Online Signature Biometrics Lab – Report

Balázs Márk Bódis, Zsófia Sára Budai

Outline

1. Introduction

The objective of this session is to DEVELOP and EVALUATE an online signature recognition algorithm. According to the theory sessions, signature recognition systems can be divided into two categories:

- **Off-line**: the input is a static image of the signature.
- **On-line**: the signature is acquired using a specific digital sensor which includes the static image and dynamic signals related with the way the signature was done: x,y coordinates and pressure as a function of time.

Figure 1 shows a block diagram of a typical online signature recognition algorithm where [x,y,p] are the captured signals by the sensor (Cartesian coordinates and pressure), f_t is the feature vector of the query signature to be compared with the f_c feature vector of the signature stored in the database (claimed identity).

In this session we will assume that the data is available (previously acquired) and we will focus on the development of two modules:

- Feature Extraction Module.
- Matcher.

You must complete the tasks proposed in this document and answer the questions included.

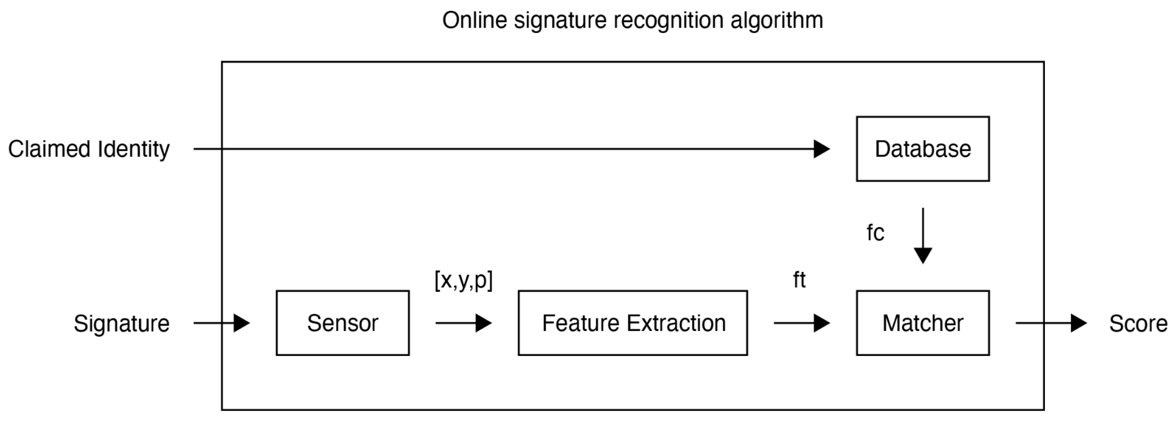


Figure 1. Block Diagram of a typical online signature recognition system

2. Data

For the practice we will use 50 users from the BiosecurID database. Each of the users have 28 signatures acquired in 4 sessions with a time lapse of 2 months. From the 28 signatures, 16 are genuine (4 per session) and 12 are forgers (3 per session). In this practice we will only consider the genuine signatures.

Each of the signatures is stored in .mat file which contains three vectors of same length with the x, y coordinates and the pressure as functions of time.

The formatting of the files is uXXXXsYYYY_sgZZZZ.mat:

- XXXX: user number
- YYYY: session number
- ZZZZ: signature number

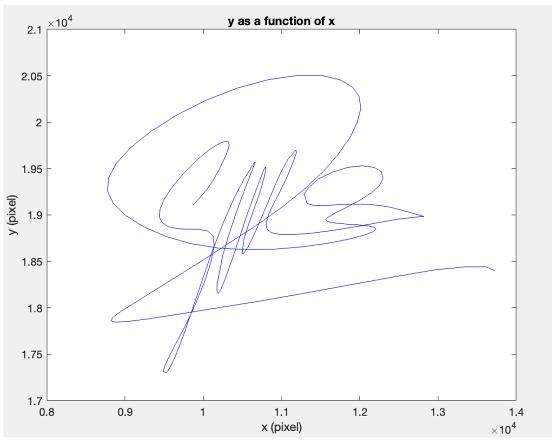
The GENUINE signatures of each session are those with ZZZZ=[0001,0002,0006,0007].

