Linear Interpolation in Time Domain for Improved Audio or Image Signal Reconstruction

Nikos Petrellis
Dept. of Computer Science and Engineering
TEI of Thessaly
41110 Larissa, Greece
npetrellis@teilar.gr

Abstract— **An extremely low cost and complexity hardware linear interpolator is presented in this paper used for the improved reconstruction of audio or image signals like the ones produced by surveillance optic or infrared cameras, X-ray images, sound detectors, microphones etc. Real time compression can also be performed if the audio or image signal has long sequences of identical values. A set of appropriate correction rules for exponential or logarithmic curves that are based on simple operations like shifts and comparisons can be employed since several parts of a signal can be approximated by such exponential or logarithmic curves. Applications in the Internet of Things (IoT) area can benefit from such an interpolator-compressor since lower resolution/power Analog Digital Converters (ADCs) can be used in lighter transceivers that exchange information with lower data rate. An improvement of about 15% in the Mean Square Error (MSE) has been measured using real signals. An interpolator with 9-bit input resolution can be implemented with only 583 Logic Elements (LE) i.e., less than 3% of an Altera Cyclone III EP3C25N Field Programmable Gate Array (FPGA) resources.**

Index Terms— **linear interpolation, image enhancement, audio enhancement, compression**

I. Introduction

The interpolation methods have been used for accurate continuous signal reconstruction by a number of discrete samples. The fundamental and easier to implement interpolation methods are the zero-order hold (which is actually the output of a Digital to Analog Converter-DAC) and the linear (or first-order hold) interpolation in the time domain. More sophisticated methods are the higher order interpolation methods in time domain like the cubic spline interpolation [1] or in the frequency domain: Lagrange, min-max interpolation, etc. The time domain linear interpolation is the most appropriate for low complexity hardware implementation since most of the other methods require more complicated operations (multiplications, divisions, matrix inversion, etc).

Interpolated values can be used for the approximation of missing data samples retrieved by ADCs that operate with low sampling rate. Although advanced signal approximation techniques can be implemented in software for off-line signal processing, these approaches are not appropriate for real-time operation that is necessary in IoT applications where there is an abundance of signal sources. Low order time-domain interpolators have been studied for decades but their low complexity/cost architecture remains the most appropriate for real time hardware implementations. In any case, the cost of the interpolator should not exceed the benefit from the use of a lower resolution ADC.

A linear interpolator is described in [2] where several interpolated values are inserted between successive known samples with the typical resolution increasing from 8 to 12 bits. The interpolator described in [2] has been fabricated in a 1×1 mm^2 CMOS integrated circuit. Several ADC output post-processing modules have been proposed for the correction of Differential or Integral Non-Linearity (DNL/INL) static errors. A histogram approach corrects the INL error of an ADC in [3] increasing its resolution by 4-bits.

Min-max and Lagrange interpolators are used in [4]. Several interpolating schemes are examined in [5] and [6] by Y. Eldar and T. Michaeli. The MSE for various parameters including interpolation rate is examined in [5]. The MSE is 30% higher than the optimal one for interpolation rate equal to 1 while the gap closes soon when the interpolation rate increases to 4. The authors measure the Peak Signal to Noise Ratio (PSNR) in [6] in various image reconstruction cases.

An adaptive example-based Super-Resolution (SR) method with a novel classification approach is presented in [7]. For the comparison against several other approaches, the Structured Similarity (SSIM) is used. Both SSIM and PSNR are used to compare the proposed interpolation method in [8] against Linear Filtering interpolation (LFI) (Cubic Spline-CS, Lanczos), Edge Directional Interpolation-EDI (NEDD, LSMD) methods.

As can be seen from the previous approaches different measures are used to demonstrate the quality of a proposed signal reconstruction method. The expected errors from various types of interpolators (linear, RC, Butterworth n-pole low pass filter) used at the output of ADCs is formally described in [9]. The SSIM is defined in [10].

In the first-order interpolation, the connection between two successive ADC samples is assumed to be linear. The shorter the successive sample distance is, the smaller the error of the first-order hold is. The curvature between the two successive values cannot be modeled by the first-order hold. The linear interpolation scheme described here is based on a first-order interpolation in the time domain that can be implemented with low cost hardware that does not require more complicated operations than additions, comparisons and counters. Error correction rules are defined for the case where the whole input signal or a part of it is logarithmic or exponential. Although there cannot be a common correction method for all types of curves, a strategy that reduces the error in the most common cases is adopted and tested.

