signal processing
department of

http://zs.utia.cas.cz

Application Note

ÚTIA Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

Noise Cancellation Using QRD RLS Algorithms

Raissa Likhonina, Jiří Kadlec
likhonina@utia.cas.cz kadlec@utia.cas.cz
phone: +420 2 6605 2216
The Czech Academy of Sciences, Institute of Information Theory and Automation

Revision history:

Rev.	Date	Author	Description

Acknowledgements:

This work has been supported by the ECSEL JU project SILENSE "(Ultra)Sound Interfaces and Low Energy iNtegrated Sensors" Project No.: ECSEL JU 737487-2 and MSMT 8A17006 (Ministry of Education Youth and Sports of the Czech Republic). See web: http://www.silense.eu/

Table of contents

Table of figures

1. Summary

1.1 Objectives

This Application Note aims to simulate a noise cancellation problem with MATLAB tools. This is purposed for pre-processing process for final gesture recognition application. It also shows advantages and disadvantages of an approach used for a noise cancellation.

In applications for gesture recognition the signals can reflect and be detected not only from a desired source (a hand), but from the environment as well, which creates undesired noise and hardens the process of precise gesture identification (see Figure 1).

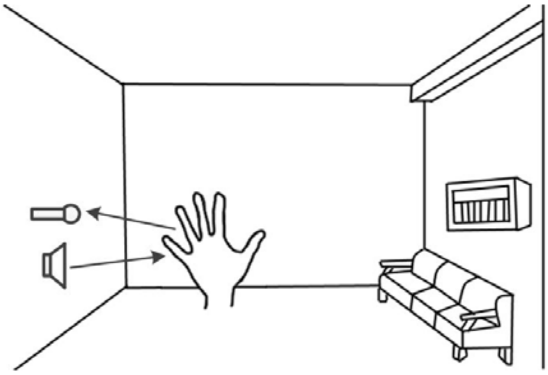

Figure 1.1. Example of gesture recognition application

Therefore, it is essential to eliminate the signals, which come from other static sources than a hand. For these purposes echo cancellation methods can be used. Echo cancellation is widely and successfully applied in telephony in a way of preventing echo from being created or removing it after it is already present. We will assume that a hand will appear just for a short time period and the additional reflections will act as an additional short period “disturbance”.
The echo cancellation in this specific case will be based on QRD algorithm with double precision arithmetic and exponential forgetting. The QRD algorithm is also called as an information filter without square root operations. It is based on QRD decomposition of the input/output information matrix. The recursively updated QRD factorization of the information matrix helps to avoid the problem with loss of positive definiteness of the information matrix due to rounding errors and, thus, provides a numerically stable solution.
The exponential forgetting is used instead of directional forgetting to keep the perspective of reduction of the computation time by applying the QRD version of the Lattice algorithm. QRD Lattice works only with the exponential forgetting with a constant exponential forgetting factor.

Our simulation of gesture recognition problem can be seen in the following block diagram (Fig. 1.2).

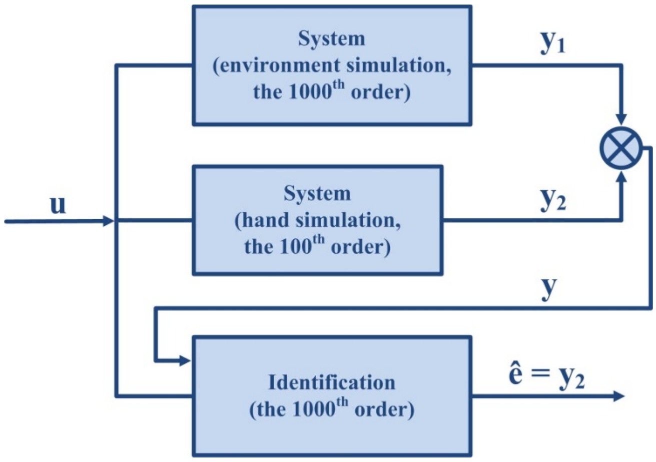

Figure 1.2. Simulation of gesture recognition problem

The diagram shows that the whole process of simulation consists of three main parts: environment simulation, hand simulation and identification block, all based on linear finite impulse response (FIR) models.

All three blocks have a common input **u**. Using this input the environment and hand models calculate outputs $\mathbf{y_1}$ and $\mathbf{y_2}$ respectively. These outputs are summed and the result of summation **y** goes like an input to identification block, which estimates parameters of the model and calculates prediction and filtration error. It is worth saying that the main interest is presented by prediction error $\hat{e}$, because these data will be an input data for beamforming and further processing for final steps of gesture recognition. For making it possible prediction error has to have the same development as output $\mathbf{y_2}$, because actually it is the hand model, which causes the disturbance and increases prediction error. The absence of a hand is modelled by a hand simulation model with zero coefficients. The short term appearance of a hand is modelled by setting coefficients of the hand simulation model to nonzero values.

This application note and the related evaluated package provides reference base for this development. The evaluated package requires Win7 (64b) or Win10 (64b) PC.

Included scripts and precompiled algorithms can be used with MATLAB R2018b or higher.

Alternatively, the package can be also used without MATLAB. All scripts are precompiled by the MATLAB Compiler (R2018.b) as packages supporting installation and standalone execution on Win 7, 64b PC without MATLAB.

1.2 Directory

Download the package and unzip to separate folder. Example

C:\VM_07\R2018b\experiments

In this folder, you will find MATLAB files executing algorithms and a sub-folder ***"private"*** with MATLAB **mexw64** files with precompiled algorithms.

It also comprises sub-folders "**experiments**" for installing and using precompiled scripts.

2. Demos

During simulation, several experiments with different inputs and parameters of environment and hand models have been executed to evaluate the identification process. Among these experiments, which will be described in details in next chapters, are the following:

- Experiment 1: an input signal is a white noise; time-invariant environment model, time-variant hand model;
- Experiment 2: an input signal is pulses with period of 500 samples; time-invariant environment model, time-variant hand model;
- Experiment 3: an input signal is pulses with period of 500 samples; both environment and hand models are time-variant;
- Experiment 4: an input signal is pulses with period of 500 samples; time-variant environment model; time-variant hand model with longer hand presence in front of detectors.

2.1 Simulation results: experiment 1

The experiment can be run using a MATLAB code under name **experiment_1.m** or a precompiled script **experiment_1.exe** without using MATLAB.

In this experiment white noise is used as an input signal, because it is rich in respect of information and therefore excites the system persistently. This makes the identification process for the algorithm simpler.

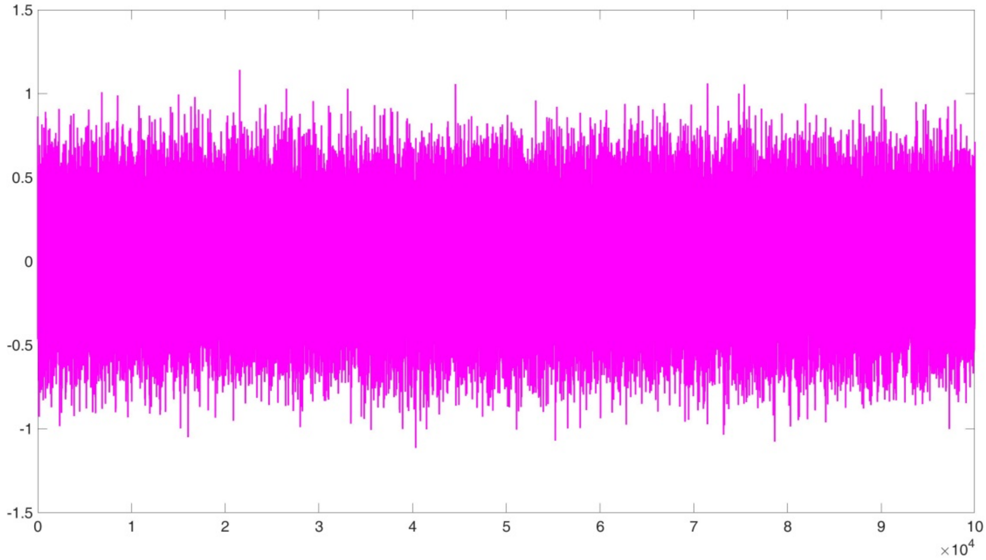

Figure 2.1.1. Input signal – white noise

To simulate the environment a regression model of the 1000th order is used. All 1000 coefficients are constant, i.e. they do not change during time. The first 500 coefficients are set to zero to allow the identification process to learn and to find correct values of estimated parameters. Other parameters are random values. The environment model is burdened with noise of 0.05 to make the situation corresponding to the reality.

The hand model is also a regression model. It has 100 time-variant parameters and noise of 0.05. All values of parameters are random.

