Tuning of a Hierarchical Fuzzy System for Video De-interlacing

Piedad Brox, Iluminada Baturone, and Santiago Sánchez-Solano

***Abstract*— The tuning of hierarchical fuzzy systems are not supported by the majority of CAD tools available at the market currently. The *xfsl* tool integrated into Xfuzzy 3 allows the tuning of complex fuzzy systems, for instance, hierarchical systems with modules in cascade. The authors propose the use of this tool for tuning a complex fuzzy system for video de-interlacing in this paper. The parameters obtained after tuning are proven by de-interlacing a wide battery of sequences. The use of tuning techniques improves the quality of de-interlacing and provides an algorithm simplification that facilitates its hardware implementation.**

I. Introduction

Independently of the description of a Simple Fuzzy Systems (SFS) comes from a set of data or heuristic knowledge, its behavior depends on both its structure (the number of rules, the number of fuzzy sets covering the input and output universes of discourse, etc) and its parameters (those which define the membership functions associated with input and output variables). A SFS is understood as a system with a unique rule base and particular membership functions, connective operators and defuzzification methods. The tuning of the membership functions representing the antecedents and the consequents of the rules usually improves the performance of the SFS. Several CAD tools developed for automatic tuning are currently available at the market [1]-[4].

Hierarchical Fuzzy Systems (HFS), that is, systems that contain several modules connected in cascade, in parallel, or in a hybrid architecture allow the description of more complex situations. For instance, a knowledge-based HFS is described in [5] for a pilot pressure control system. Furthermore, HFS have a nice property since the total number of rules increases only linearly with the number of input variables instead of an exponential increase of SFS [6]. This issue is particularly relevant to carry out the implementation of the whole system. All the above considerations have encouraged the development of algorithms for the automatic design of the Takagi-Sugeno HFS during the last years [7]-[8].

Only a few number of CAD tools support the tuning of a hierarchically structured knowledge. As far as we know, *xfsl* integrated into the environment of Xfuzzy 3 is the most versatile tool since it includes a large number of learning algorithms [9]. Besides, *xfsl* does not impose any severe constraints on the description system: a large kind of membership functions can be used, linguistic hedges, connective operators, and defuzzification methods, and a hierarchical structure of any of the system type is supported.

This paper describes the use of *xfsl* to tune a knowledge-based HFS for video de-interlacing application. De-interlacing is one of the main tasks in video processing. It is necessary whenever the transmission standard uses an interlaced format but the receiver requires a progressive scanning, as happens to many consumer electronics equipments (LCDs and plasma displays, projectors, and DVDs) [10]. The goal of de-interlacing algorithms is the interpolation of missing lines during the TV transmission.

This paper is organized as follows: Section II briefly describes the algorithm for video de-interlacing. Its description into Xfuzzy 3 environment is detailed in Section III. including the tuning stage by using *xfsl*. The results of the tuned system are analyzed in Section IV. Finally, some conclusions are expounded in Section V.

II. Description of the algorithm

Among de-interlacing algorithms, motion-adaptive approaches are reported in the literature as a good midpoint between simple linear algorithms and complex motion-compensated-ones[10]. They are based on the fact that linear temporal interpolator are perfect in the absence of motion, whereas linear spatial methods offer a most adequate solution in case that motion is detected. Motion detection can be implicit, as in median-based techniques [11]-[12], or explicit, using a motion detector [13]-[16].

The explicit motion-adaptive algorithms de-interlace video by applying [1]:

$$I_p(x,y,t) = (1-\alpha)I_T(x,y,t) + \alpha I_S(x,y,t) \qquad (1)$$

where I_S is the output of a spatial interpolator, and I_T is the output of a temporal interpolator. The variable α, which is the output of a motion detector, ranges from 0 to 1 and determines the level of motion.

The performance of explicit motion-adaptive algorithms relies on the quality of the motion detector, since it is strongly dependent on the combination of both de-interlacing algorithms. One way to improve the performance of this kind of algorithms is to use a heuristic knowledge-based system.