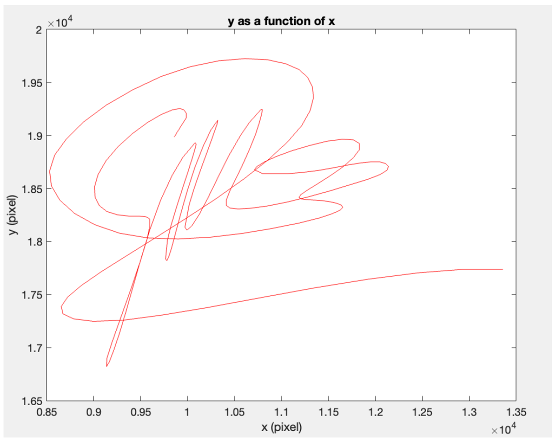
The signatures with ZZZZ=[0003,0004,0005] are the FORGERS and they will NOT be used in this practice.

QUESTION. Choose a signature (from a random user) and show (assuming that the sensor has a 200 samples/second acquisition rate):

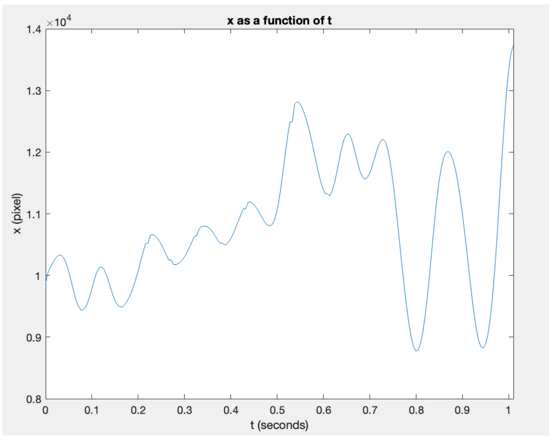
- Signal y as a function of signal x.
- Signal x as a function of time.
- Signal y as a function of time.
- Signal p as a function of time.