During simulation, which has 100 000 steps, "the hand" appears three times: firstly at time step 10 000, then at time step 50 000 and 80 000. The first and the third appearance lasts only 50 samples, while the second one lasts 5 000 samples.

During this experiment two values of exponential forgetting are used: φ=0.999 and φ=0.9999999999. Note that the values of exponential forgetting can be changed in a MATLAB script by changing a value of **fim**.

The results of identification given φ=0.999 are shown in Figure 2.1.2.

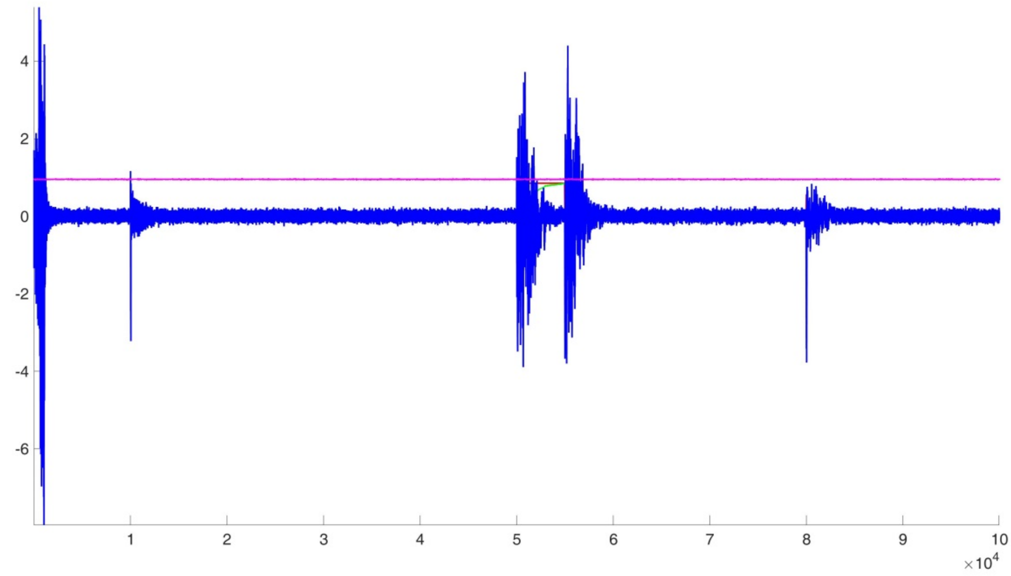

Figure 2.1.2. Identification results: φ=0.999

The magenta line on the graph shows an input signal, while the blue one is a prediction error. It is clearly from the graph that at first the system needs some time to learn to identify parameters correctly. It is the reason of large values of prediction error in the beginning of identification. After a while the process of identification becomes smooth and estimates parameters correctly. The moment, "the hand" is placed, is characterized with increasing values of prediction error again. However, there are four peaks on the graph of error increase, though "the hand" is placed only three times. The short appearance of the hand at time step 10 000 and 80 000 causes two increases of prediction error. However, a longer presence of the hand at time step 50 000 causes two increases of prediction error as well due to the fact, that given φ=0.999 the system learns fast and after a short time it takes the presence of the hand as a normal situation and after its disappearance later on it gives the increase of prediction error again.

The situation changes when exponential forgetting is set close to 1. In this case the system learns slower and during the whole time of the hand presence it evaluates the hand as a disturbance and its disappearance normalizes the identification process and causes decrease of prediction error (see Fig. 2.1.3.).

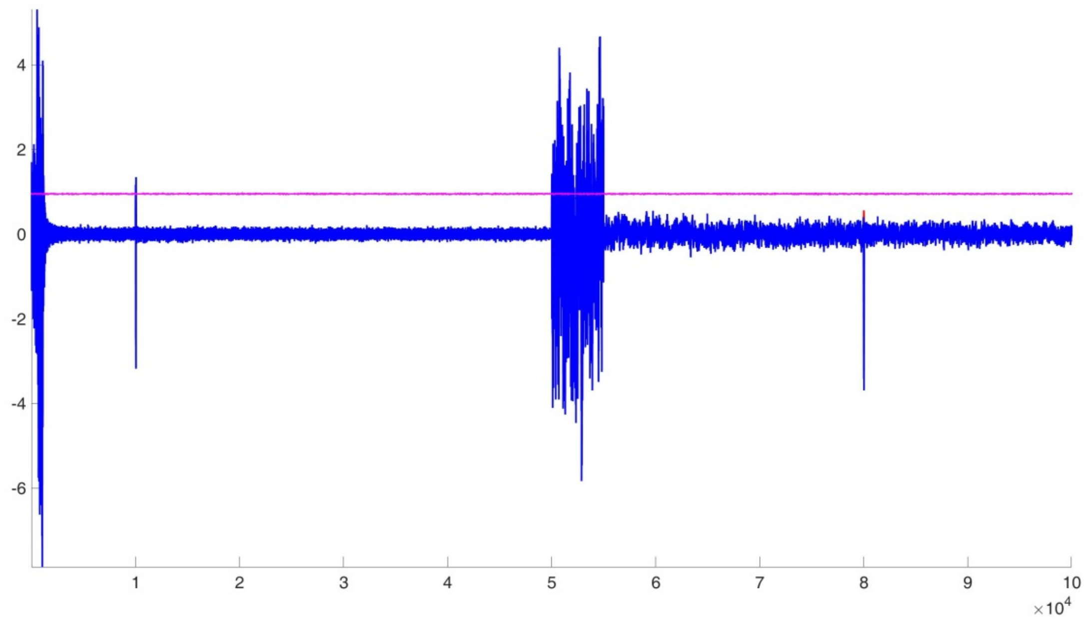

Figure 2.1.3. Identification results: φ=0.9999999999

Figure 2.1.4 shows the example of parameter estimation.

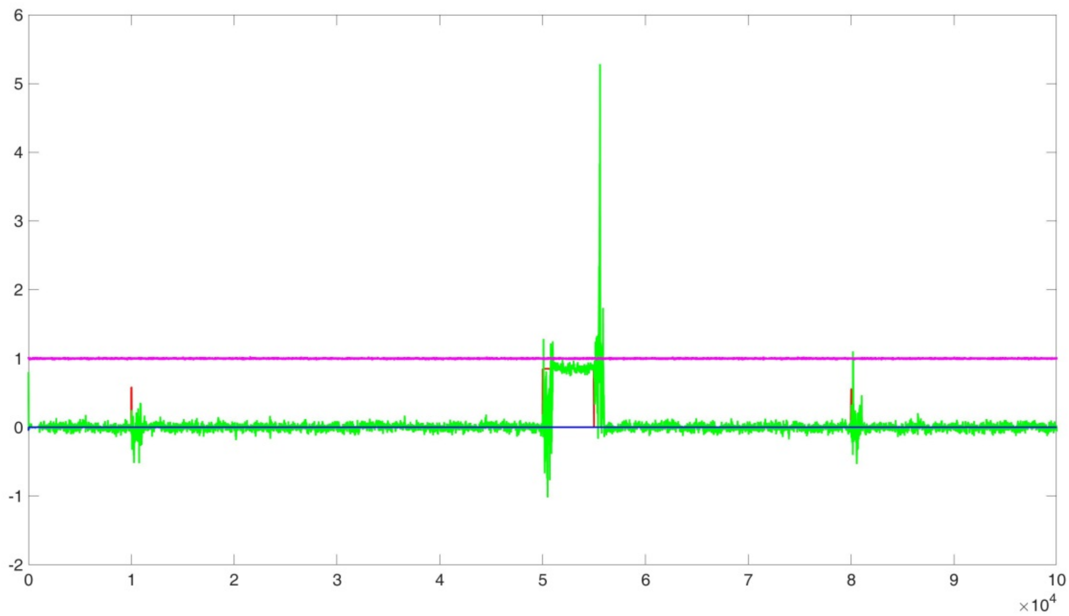

Figure 2.1.4. Parameter estimation: φ=0.999

For the purpose of clarity only one parameter of the environment model and the hand model is drawn. The parameter of the environment model is not seen on the graph, because it is set to zero, but the parameter of the hand model is colored in red and clearly shows time of hand appearance. The magenta line represents an input signal. A green curve stands for parameter estimation.

It is significant to point out that the main interest lays in prediction error development. It should be very similar to output of the hand model $\mathbf{y_2}$, because it is the hand appearance, which causes this error. For the case with the exponential forgetting equal to 0.999 the shape of two curves are very different (see Fig. 2.1.5 and Fig. 2.1.6).