Piedad Brox is with the Microelectronics Institute of Seville (CNM-CSIC), and the Department of Electronics and Electromagnetism, University of Seville, Américo Vespucio s/n, 41092 Seville, Spain (phone: +34 954 466666; email: brox@imse-cnm.csic.es).

Iluminada Baturone is with the Microelectronics Institute of Seville (CNM-CSIC), and the Department of Electronics and Electromagnetism, University of Seville, Spain (email: lumi@imse-cnm.csic.es).

Santiago Sánchez-Solano is with the Microelectronics Institute of Seville (CNM-CSIC), Spain (email: santiago@imse-cnm.csic.es).

[1] Motion-adaptive de-interlacing process may be performed separately on each color space component for color transmission standards. However, based on the fact that the details of an image are mainly determined by the luminance component of the video signal, and much less by the chrominance components, the motion detection is usually done with the luminance component

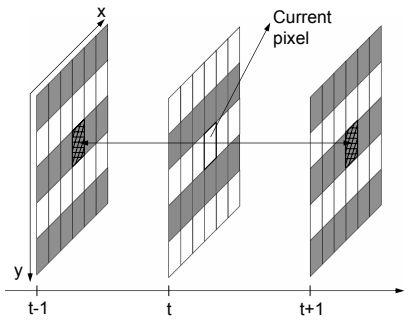

Fig. 1. $H(x, y, t)$ measures the difference between the luminance values of two pixels with the same spatial co-ordinates but belonging to different fields.

In this sense, heuristics states that a spatial interpolator is the most adequate when motion is large and a temporal interpolation when motion is scarce. This algorithm was presented in [17] and the aim of the present work is the improvement of the tuning that was applied to the fuzzy system.

The detection of motion is based on the assumption that motion produces variations in the luminance of pixels with the same spatial co-ordinates but in different fields. Therefore, the simplest motion detector only evaluates the difference between two fields of the same polarity as follows:

$$H(x, y, t) = \frac{|I(x, y, t + 1) - I(x, y, t - 1)|}{2} \tag{2}$$

The function $H(x, y, t)$ involves the pixels shown in Fig. 1.

If the field differences of several pixels around the current pixel are considered, the motion measurement is more reliable. Our proposal in [17] uses a bi-dimensional convolution since the use of sum-product allows to distinguish the different levels of motion. Furthermore, convolution allows to apply different weights to each neighbor when estimating motion. As detailed in [17], the level of $motion$ is estimated as follows:

$$motion(x, y, t) = \frac{\Sigma_{a=1}^{3}(\Sigma_{b=1}^{3} H_{a,b} C_{a,b})}{\Sigma_{a=1}^{3}\Sigma_{b=1}^{3} C_{a,b}} \tag{3}$$

where $H_{a,b}$ are the elements of the following frame difference matrix:

$$H = \begin{pmatrix} H_{(-2,-1,-1)} & H_{(-1,-1,-1)} & H_{(0,-1,-1)} \\ H_{(-2,0,0)} & H_{(-1,0,0)} & H_{(0,0,0)} \\ H_{(-2,1,-1)} & H_{(-1,1,-1)} & H_{(0,1,-1)} \end{pmatrix} \tag{4}$$

with $H_{(i,j,k)}$ corresponding to the frame difference $H(x + i, y + j, t + k)$ (see equation (2)).

And $C_{a,b}$ are the elements of the following weight matrix[2]:

$$C = \begin{pmatrix} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{pmatrix} \tag{5}$$

[2]The selection of these elements seem logical since the highest weight is assigned to the position placed on the current pixel while the values of the rest of weights are lower as the corresponding pixel is further from the current one

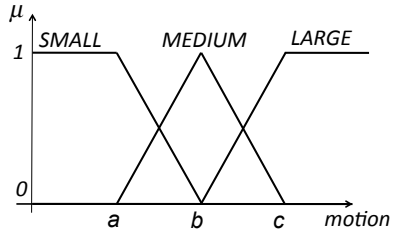

Fig. 2. Membership functions for the *SMALL*, *MEDIUM*, and *LARGE* fuzzy concepts.