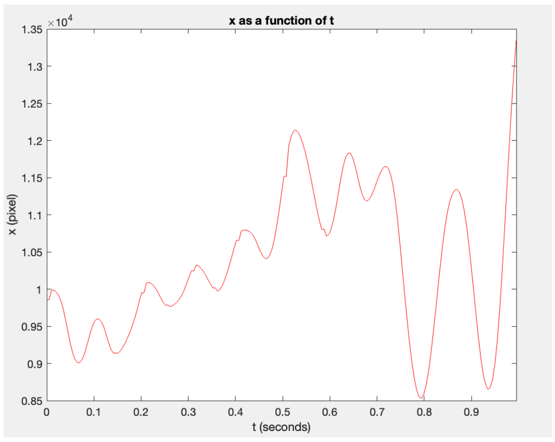
y as a function of x



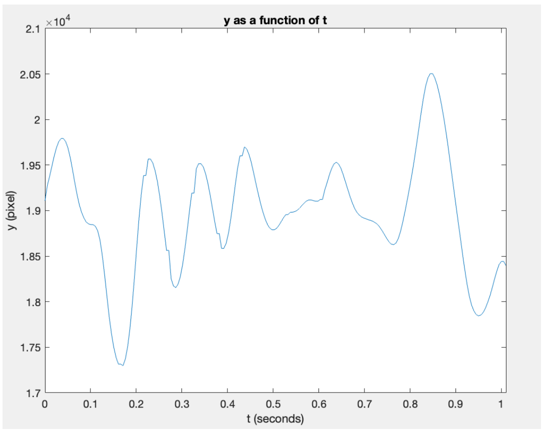


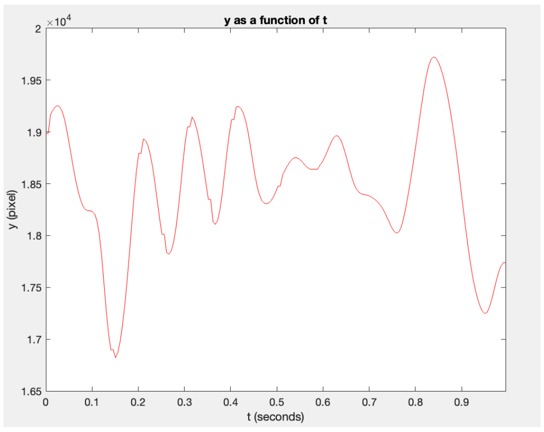
x as a function of t



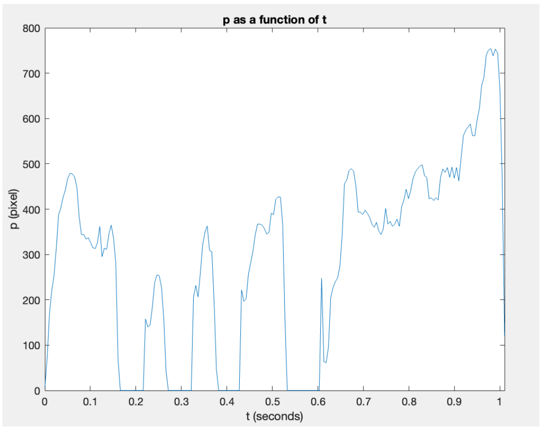


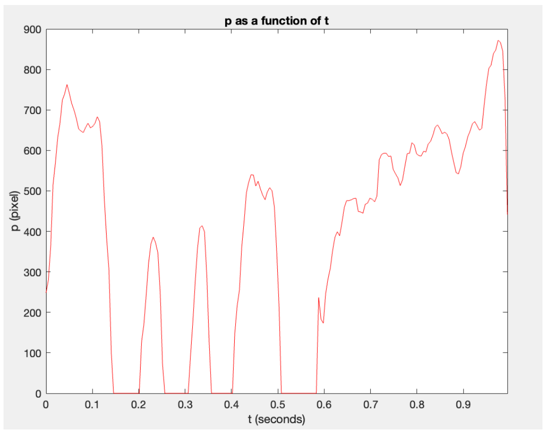
y as a function of t





p as a function of t





Repeat the task with another signature of the same user.

QUESTION. Are the different signals reasonable? Do they have the same length? Why?

Blue (random signature):
Yes, they all look reasonable.
The length of the curves are not the same, but the number of samples is the same for all 4 plots. In the last 3 plots the time is normalized with the sample frequency of the input device.
All 4 curves have the same number of samples, since each datastream was recorded with the same time intervals. The time variable is generated using the sample frequency and the number of samples.
Blue vs Red (same user, random signatures (different session)):
As we can see on the first pair of plots the 2 signatures do look familiar. This is also reflected in the following pairs, as all x,y and p look very similar in both timing and the absolute value of the signal.

3. Feature Extraction

The comparison of signals with different lengths is not trivial. Therefore, we will extract 4 global parameters of each of the signatures. So, each signature will be represented by a feature vector with fixed size equal to 4. These parameters are:

- Total duration of the signature: T
- Number of *pen-up* (number of times the pen was lifted). It means the number of times (not the number of samples) that p is equal to 0.
- Duration of *pen-down* (signal p is different to 0) T_d divided by the total duration T: T_d/T
- Average pressure in *pen-down* (signal p is different to 0).

You have to develop 4 functions to extract each of the parameters:

- T=Ttotal(x)
- Npu=Npenups(p)
- Tpd=Tpendown(p)
- Ppd=Ppendown(p).

According to those functions, we will develop a new function with input data (x,y,p) of a given signature and output data the feature vector containing the 4 parameters (FeatVect=featureExtractor(x,y,p)).

Based on your function featureExtractor you have to develop a program (ProcessBiosecurID.m) to extract all the feature vectors from the database and store it in a matrix with 3 dimensions:

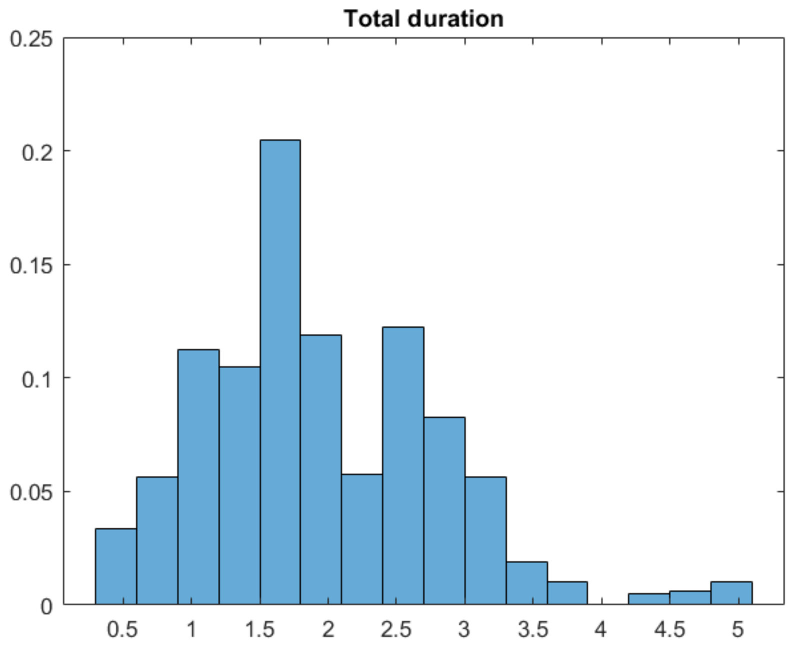
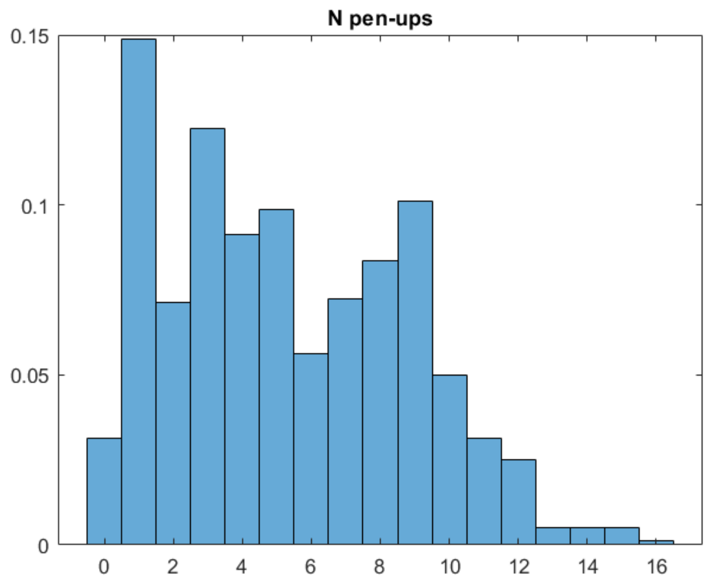
- Dimension 1: number of user (1:50)
- Dimension 2: number of signature (1:16)
- Dimension 3: number of parameter (1:4)

You have to save this matrix into the file BiosecurIDparameters.mat

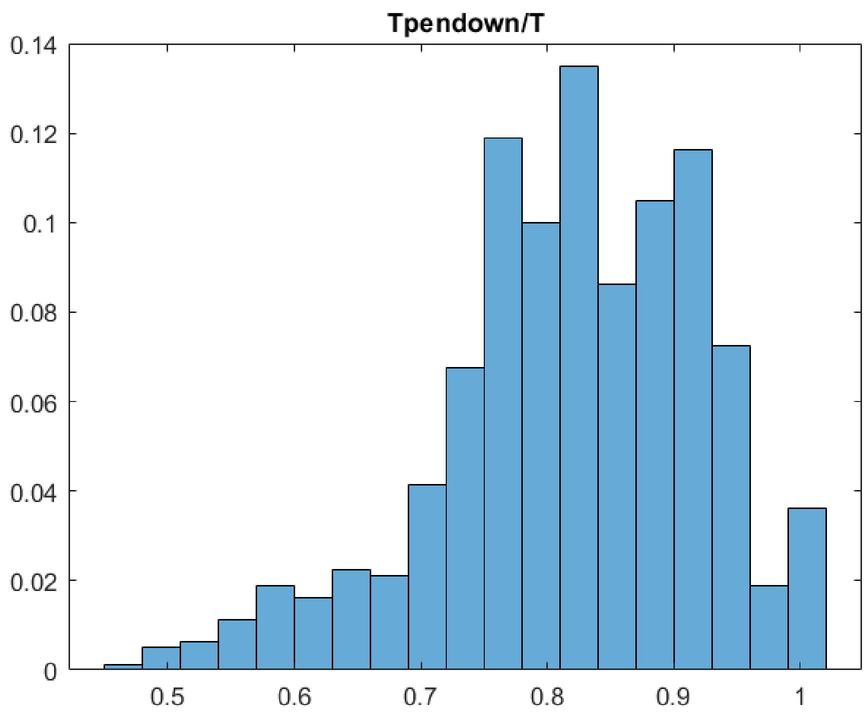
Once you have the file BiosecurIDparameters.mat, you have to plot the distributions normalized between 0 and 1 (dividing by the total number of points of the distribution) for each of the 4 parameters.