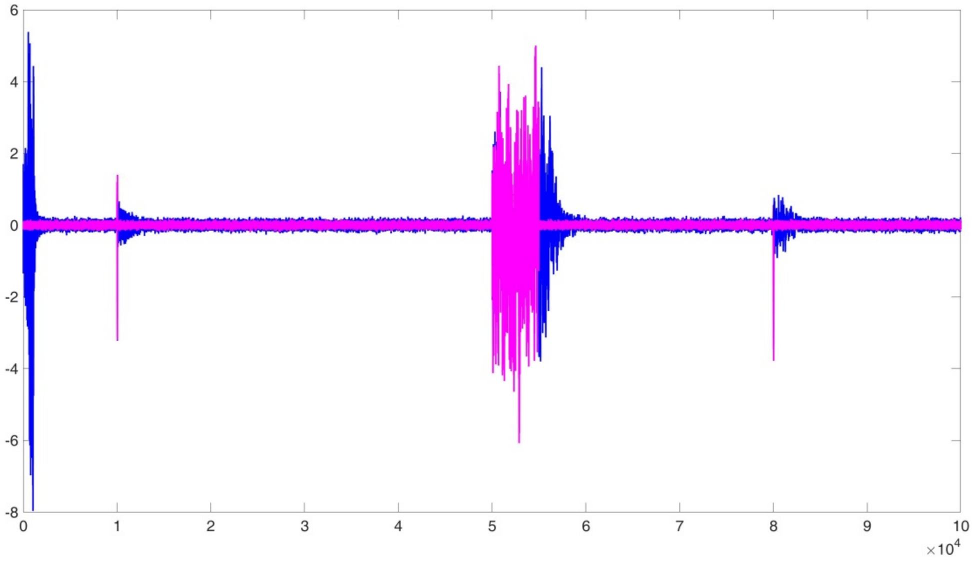

Figure 2.1.5. Comparison of y_2 and ê

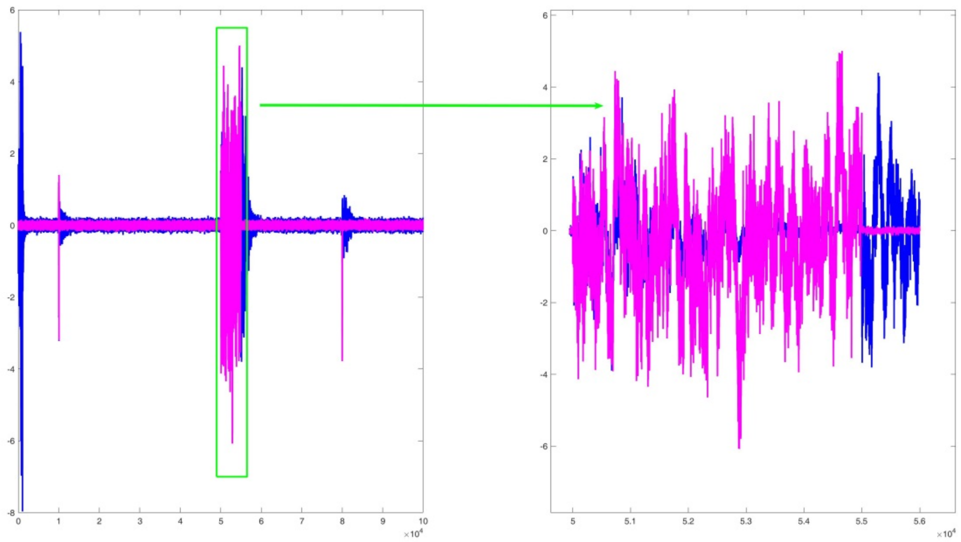

Figure 2.1.6. Comparison of y_2 and ê: one segment in details

Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

Output $\mathbf{y_2}$ is shown in a magenta colour, while a blue curve stands for prediction error development. It is obvious that in the middle of the graph, where the hand stays a bit longer, i.e. 5 000 samples, there is mismatching in the shapes. It can be explained by the fact that the system learns very quickly and after a short time period it does not take the presence of the hand as a disturbance anymore. Contrary it takes its disappearance as a new situation and, therefore, the prediction error increases again.

The situation is much better when the exponential forgetting is close to one. It is clearly seen from Figure 2.1.7 and Figure 2.1.8 that the shapes of output $\mathbf{y_2}$ and prediction error $\hat{\mathbf{e}}$ are very similar.

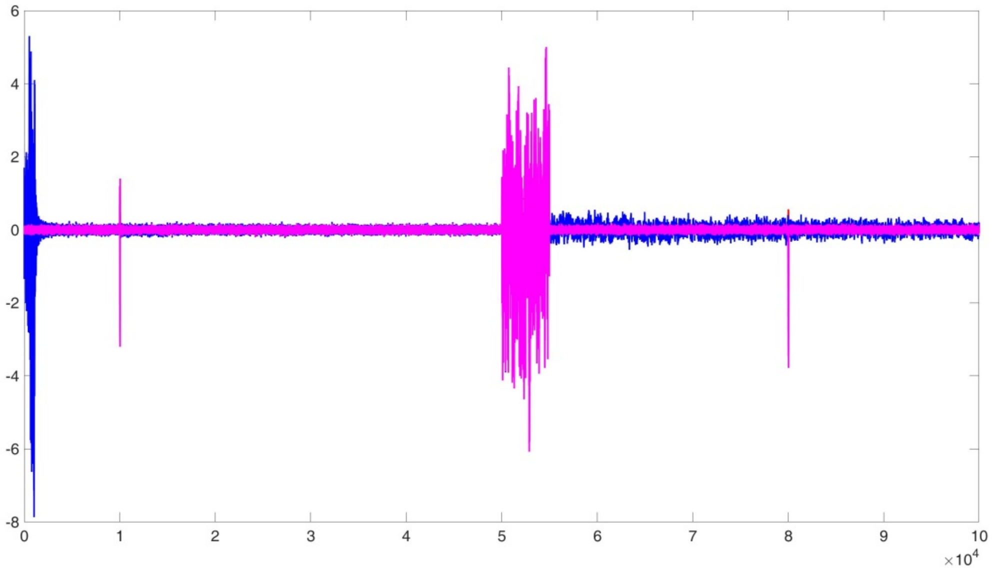

Figure 2.1.7. Comparison of y_2 and ê

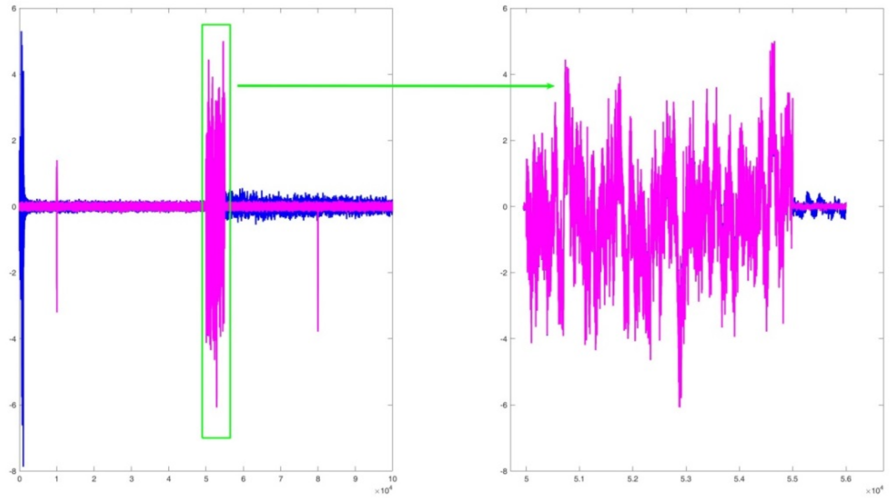

Figure 2.1.8. Comparison of y_2 and ê: one segment in details

It is worth saying that prediction error is of an essential interest for us, because it will be an input data for beamforming and further information processing. Therefore, it is of a great importance to receive prediction error corresponding to the reality.

2.2 Simulation results: experiment 2

To make the situation more difficult for identification process, on the one hand, and to make possible to define the distance between the signals, on the other hand, during the second experiment an input signal created from pulses is used (see Fig. 2.2.1). To run the experiment use a Matlab code **experiment_2.m** or a precompiled script **experiment_2.exe**.

