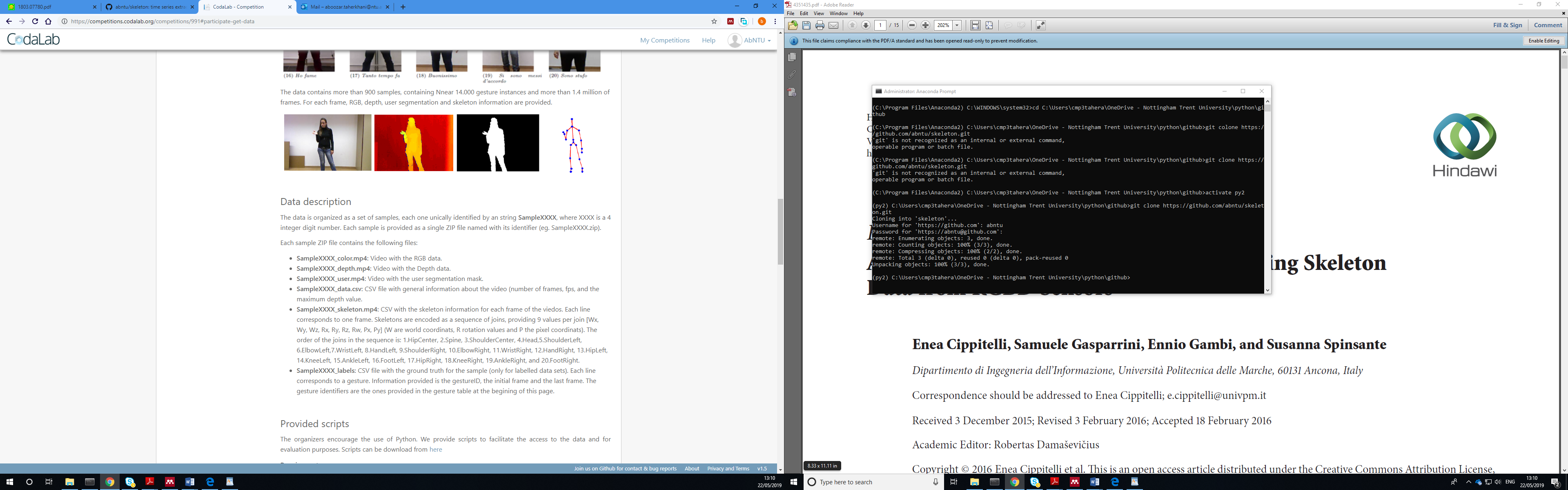
Track 3: Gesture Recognition





## **Provided scripts**

The organizers encourage the use of Python. We provide scripts to facilitate the access to the data and for evaluation purposes. Scripts can be download from [here](https://competitions.codalab.org/competitions/edit_competition/)

**Requirements**

* OpenCV 2.4.8
* Python Imaging Library (PIL) 1.1.7
* NumPy 1.8.0

### **Data access**

On the file **ChalearnLAPSample.py** there is a class **GestureSample** that allows to access all information from a sample. In order to open a sample file, use the constructor with the ZIP file you want to use:

>> from ChalearnLAPSample import GestureSample

>> gestureSample = GestureSample("SampleXXXX.zip")

With the given object you can access to the sample general information. For instance, get the number of frames, the fps or the max depth value:

>> numFrames=gestureSample.getNumFrames()

>> fps=gestureSample.getFPS()

>> maxDepth=gestureSample.getMaxDepth()

Additionaly we can access to any information of any frame. For instance, to access the RGB, depth, and user segmentation information for the 10th frame, we use:

>> rgb=gestureSample.getRGB(10)

>> depth=gestureSample.getDepth(10)

>> user=gestureSample.getUser(10)

Finally, we can access to an object that encodes the skeleton information in the same way:

>> skeleton=gestureSample.getSkeleton(10)