You can use the Matlab functions hist.m and histc.m

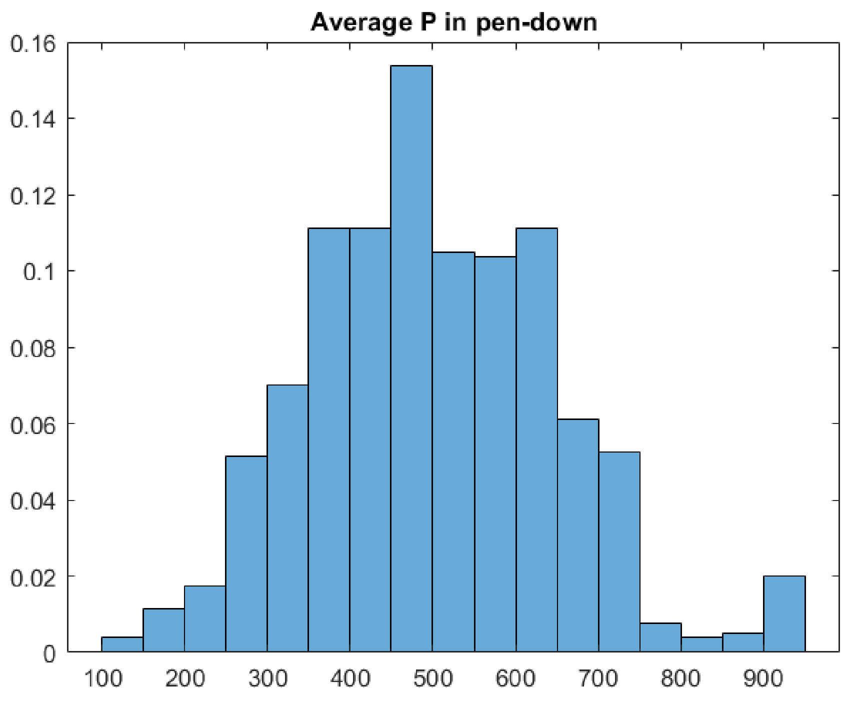
QUESTION: Plot the 5 distributions.

Total duration

N pen-ups


T pen-down / T



Average P in pen-down



4. Performance Evaluation

We will evaluate the performance of our system according to the number of signatures N in the enrollment set (N=1, N=4 and N=12).

The similarity score between a query/test signature and the enrollment signatures (signatures in the database) will be the Euclidean distance between feature vectors (vectors with 4 parameters). The final score will be the average score of the N comparisons (comparison between the query/test sample and the N enrollment samples).

You have to develop the function Score=Matcher(test,Model) where:

- Score: is the final score of the comparison.
- test: is the feature vector of the query/test signature (1x4)
- Model: is a matrix containing the feature vectors of the signatures enrolled in the database. Therefore, this matrix contains Nx4 values in which N is the number of signatures enrolled for the claimed identity.

There are two cases to be analyzed:

Genuine Scores: scores obtained when you compare a signature with his real enrolled identity (claimed identity = enrolled identity). So these users should be accepted by the system. For each user you will use N signatures as enrolled samples and the rest for testing:

- For N=1 we will have SG=15 genuine scores.
- For N=4 we will have SG=12 genuine scores.
- For N=12 we will have SG=4 genuine scores.

For each of the scenarios (N=1,4,12) you have to save all the genuine scores into a matrix (with dimension 50xSG). Each of the three matrixes will be stored into a .mat file with name: GenuineScores_N.mat.

Impostor Scores: scores obtained when you compare a signature with the enrolled samples of other users (claimed identity ≠ enrolled identity). So these users should be rejected by the system. In this case, we will compare one signature of each user (the first one) with the models of the rest of the users (excluding the genuine case). Therefore, we will obtain SI=49 impostor scores for each user and each scenario (N=1,4,12).

For each scenario (N=1,4,12) these impostor scores will be saved into a matrix with dimensions 50xSI (50x49). Each of the three matrixes will be stored into a .mat file with name: ImpostorScores_N.mat.

Once we obtain the genuine and impostor scores, we will evaluate the performance of our system for each of the three scenarios (N=1,4,12) as a function of: FAR/FRR, EER and DET curves.