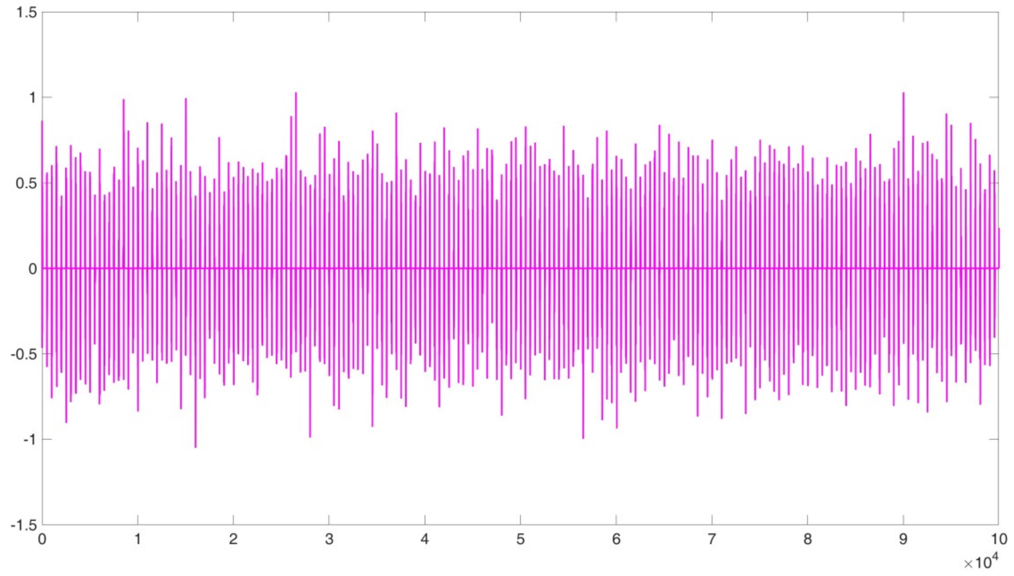

Figure 2.2.1. Input signal – pulses

The pulses are created from the previous input signal of white noise with period of 500 samples. The width of a pulse is 50 samples. A time scale of 100 000 steps is used.

The environment model is again a time-invariant regression model of the 1000th order with noise 0.05. The hand model is the 100th order regression model. The hand appears three times as in previous case: at 10 000, 50 000 and 80 000 time step. The first and the third appearance lasts for 50 samples, while the second one is longer and lasts for 5 000 samples. The exponential forgetting is used.

The example of identification process with the exponential forgetting equal to 0.999 is shown in Figure 2.2.2. Note that it is possible to change the exponential forgetting factor by changing **fim** value in a MATLAB script.

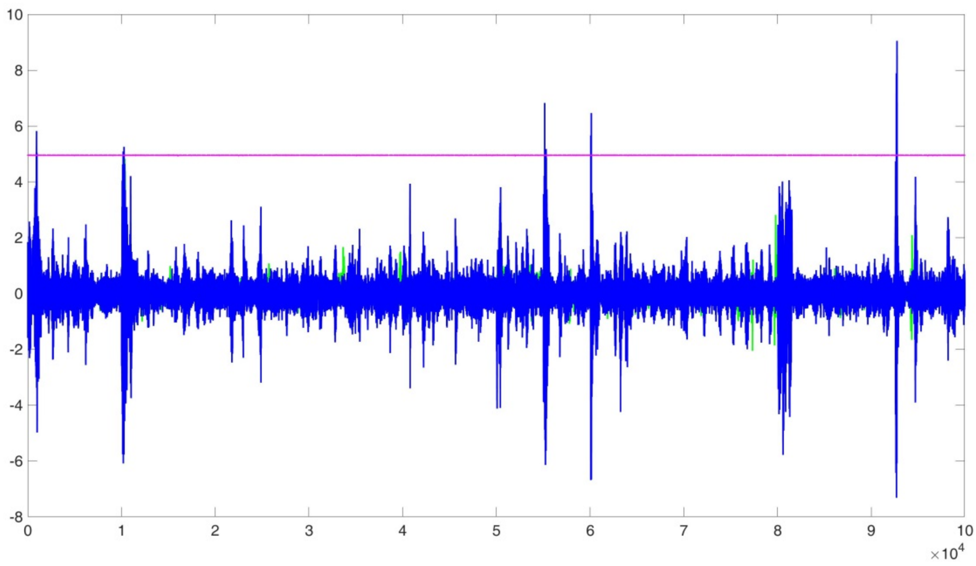

Figure 2.2.2. Identification results: φ=0.999

The input signal is coloured in magenta. The prediction error is put in blue. It is seen from the graph that it is almost impossible to define where the hand was placed.

If the development of prediction error **ê** and output $\mathbf{y_2}$ is compared, it is obvious that the difference between the shapes is drastic (see Fig. 2.2.3. and Fig. 2.2.4).

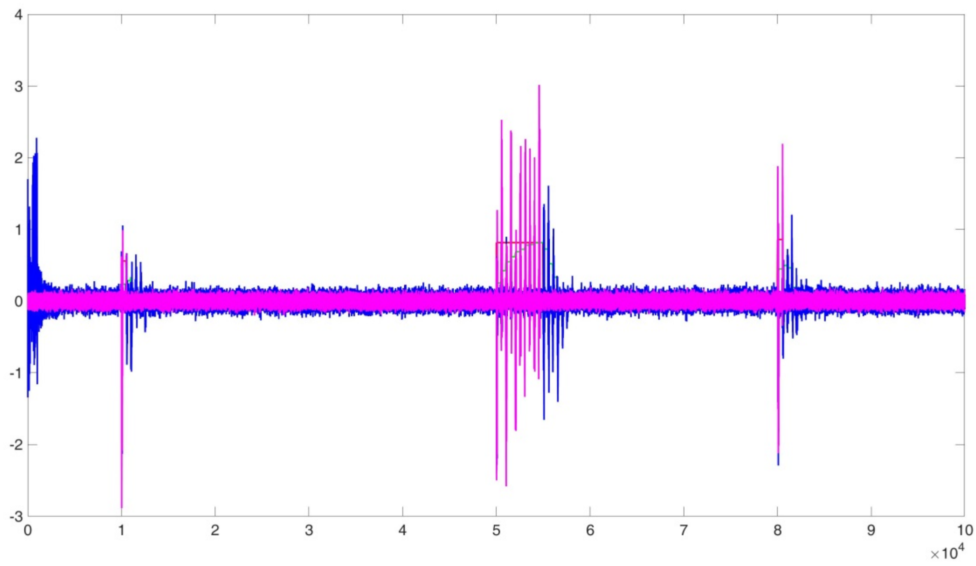

Figure 2.2.3. Comparison of $\mathbf{y_2}$ and ê

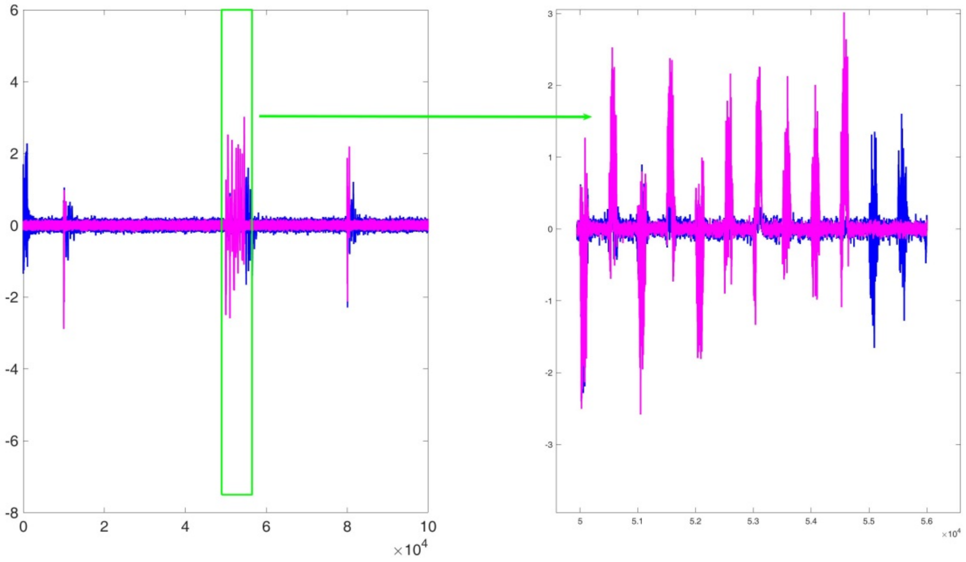

Figure 2.2.4. Comparison of y_2 and ê: one segment in details

The situation changes when the exponential forgetting is close to 1 (see Fig. 2.2.5).