To get the skeleton information, we have some provided functionalities. For each join the [Wx, Wy, Wz, Rx, Ry, Rz, Rw, Px, Py] description array is stored in a dictionary as three independent vectors. You can access each value for each join (eg. the head) as follows:

>> [Wx, Wy, Wz]=skeleton.getAllData()['Head'][0]

>> [Rx, Ry, Rz, Rw]=skeleton.getAllData()['Head'][1]

>> [Px, Py]=skeleton.getAllData()['Head'][2]

The same information can be retrieved using the especific methods:

>> [Wx, Wy, Wz]=skeleton.getWorldCoordinates()['Head']

>> [Rx, Ry, Rz, Rw]=skeleton.getJoinOrientations()['Head']

>> [Px, Py]=skeleton.getPixelCoordinates()['Head']

Additionally, some visualization functionalities are provided. You can get an image representation of the skeleton or a composition of all the information for a frame.

>> skelImg=gesture.getSkeletonImage(10)

>> frameData=gesture.getComposedFrame(10)

To visualize all the information of a sample, you can use this code:

import cv2  
from ChalearnLAPSample import GestureSample  
  
gestureSample = GestureSample("Samplexxxx.zip")  
cv2.namedWindow("Samplexxxx",cv2.WINDOW\_NORMAL)   
for x in range(1, gestureSamle.getNumFrames()):  
img=gestureSample.getComposedFrame(x)  
cv2.imshow("Samplexxxx",img)  
cv2.waitKey(1)  
del gestureSample  
cv2.destroyAllWindows()

### **Evaluation**

On the file **ChalearnLAPEvaluation.py** there are some methods for evaluation. The first important script allows to export the labels of a set of samples into a ground truth folder, to be used to get the final ovelap value. Let's assume that you use the samples 1 to 10 for validation purposes, and have a folder **valSamples** with the files Sample0001.zip to Sample0010.zip as you downloaded from the training data set. We can create a ground truth folder **gtData** using:

>> from ChalearnLAPEvaluation import exportGT\_Gesture

>> exportGT\_Gesture(valSamples,gtData)

This method exports the label files and data files for each sample in the **valSample** folder to the **gtData** folder. This new ground truth folder will be used by evaluation methods.

For each sample, we need to store the gesture predictions in a CSV file in the same format that labels are provided, that is, a line for each gesture with the gestureID, the initial frame and the final frame. This file must be named as Samplexxxx\_predictions.csv. To make it easy, the class GestureSample allows to store this information for a given sample. Following the example from last section, we can store the predictions for sample using:

>> from ChalearnLAPSample import GestureSample

>> gestureSample = GestureSample("SampleXXXX.zip")

Now, if our predictions are that we have the gesture 1 from frame 102 to 203 and gesture 5 from frame 250 to 325, and we want to store predictions in a certain folder **valPredict**, we can use the following code:

>> gestureSample = GestureSample("SampleXXXX.zip")

>> gestureSample.exportPredictions(([1,102,203], [5,250,325]),valPredict)

Assuming previous defined paths and objects, to evaluate the overlap for a single labeled sample prediction, that is, prediction for a sample from a set where labels are provided, we can use:

>> overlap=gestureSample.evaluate(([1,102,203], [5,250,325]))

Finally, to obtain the final score for all the predictions, in the same way performed in the Codalab platform, we use:

>> from ChalearnLAPEvaluation import exportGT\_Gesture

>> score=evalGesture(valPredict,gtData)

## **Data download**

All the scripts are provided though the [downloads page](http://sunai.uoc.edu/chalearnLAP).

Download the data from one of the mirrors:

* Development:
  + [Training Data](http://sunai.uoc.edu/chalearnLAP/)
  + [Validaton Data](http://sunai.uoc.edu/chalearnLAP/)
* Final Evaluation:
  + [Challenge Data](http://sunai.uoc.edu/chalearnLAP/)

Reference

<https://competitions.codalab.org/competitions/991#participate-get-data>

<http://gesture.chalearn.org/2014-looking-at-people-challenge/data-2014-challenge>