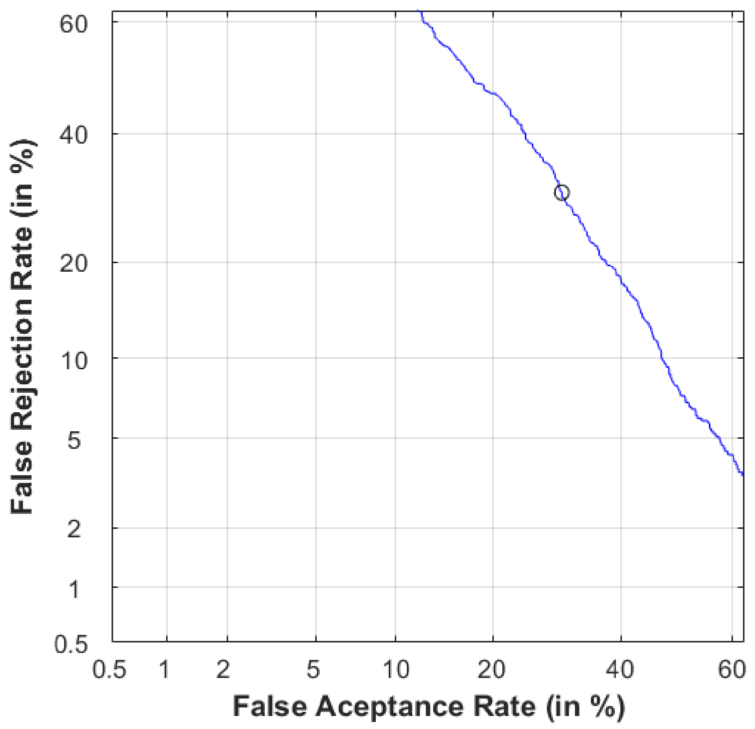
To obtain these performance metrics you will have available the next functions:

[EER]=Eval_Det(GenuineScores, ImpostorScores, 'b')

- EER: value of the Equal Error Rate (error when FAR and FRR are equal)
- GenuineScores: the scores from target or genuine comparisons. These scores are obtained after applying the following normalization: GenuineScores = 1./(GenuineScores_N+0.00000001)
- ImpostorScores: the scores form non target or impostor comparisons. These scores are obtained after applying the following normalization: ImpostorScores =1./(ImpostorScores_N+0.00000001)

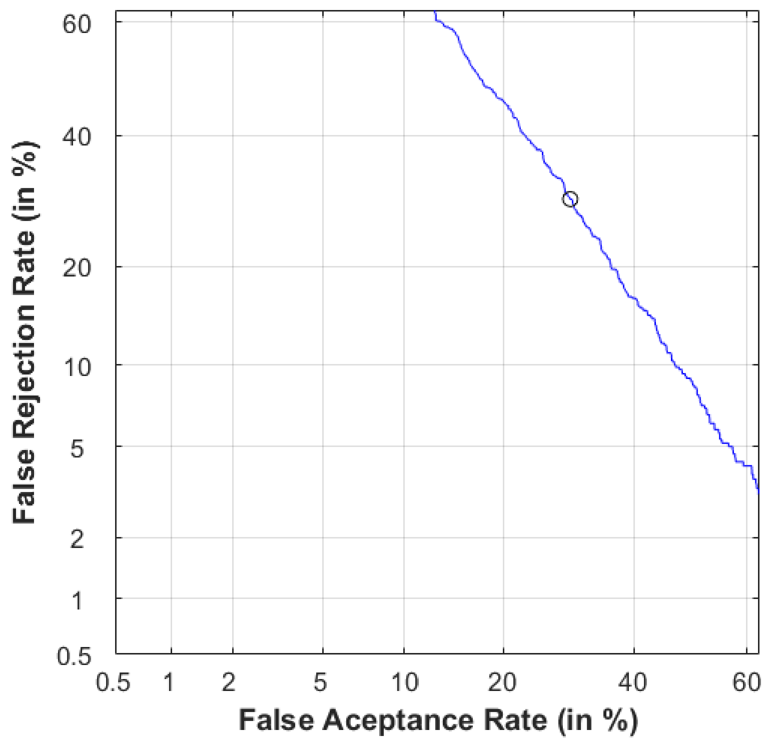
QUESTION. Plot the performance graphics (DET curves) using the genuine and impostors score stored in their respectively matrixes (for each of the scenarios N=1,4,12). Indicate the EER value.