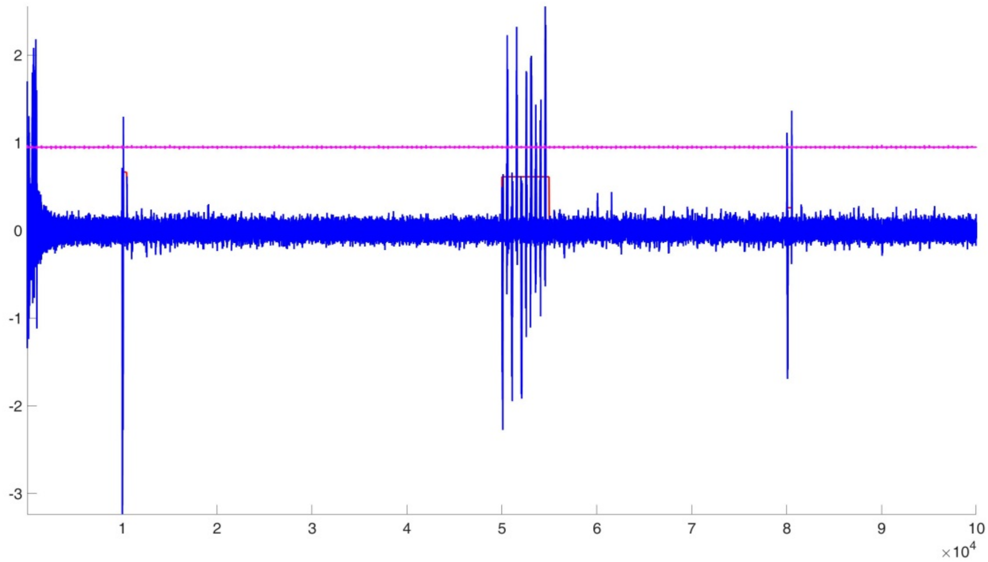

Figure 2.2.5. Identification results: φ=0.9999999999

It is obvious that in this case the hand appearance can be easily detected. Besides, because the input signal is composed from pulses with certain period, the distance can be easily detected too.

If the shapes of the curves of prediction error **ê** and output signal $\mathbf{y_2}$ are compared, it is also clear that they coincide (see Fig. 2.2.6 and Fig. 2.2.7).

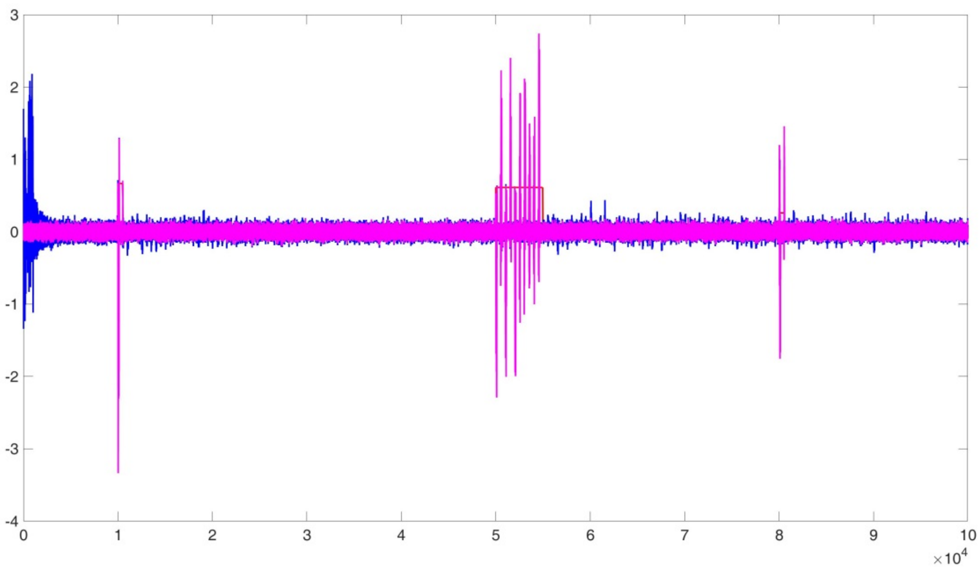

Figure 2.2.6. Comparison of y_2 and ê

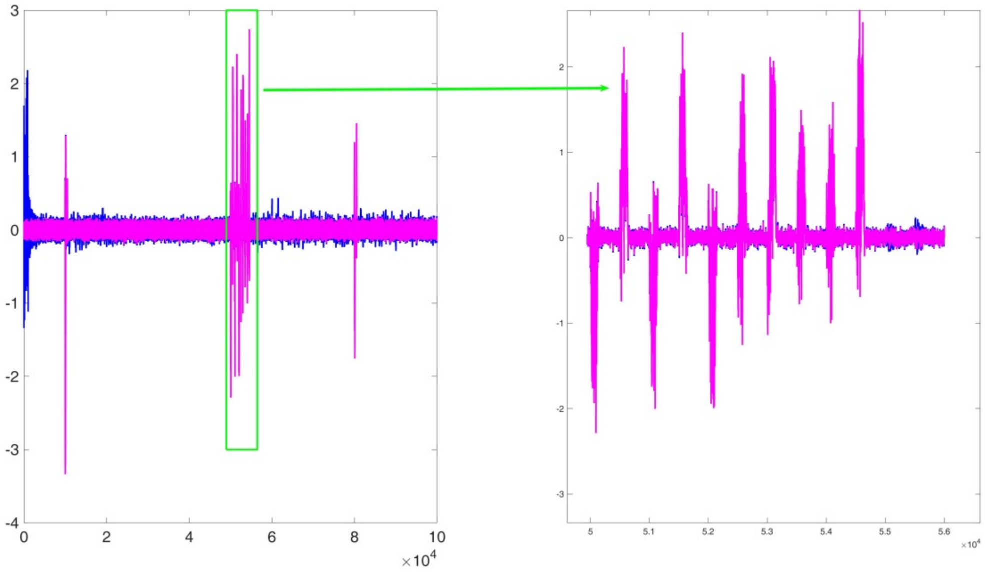

Figure 2.2.7. Comparison of y_2 and ê: one segment in details

With the exponential forgetting close to 1 the system learns slower and the identification process is more precise and allows recognizing hand appearance more accurate.

2.3 Simulation results: experiment 3

In the experiments above the environment model was a time-invariant regression model, i.e. a model with constant parameters. In reality some subjects, which can send undesired response to the detecting device, can move time from time. This slow movement can be described by the environment model with parameters changing in time. These time-variant parameters add more complexity for correct identification (see a Matlab file **experiment_3.m** or a precompiled script **experiment_3.exe**). The identification results with exponential forgetting φ =0.999 are shown in Figure 2.3.1. Note that it is possible to change the exponential forgetting factor by changing **fim** value in a MATLAB script.

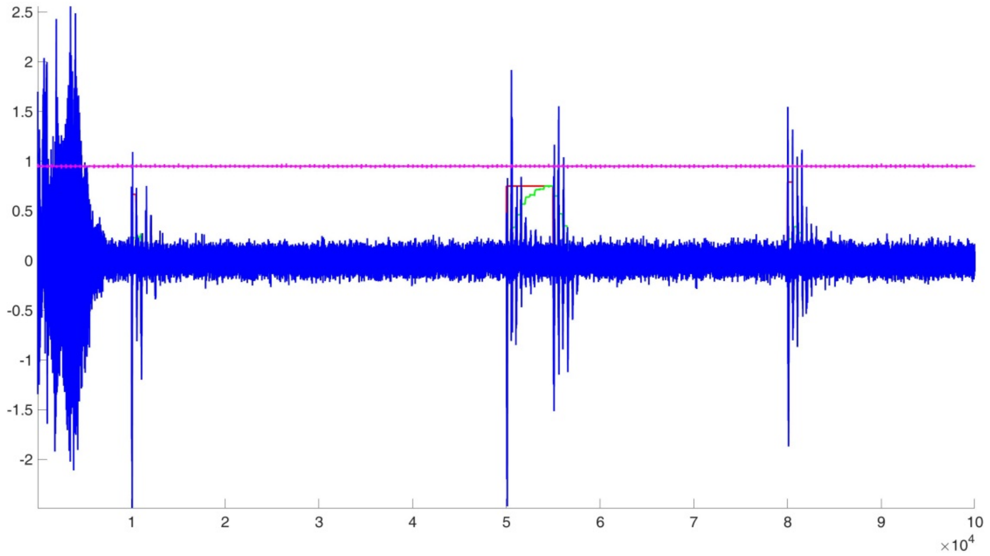

Figure 2.3.1. Identification results: φ=0.999

The environment model is represented by a time-variant regression model of the 1000th order with noise 0.05. The hand model is again a time-variant 100th order regression model with noise 0.05. The hand appears three times again: at 10 000, 50 000 and 80 000 time step lasting for 50, 5000 and 50 samples respectively.

The development of prediction error curve and the output of the hand model $\mathbf{y_2}$ are very different (see Fig. 2.3.2. and Fig. 2.3.3.).