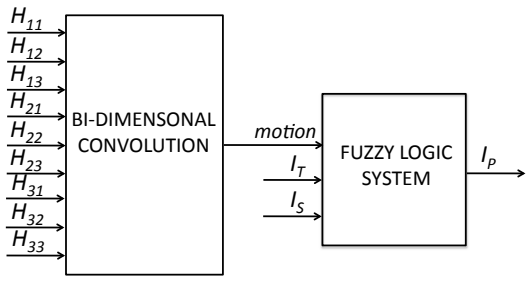

Fig. 3. Block diagram of the proposed algorithm.

The measure of motion (*motion*) is the input of a SFS. The rule base of this system only contemplates three rules (see Table I):

1) If *motion(x,y,t)* is *SMALL* then the interpolation is realized by applying a temporal de-interlacing algorithm (I_T).
2) If *motion(x,y,t)* is *MEDIUM* then a linear combination of the spatial (I_S) and temporal (I_T).
3) If $motion(x, y, t)$ is $LARGE$ then the interpolation is realized by applying a spatial de-interlacing algorithm (I_S).

Because of linguistic coherence, the membership functions *SMALL* and *MEDIUM* are complementary, and something similar happens to *MEDIUM* and *LARGE* as shown in Fig. 2. In principle, there is no limitation to choose the shape of these functions. Due to its simplicity the piece-wise linear functions in Fig. 2 have been finally selected to describe the fuzzy sets $SMALL$, $MEDIUM$, and $LARGE$.

The new luminance component of a de-interlaced pixel provided by the system is calculated by applying the FM defuzzification method as follows:

$$I_P(x, y, t) = \frac{\sum_{i=1}^{3} \beta_i c_i}{\sum_{i=1}^{3} \beta_i} \tag{6}$$

where β_i is the activation degree of the i-th rule and c_i is the consequent of the i-th rule in Table I.

A block diagram of the algorithm to adapt the strategy of de-interlacing to motion is shown in Fig. 3. In the work presented in [17], a tuning technique is applied to the SFS shown in Fig. 3. The aim of the work presented herein is to exploit the capabilities of *xfsl* to tune the whole system, that is, the module that performs the bi-dimensional convolution and the SFS.

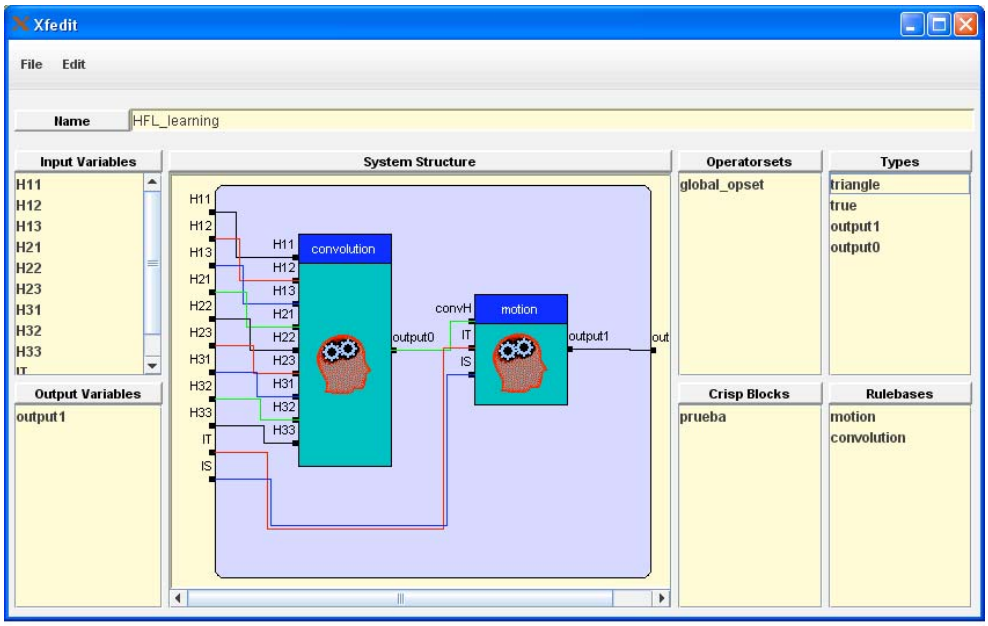

Fig. 4. Description of the HFS into Xfuzzy 3 environment.