N=1

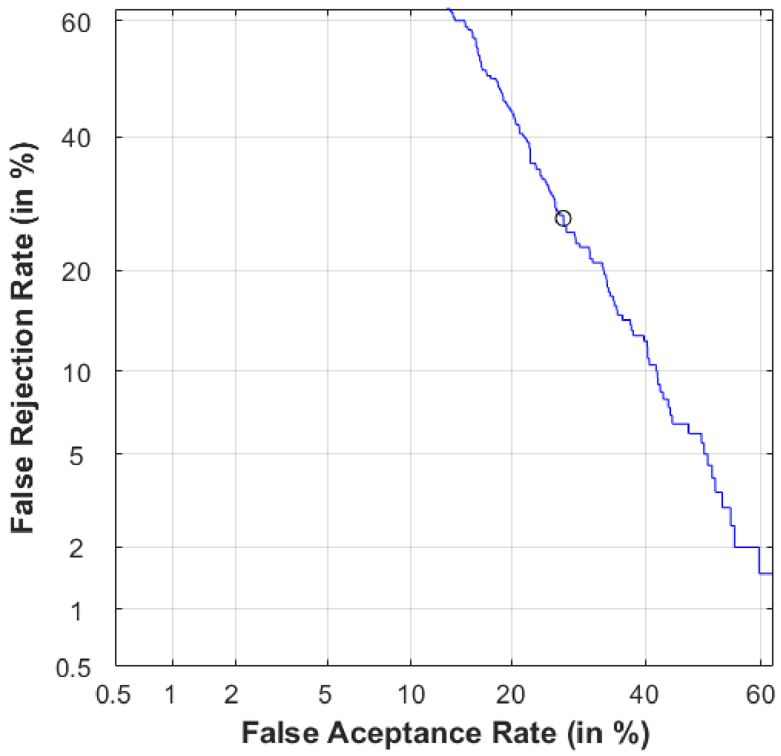


EER1 = 30

N=4
EER4 =   29.5000



N=12
EER12 =   27



QUESTION. According to the results, are they reasonable? What metrics are more illustrative? When do you obtain the best performance?

The results obtained may not be excellent, but they are reasonable considering that only global features were used in the model. The EER metric effectively demonstrates the model's performance. To improve the model's robustness, more data from users can be used, and local features can be extracted and incorporated to enhance the model's resilience and achieve better performance. It is demonstrated in the extra work 2.

With all the previous exercises done correctly you can obtain a mark up to 6 points out of 10.

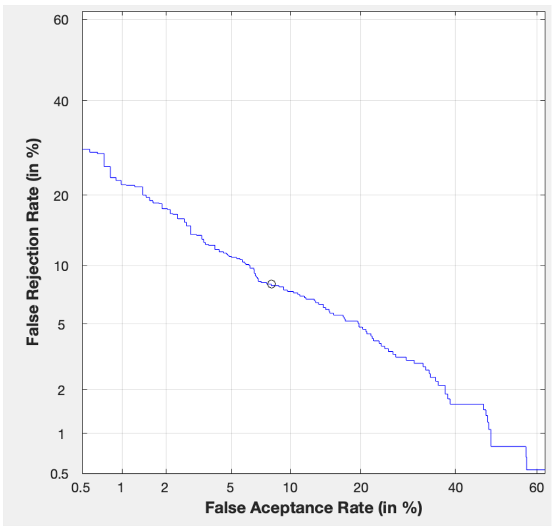
Extra work 1: If you want to obtain a mark up to **8 points out of 10** you should complete one of the following points:

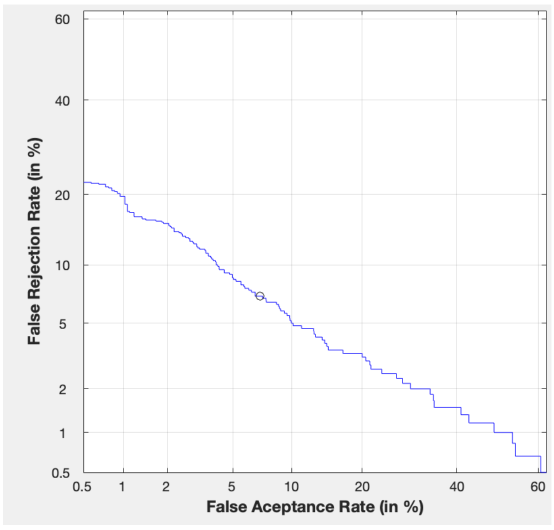
We skipped extra work 1.