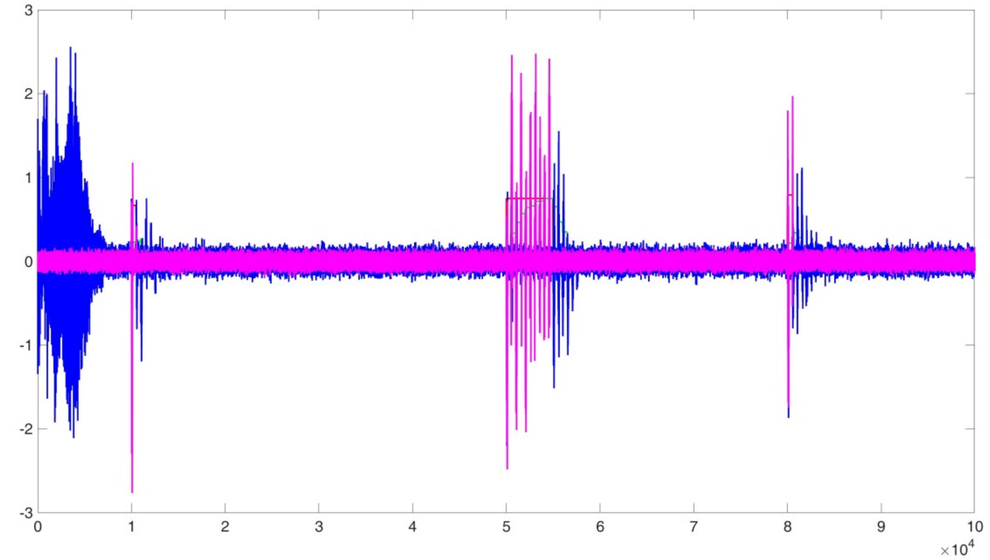

Figure 2.3.2. Comparison of $\mathbf{y_2}$ and ê

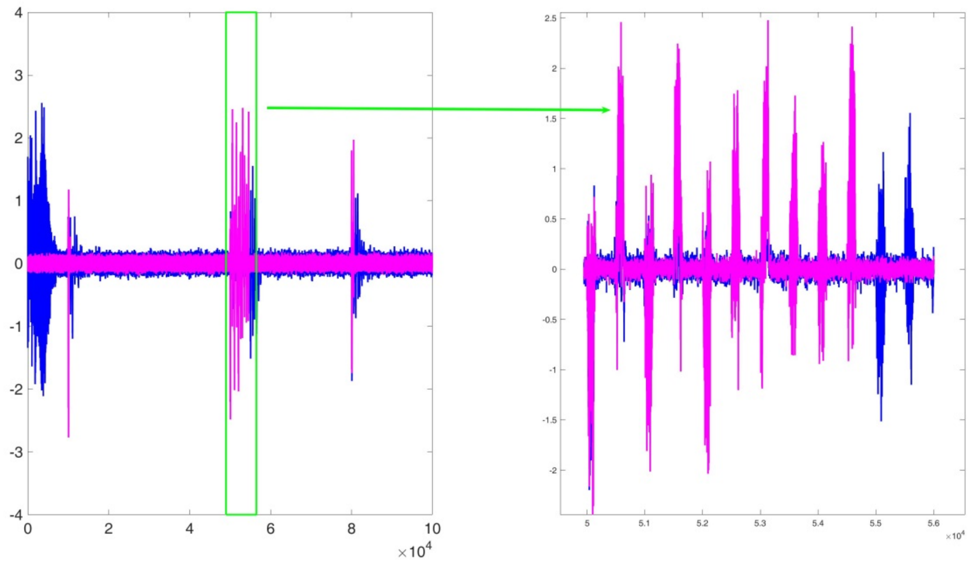

Figure 2.3.3. Comparison of y_2 and ê: one segment in details

The situation is better when the exponential forgetting is close to 1 (see Fig. 2.3.4).

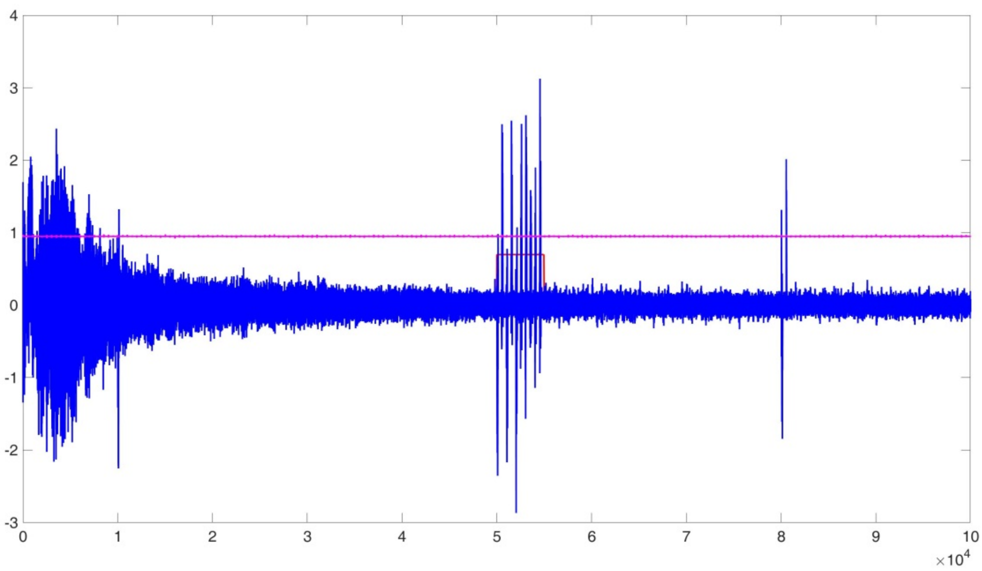

Figure 2.3.4. Identification results: φ=0.9999999999

Though the system requires more time to learn and as the result the first appearance of the hand is not clear, but then the identification process converges to the right values and the second and the third appearance of the hand are well seen from the graph.

Figure 2.3.5 and Figure 2.3.6 compares the shape of prediction error and the output $\mathbf{y_2}$. Again it is clear that the system needs more time to learn and to identify parameters correctly than in case with time-invariant environment model. But once it learnt the identification results are precise.

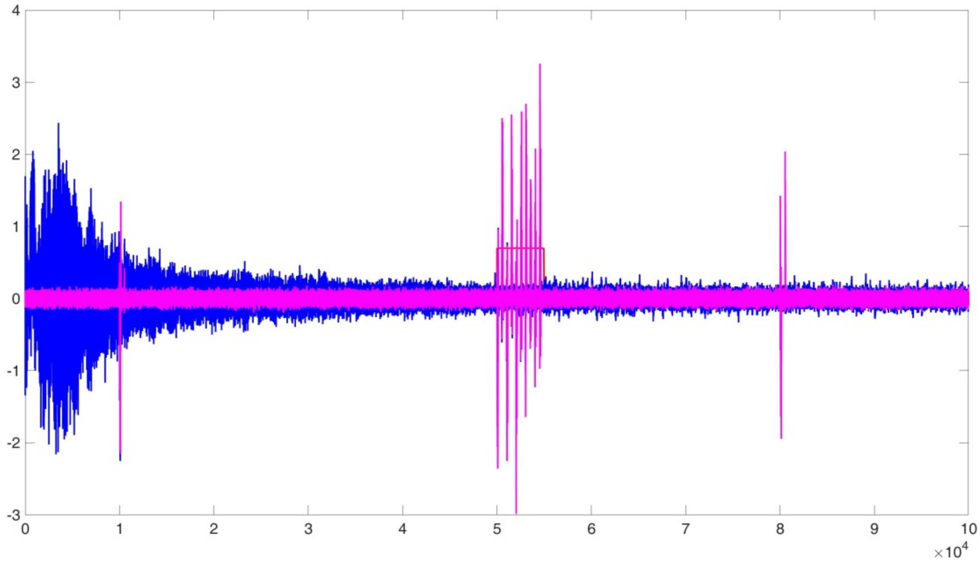

Figure 2.3.5. Comparison of y_2 and ê

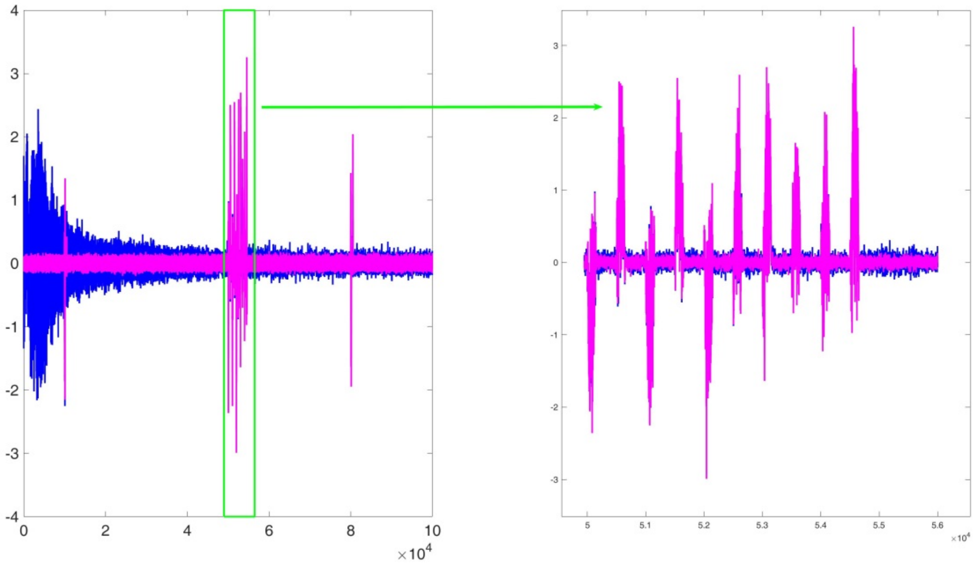

Figure 2.3.6. Comparison of y_2 and ê: one segment in details

2.4 Simulation results: experiment 4

Consider the situation when the hand remains for a longer time period in front of the transducers. It can influence the identification process in a way of complicating parameter estimation (for more information run a Matlab code **experiment_4.m** or a precompiled script **experiment_4.exe**).