TABLE I

FUZZY RULE SET FOR MOTION-ADAPTIVE DE-INTERLACING

Rule	Antecedents	Consequent
1.	*motion(x,y,t)* is *SMALL*	$I_T(x,y,t)$
2.	*motion(x,y,t)* is *MEDIUM*	$\gamma \cdot I_T(x,y,t) + \lambda \cdot I_S(x,y,t)$
3.	*motion(x,y,t)* is *LARGE*	$I_S(x,y,t)$

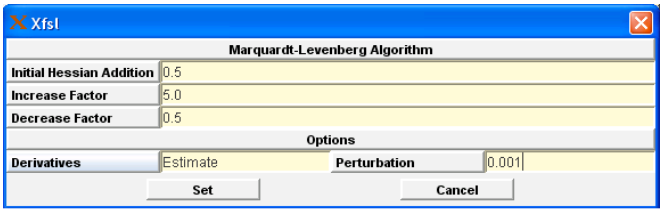

Fig. 6. Configuration of the learning algorithm.

III. TUNING OF THE HFS

The whole system in Fig. 3 can be understood as a HFS that is composed by two modules connected in cascade. The first one performs the convolution and its output is a non-fuzzy measurement (*motion* in (3)), and the second one is the SFS for motion adaptation. The goal of this work is the tuning of the whole HFS, the tuning of the convolution coefficients (see equation in (5)) and the tuning of the SFS parameters.

A. Description of the system into Xfuzzy 3

The inputs of the whole HFS are the nine frame differences, the spatial (I_S) and the temporal (I_T) interpolators. The HFS described in Xfuzzy 3 is shown in Fig. 4. The modules of the HFS are named 'convolution' and 'motion', and both are described in Xfuzzy 3 as first order Takagi-Sugeno systems.

The inputs of the convolution module are the nine frame differences. Since the aim of this first module is to have a linear combination of the inputs, the rule base only contains a unique rule that is always activated: *'if any of the inputs (H_{ij}) is DUMMY then output0 is motion'*.

The output variable called 'output0' of this system has a unique label called 'motion' that is a parametric membership function with ten parameters, nine for each one of the input variables and one parameter that corresponds to the independent term.

The second fuzzy system uses three input variables. One of them is the output of the first SFS, and the other two corresponds to the spatial and temporal interpolators. The edition of the rule base of this system is shown in Fig. 5. The membership functions of the fuzzy concepts *SMALL*, *MEDIUM*, and *LARGE* are defined as triangular functions (see Fig. 2). The output variable of this second module corresponds to the interpolated pixel (I_P in equation (1)).

B. Tuning into xfsl

Since input/output training data can be extracted from progressive standard video sequences, supervised learning algorithms have been selected to tune the whole system.

Once the training file is available, a learning algorithm has to be chosen. Among second-order conjugate gradient algorithms, the Marquardt-Levenberg algorithm was selected. The gradient of the error function is not supported for this particular complex system but *xfsl* offers the possibility of estimating the derivaties of the error function. The configuration of the learning algorithm is shown in Fig. 6.