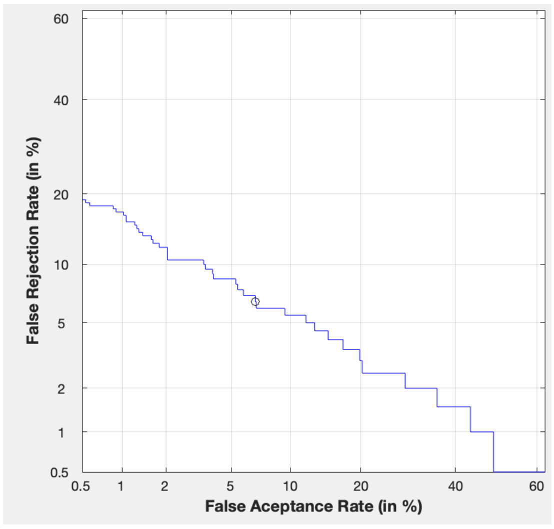
Extra work 2: If you want to obtain a mark up to **10 points out of 10** you should do the following (directly, without doing the Extra work 1):

Implementation of the Local Feature Matcher is in **localFeatureMatcher.m** and **localFeatureExtractor.m**.
The modified **Evaluation.m** uses the local features extracted to calculate the performance of the system with the local features.

QUESTION. Plot the performance graphics (DET curves) using the genuine and impostors score stored in their respectively matrixes (for each of the scenarios N=1,4,12). Indicate the EER value.

N=1
EER=8.1333
	

N=4
EER=7.0000
	

N=12
EER=6.8878
	

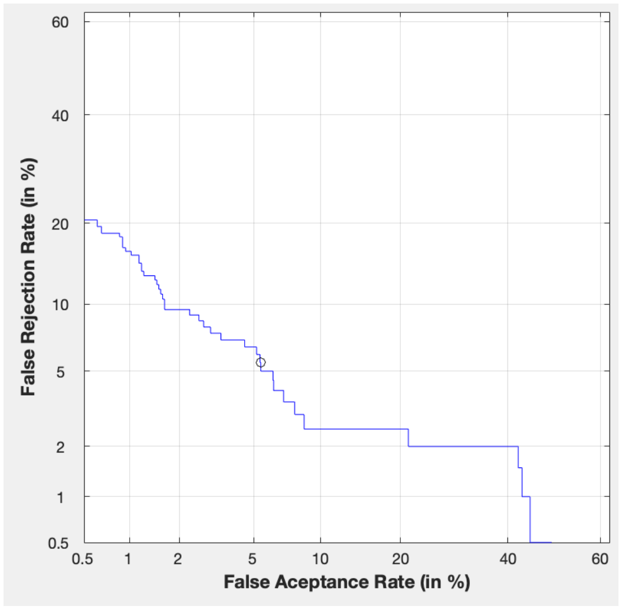
QUESTION. According to the results, are they reasonable? What metrics are more illustrative? When do you obtain the best performance?

The results are still not perfect. The false acceptance rate is still very high. There is a more significant reduction in false rejection rate. Overall it is clearly visible that using the local features we can achieve better scores.

While we can draw conclusions from the graphs, the EER metric is more easily comparable. From this we can see that our local feature matcher outperforms our best score with the global matcher even with a single sample.

We tried to introduce some parameters into the local feature matcher, like using different weights for the different features (for example the derivative of the location is more important than the location…) for this we used a `weights=[1,1,1,3,3,3,1,1,1]`vector.

In this example the first order derivatives are considered 3 times as important as the location and pressure.
The distance is calculated as `D=dist/sum(weights)`.
The scores for these (1,3,1) weights are worse: EER1: 8.6667, EER4: 7.8333, EER12: 8.3265
The scores for these (2,1,1) weights are better: **EER1: 7.7333, EER4: 6.0000**, EER12: 6.4830
The scores for these `[3,3,6,1,1,1,1,1,1]` weights are our best:
EER1: 7.7333,
EER4: 6.6429,
EER12: 5.5000
but interestingly it is only an improvement when we have N=12 samples.
These indicate that the parameters x,y,p are more important for determining whether a signature is genuine or not. Our best score with N=12 is the following DET curve:



Pressure seems to be very important in filtering fake entries, but more samples are needed to use it effectively.