Figure 2.4.1. shows the identification results for this situation when the exponential forgetting is equal to 0.999. Note that it is possible to change the exponential forgetting factor by changing **fim** value in a MATLAB script.

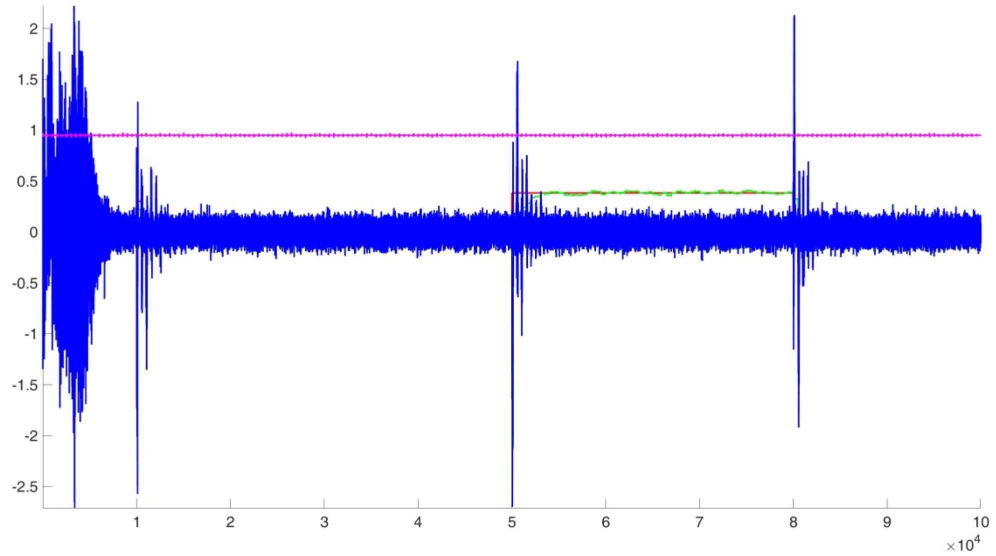

Figure 2.4.1. Identification results: φ=0.999

The input signal remains the same: it presents pulses with period of 500 samples and the width of 50 samples. It is drawn in magenta.

The environment model is again a time-variant regression model of the 1000th order. The parameters are random values and noise is equal to 0.05.

The hand model is a time-variant regression model of the 100th order. The parameters are also random values and noise is equal to 0.05.

This time the hand appears two times. The first time is at time step 10 000 and it lasts only for 50 samples. The second time is from 50 000 and it lasts for 30 000 samples.

As it is seen from Figure 2.4.1, predication error is large (coloured in blue) and it is hard to define the appearance of the hand from the graph.

Looking at Figure 2.4.2 and Figure 2.4.3 it is obvious that the development of prediction error and output signal $\mathbf{y_2}$ is different and, therefore, prediction error $\mathbf{\hat{e}}$ cannot present good input data for further gesture recognition processing.

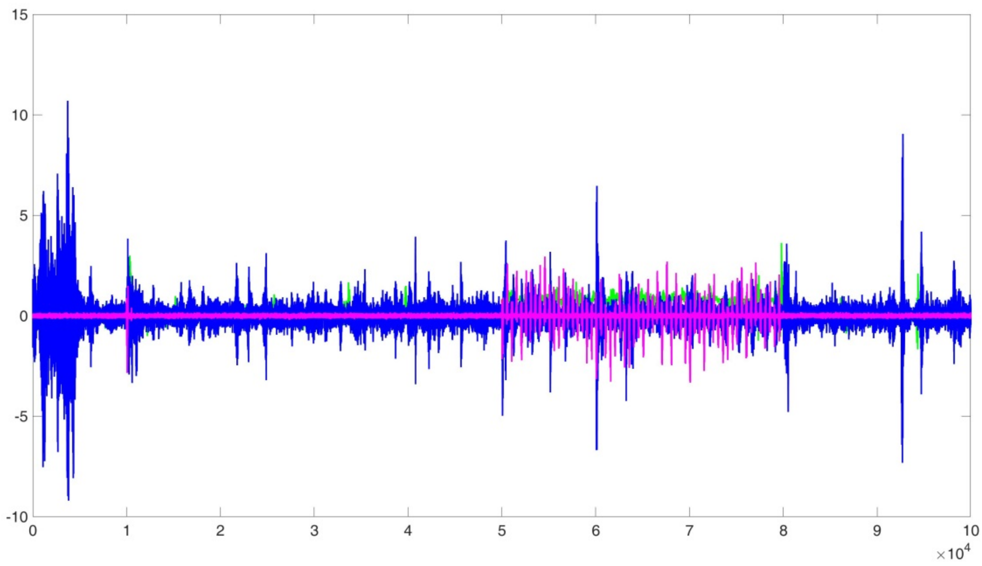

Figure 2.4.2. Comparison of y_2 and ê

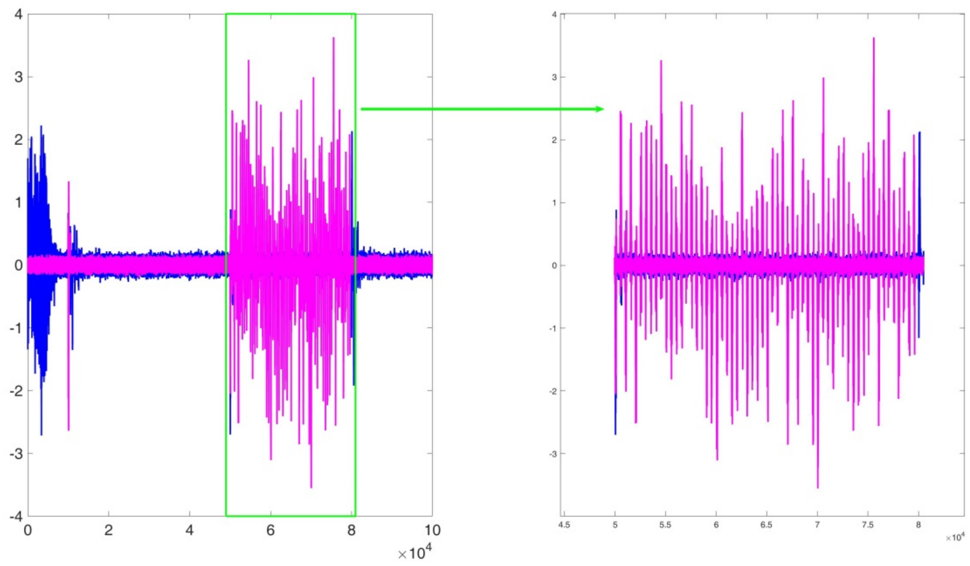

Figure 2.4.3. Comparison of y_2 and ê: one segment in details

Setting the exponential forgetting close to 1 will change the situation; however, the results of estimation are still not very satisfactory (see Fig. 2.4.4.).