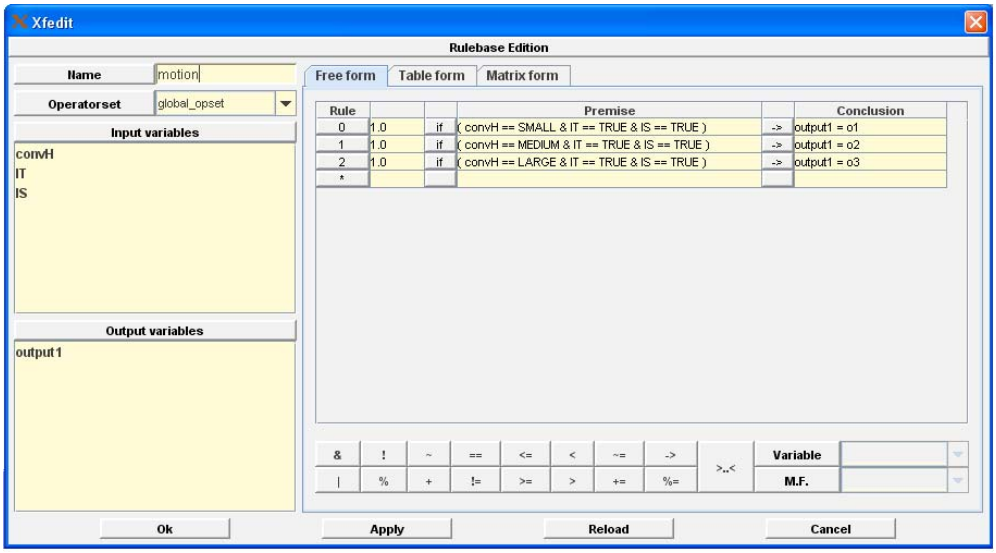

Fig. 5. Edition of the rule base used in the SFS for motion-adaptive de-interlacing.

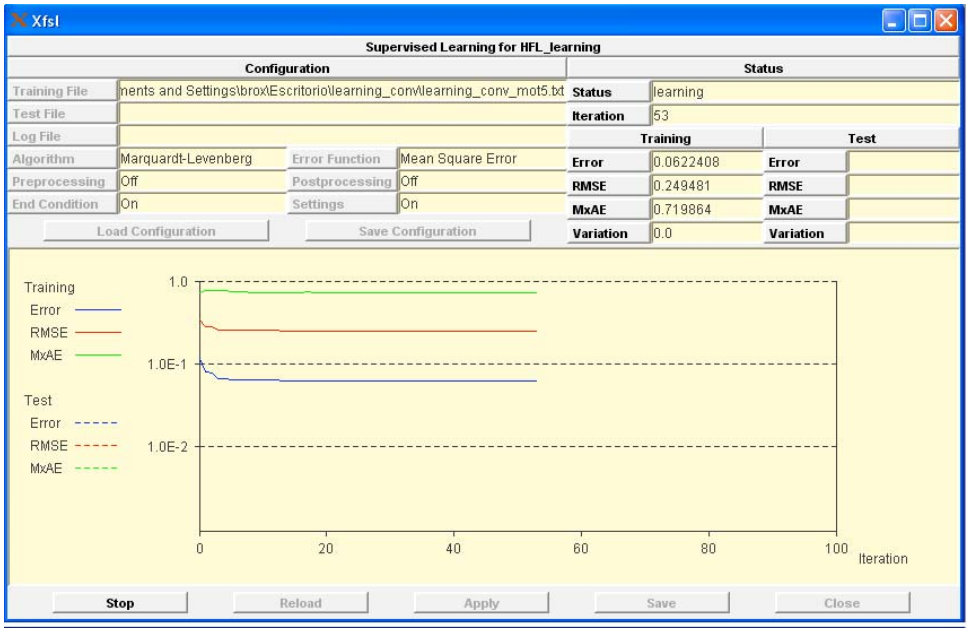

Fig. 7. Evolution of the tuning process.

The strategy used to tune the whole HFS is the following[3]:

- Since the second system, named as 'motion' in Fig. 4, has more influence in the result of de-interlacing, this system has been tuned firstly.
- After fixing the parameters of the second system, the first one is adjusted. Fig. 8 shows the inclusion of the tuning parameters for the first fuzzy system.

The evolution of the tuning process is shown in Fig. 7. The picture shows that the system learns after a few number of iterations. The results obtained are analyzed in the next section.