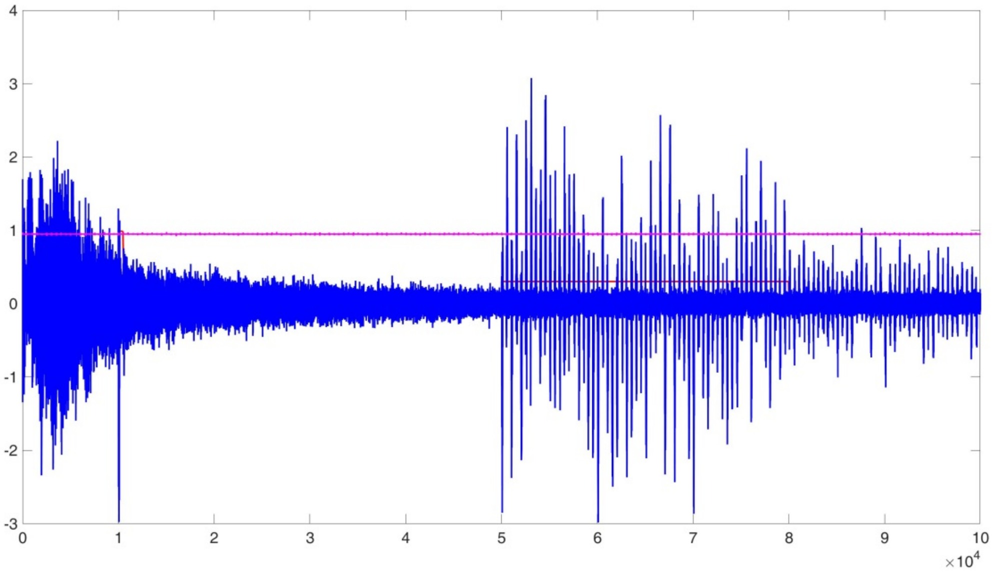

Figure 2.4.4. Identification results: φ=0.9999999999

As it is see from the figure, the first appearance of the hand is not detectable at all, because the system needs more time to learn, therefore, the prediction error (in blue) is so large. After learning the process of identification goes on better and easily detects the second appearance of the hand. From the graph it is seen, that the hand remains for some time (a red coloured line representing one of the parameters of the hand model). During this time the system learns and starts taking into consideration the parameters of the hand model, so after some time it does not consider the hand as a disturbance. Therefore, after the hand disappears there is still increase in the prediction error development.

The comparison of the shapes of prediction error **ê** and output signal $\mathbf{y_2}$ is shown in Figure 2.4.5. and Figure 2.4.6.

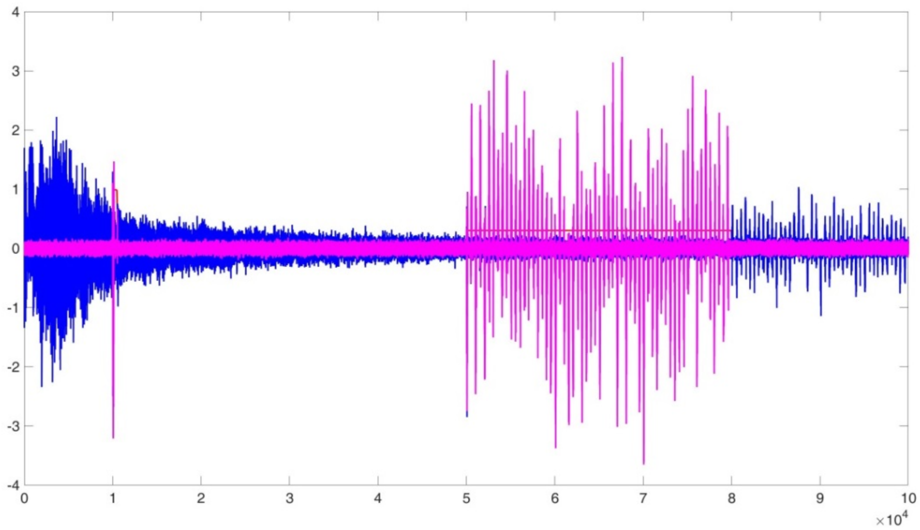

Figure 2.4.5. Comparison of $\mathbf{y_2}$ and ê

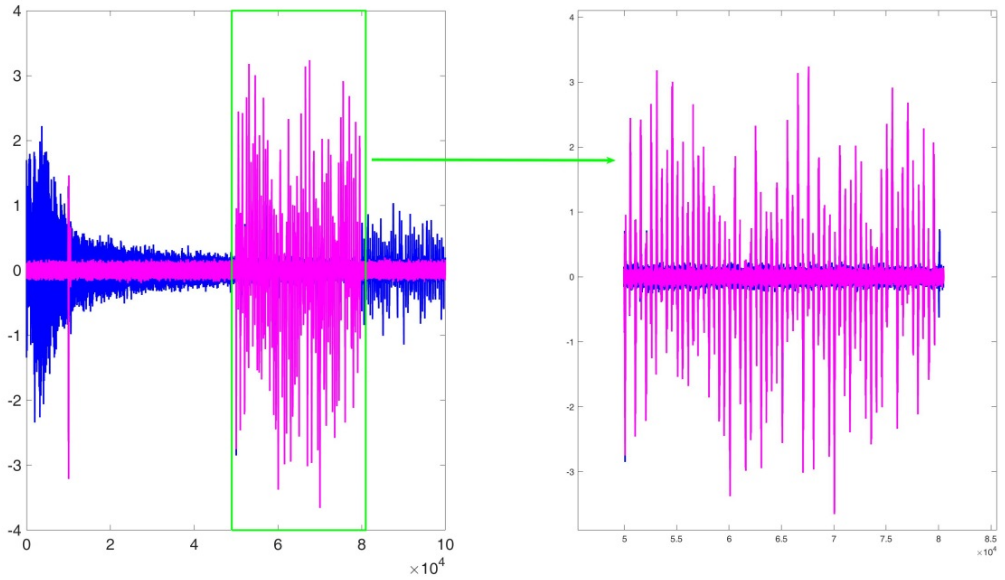

Figure 2.4.6. Comparison of y_2 and ê: one segment in details

3. Conclusion

Simulations represent a hand detection problem based on recursive QRD RLS algorithm serving for identification of the parameters of the FIR filter and for recursive calculation of the prediction error. During simulation several experiments have been performed, including experiments with time-variant and time-invariant FIR filter based environment model, different types of input signals and two values of the constant exponential forgetting factor.

The experiments show how important it is to set an exponential forgetting factor to the right value, so that the identification process corresponds to the specific use – the detection of a reflected signal from a hand present for a short period of time.

The experiments also show advantages and disadvantages of the approach to a noise cancellation problem, which was described in details in previous chapters.

The algorithm serves as a pre-processing stage for measurement of a hand distance and possibly also beam former based detection of a hand position. The final objective is the detection of hand gestures expressed by a short presence of a hand at certain distance from the ultrasound source.

Presented simulation results serve for us as a reference model before the implementation of the recursive QRD or QRD Lattice identification of the embedded Xilinx Zynq device, operating in real time with a microphone and an ultrasound transducer.

For more details and data availability the following MATLAB files can be used: **experiment_1.m, experiment_2.m, experiment_3.m, experiment_4.m**. If there is no MATLAB installed on a computer, the following precompiled scripts can be used: **experiment_1.exe, experiment_2.exe, experiment_3.exe, experiment_4.exe.**

4. Applications for Evaluation

4.1 Compiled scripts

The algorithms described in previous subchapters are created in MATLAB 2018b and are available in two forms:

- as **.m** files with .**mexw64** (MATLAB 2018b or higher has to be installed to use the files),
- as application with installation package for Win7 64b or Win10 64bit. (MATLAB is not necessary to be installed on the computer in this case).

4.2 Availability and Licensing

The evaluation package includes .m scripts with DSP algorithms pre-compiled as .mexw64 files for MATLAB R2018.b (or higher) and standalone applications for Win7 64b or Win10 64b (for users without MATLAB).

- These included DSP algorithms pre-compiled as .mexw64 files have no time restriction.
- The evaluation package can be downloaded and used free of charge.
- Source code of these DSP algorithms is not provided in this evaluation package.

4.3 References

Disclaimer

This disclaimer is not a license and does not grant any rights to the materials distributed herewith. Except as otherwise provided in a valid license issued to you by UTIA AV CR v.v.i., and to the maximum extent permitted by applicable law:

(1) THIS APPLICATION NOTE AND RELATED MATERIALS LISTED IN THIS PACKAGE CONTENT ARE MADE AVAILABLE "AS IS" AND WITH ALL FAULTS, AND UTIA AV CR V.V.I. HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and
(2) UTIA AV CR v.v.i. shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under or in connection with these materials, including for any direct, or any indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or UTIA AV CR v.v.i. had been advised of the possibility of the same.

Critical Applications:
UTIA AV CR v.v.i. products are not designed or intended to be fail-safe, or for use in any application requiring fail-safe performance, such as life-support or safety devices or systems, Class III medical devices, nuclear facilities, applications related to the deployment of airbags, or any other applications that could lead to death, personal injury, or severe property or environmental damage (individually and collectively, "Critical Applications"). Customer assumes the sole risk and liability of any use of UTIA AV CR v.v.i. products in Critical Applications, subject only to applicable laws and regulations governing limitations on product liability.