[3]This strategy was determined after analyzing several tests

IV. Simulation Results

These results were obtained using a training file with data that contemplates numerous situations. This means that the data cover uniformly the universe of discourse of the variables.

The more interesting results were obtained by following the procedure explained in Section III.B. Firstly, the parameters of the second module were tuned. The three membership functions after tuning are shown in Fig. 9. The most relevant result is the increase of the parameter b in Fig. 2 up to a numerical value higher than 50. The values of weights for the spatial and the temporal interpolator in the rules' consequents (see Table I) are also tuned. The parameter γ is modified

TABLE II

COMPARISON RESULTS AMONG METHODS BEFORE AND AFTER LEARNING

Method	Mother Video QCIF	Carphone Video QCIF	Missa Video CIF	Paris Video CIF	Salesman Video CIF	Trevor Video CIF	Fire Rose Film PAL TV	Fargo1 Film PAL TV
VT 2 fields [10]	39.61	34.08	40.25	30.73	36.54	36.61	40.32	35.87
Median motion [10]	38.49	33.31	39.44	30.27	36.61	35.43	39.45	35.32
MC field insertion [10]	39.72	34.63	40.89	34.48	38.32	37.11	42.61	42.77
Before learning	41.23	36.64	40.21	34.67	38.14	36.22	39.79	36.48
After learning +C_{tuned} in (7)	41.45	35.35	40.76	35.94	38.46	37.49	40.15	36.76
After learning +$C_{vertical}$ in (10)	41.35	35.05	40.51	35.77	38.45	37.29	39.91	36.74

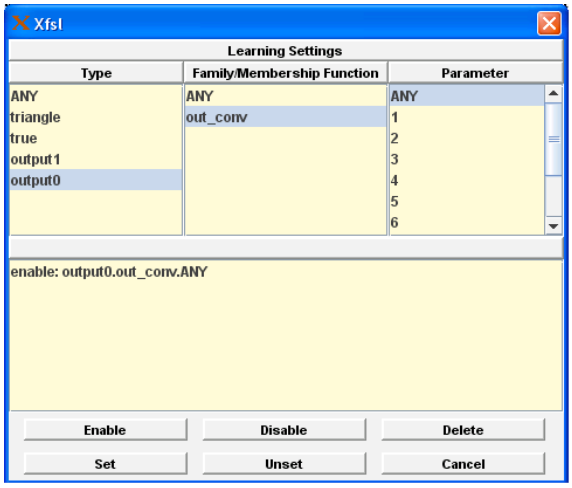

Fig. 8. The parameters of the first system are enable to participate in the tuning process.

from 0.5 up to 0.4, and the parameter λ from 0.5 up to 0.6. Although the sum of these parameters are not constrained to be 1, the learning process converges to values that always fulfill the condition γ+λ=1.

After the tuning stage, the matrix of coefficients is modified from equation (5) to the following one:

$$C_{tuned} = \begin{pmatrix} 0.049 & 0.2052 & 0.043 \\ 0 & 0.381 & 0 \\ 0.047 & 0.2365 & 0.038 \end{pmatrix} \quad (7)$$

The tuning process increases the weights of the pixels in the vertical direction at expense of reducing the rest of weights.

The validation of the tuning process has been proven by de-interlacing several sequences that have been widely used as benchmarks in video processing applications. The origin of the test material is categorized into two types: video and film. Many error measures have been proposed as figures of merit to evaluate the quality of digital video. However, as a consequence of the complexity of the human visual system, it is difficult to find an objective criterion to entirely quantify image distortions. Nevertheless, some measures seem to have a higher correlation than others with the perceived quality. A very popular quality criteria is the MSE (Mean Squared Error) between the original and the reconstructed images, which is given by the following expression:

$$MSE = \frac{1}{M \cdot N} \Sigma_{x,y} (I_P(x,y,t) - I_{original}(x,y,t))^2 \quad (8)$$

where one frame has a resolution of MxN pixels, I_P is the value of the interpolated pixel, and $I_{original}$ is the pixel value in the original progressive image. Strongly related to the MSE is the PSNR, which is defined as follows:

$$PSNR = 20 \log \frac{255}{\sqrt{MSE}} \quad (9)$$

Table II shows the average PSNR values obtained when de-interlacing 50 fields of different sequences with three different formats: QCIF, CIF, and PAL TV. Table II includes results after de-interlacing with the version of the algorithm before learning, the algorithm after learning with the convolution mask in equation (7), and finally, a slight modified version of the algorithm after learning with a convolution mask that only considers neighbors in vertical direction as follows:

$$C_{vertical} = \begin{pmatrix} 0.25 \\ 0.5 \\ 0.25 \end{pmatrix} \quad (10)$$

All the algorithms use two simple de-interlacing algorithms, *line average* as spatial interpolator and *field insertion* as temporal interpolator.

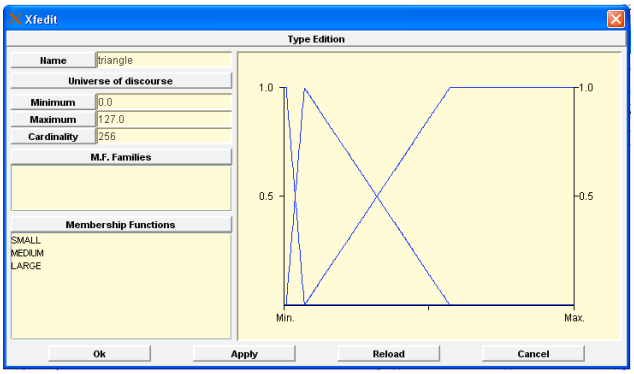

Fig. 9. Membership function after the learning stage.

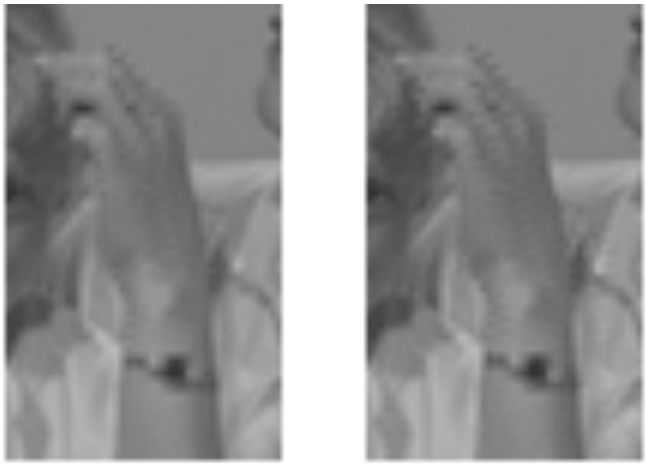

Fig. 10. De-interlaced image of the Mother sequence: (a) after learning: (b) before learning.

A unique PSNR value in Table II is not meaningful, but the comparison between two values from different methods gives a measurement of quality. Generally, an improvement of 0.5 dBs in PSNR is quite perceptible by the human visual system in the de-interlaced image. Taking into account all the above considerations, the improvements after learning are evident when comparing the first and second rows in Table II. The loss of quality in terms of PSNR when the convolution mask is modified from equation (7) to (10) is not significative as can be seen after comparing the second and the third rows.

Fig. 10 corroborates the superior quality of a de-interlaced frame after learning versus a picture without tuning. The differences between the convolution masks in equations (7) and (10) are not significant and they are not evident at naked eye.

The resulting algorithm after the tuning stage performs better than other algorithms of similar complexity such as Vertico-Temporal (VT) and median-based approaches [10] (see results in Table II). It also improves the results of many sequences when an algorithm of much greater complexity, such as a motion-compensated (MC) algorithm is used.

V. Conclusions

The motivation of our work is the improvement of a fuzzy logic-based system for motion-adaptive de-interlacing by using tuning techniques. The whole system can be understood as a HFS that has been tuned by using the tool called *xfsl* integrated into Xfuzzy 3 environment.

Simulation results demonstrate that the system after tuning improves the quality of de-interlaced images. The coefficients of the convolution mask after tuning show that the weights of the mask in the vertical direction have a significant influence on the de-interlaced results. The convolution mask after tuning is slightly modified to a new version that only considers neighbors in the vertical direction. This modification allows the simplification of the algorithm from a hardware point of view without a relevant loss of quality.

Acknowledgment

This work was partially supported by MOBY-DIC project FP7-INFSO-ICT-248858 (www.mobydic-project.eu) from European Community, TEC2008-04920 from the Spanish Ministry of Science and Innovation, and P08-TIC-03674 from the Andalusian regional Government. Some of the sequences included in this paper have been provided by Philips Research Laboratories, Eindhoven, The Netherlands.

References

[1] D. Nauck, R. Kruse, "NEFCON-I: an X-Window based simulator for neural fuzzy controllers," *Proc. IEEE International Conference on Neural Networks*, Orlando, 1994, pp. 1638–1643.

[2] D. Nauck, U. Nauck, R. Kruse, "NEFCLASS for Java-New learning algorithms," *Proc. IEEE International Conference North American Fuzzy Information Processing Society*, New York, 1999, pp. 472–476.

[3] Fuzzy Logic Toolbox from Matlab. Available at: http://www.mathworks.es/products/fuzzylogic/

[4] The Xfuzzy development environment for fuzzy systems. Available at: http://www.imse-cnm.csic.es/Xfuzzy

[5] N. Kanagaraj, P. Sivashanmugam, S.Paramasivam, "A fuzzy logic based supervisory hierarchical control scheme for real time pressure control," *International Journal of Automation and Computing*, vol. 6, pp. 88–96, 2009.

[6] L-X. Wang, "Analysis and design of hierarchical fuzzy systems," *IEEE Transactions on Fuzzy Systems*, vol. 7, pp. 617–624, 1999.

[7] Y. Chen, L. Peng, A. Abraham, "Programming hierarchical TS fuzzy systems," *Proc. IEEE International Symposium on Evolving Fuzzy Systems*, Lake District, 2006, pp. 157–162.

[8] Y. Chen, B. Yang, A. Abraham, L. Peng, "Automatic design of hierarchical Takagi-Sugeno type fuzzy systems using evolutionary algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 15, pp. 385–397, 2007.

[9] F. J. Moreno-Velo, I. Baturone, A. barriga, S. Sánchez-Solano, "Automatic tuning of complex fuzzy systems with Xfuzzy," *Fuzzy Sets and Systems*, vol. 158, pp. 2026–2038, 2007.

[10] G. de Haan, E. B. Bellers, "De-interlacing: An overview," *Proc. of the IEEE*, Sept. 1998, pp. 1839–1857.

[11] P. Haavisto, J. Juhola, Y. Neuvo, "Fractional frame rete up-conversion using weighted median filters," *IEEE Trans. on Consumer Electronics*, vol.35, pp.272-278, 1989.

[12] P. Haavisto, Y. Neuvo, "Motion adaptive scan rate up-conversion," *Multidimensional Systems Signal Processing*, no.3, pp.113-130, 1992.

[13] A. M. Bock, "Motion-adaptive standards conversion between formats of similar field rates," *Signal Processing: Image Communication*, vol.6, pp.275-280, 1994.

[14] H. Jiang, D. Huu, E. Tinyork, "Motion adaptive deinterlacing," *United States Patent (US 6,459,455)*, Oct. 2002.

[15] W-K. Lin, C-Y. Lu, "Method for motion pixel detection," *United States Patent (US 7,034,888)*, Apr. 2006.

[16] W-K. Lin, C-Y. Lu, "Method for motion pixel detection with adaptive thresholds.," *United States Patent (US 7,242,435)*, Jul. 2007.

[17] P. Brox, I. Baturone, S. Sánchez-Solano, J. Gutiérrez-Ríos, F. Fernández-Hernández, "A fuzzy edge-dependent motion adaptive algorithm for de-interlacing.," *Fuzzy Sets and Systems*, vol.158(3), pp.337-347, 2007.