The proposed interpolation method can increase the Signal to Noise and Distortion Ratio (SNDR) as well as the Spurious Free Dynamic Range (SFDR) of sinusoidal signals by up to 100% as was shown in [11]. In the present work we focus on real audio and image signals and see that a significant improvement of up to 15% can be achieved in their quality. The interpolator described in this paper can also perform real time compression in the case of sparse or signals with long sequences of identical samples. For example, the output of the pressure sensor was compressed to the 11% of its original size. This is due to the fact that the compressed output represents the signal in pairs of the form (V, C) where the V is a sample value that appears C times. However, if no identical values appear in successive samples in the original signal then no compression can be achieved.

The proposed interpolator was evaluated using reconfigurable hardware. An interpolator with 9-bits output along with its decompression unit required 583 Logic Elements (LEs) or less than 3% of the LEs of a Cyclone III EP3C25N FPGA.

The interpolation method and the corresponding architecture are described in Section 2. The error correction rules that can be employed when the input signal is exponential or logarithmic are presented in Section 3. Finally, experimental results are discussed in Section 4.

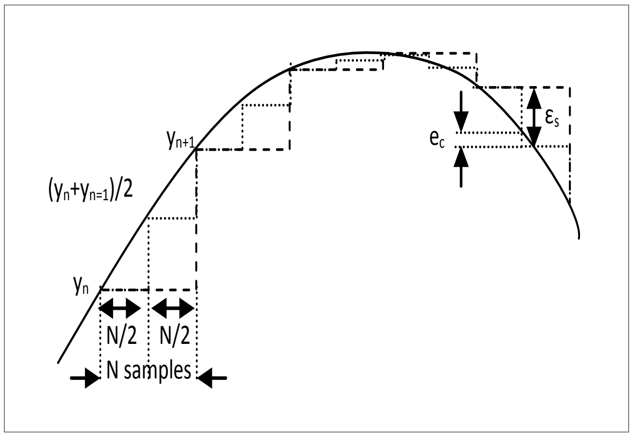


Fig. 1. The proposed interpolation method with no error correcting rules.

II. Interpolation Method

The proposed interpolation method is a combination of zero and first order hold as can be seen from Fig. 1. The continuous line is the real signal while the dashed line corresponds to the output of a DAC that would accept the discrete values y_n, y_{n+1}, etc at its input. These values can have been retrieved by an ADC that samples a real signal y(t) at the intervals $n{\cdot}T_a$, n=1,2,.. where T_a is the ADC clock period. Each one of the $y_n=y(n{\cdot}T_a)$ values appears for N interpolator clock periods T_i. The interpolator clock frequency f_i is equal or higher than the ADC sampling frequency thus, each y_n value appears multiple times at the DAC input. The ratio of the interpolator clock frequency and the ADC sampling rate may be in the order of 10 or higher but this can be achieved easily since it is performed by digital circuits while the sampling rate of the ADC is harder to be increased due to the conversion latency. Assuming y_n appears for N samples it is obvious that all but the first of these samples have an error:

$$\varepsilon_i = y(nT_a + t_i) - y(nT_a) \text{ with } t_i < T_a \,. \tag{1}$$

For monotonically increasing or decreasing y(t) in the interval $[n{\cdot}T_a,(n+1)T_a)$, the sum of the absolute error of all the N samples is decreased if half of them is replaced by the value $(y_n+y_{n+1})/2$ as shown in Fig. 1. Consider for example a monotonically increasing y(t). By definition:

$$0 < y(nT_a) < y(nT_a + t_i) < y(nT_a + t_{i+1}) \,. \tag{2}$$

It is obvious from (1) and (2) that:

$$\varepsilon_0 < \varepsilon_t < \varepsilon_{t+1} \,. \tag{3}$$

where ε_0 is 0 if the ADC and DAC are ideal or equal to e_c (see Fig. 1) if the ADC and DAC are non-ideal. If the last N/2 samples of the interval between y_n and y_{n+1} are replaced by the value: $(y_n+y_{n+1})/2$, the error ε_i' ($N/2 \le i < N$) of these samples are lower than ε_i since:

$$\begin{aligned}
&y((n+1)T_a) > y(nT_a) \Rightarrow \\
&y(nT_a) + y((n+1)T_a) > 2y(nT_a) \Rightarrow \\
&\frac{y(nT_a) + y((n+1)T_a)}{2} > y(nT_a) \Rightarrow \\
&y(nT_a + t_i) - \frac{y(nT_a) + y((n+1)T_a)}{2} < \\
&y(nT_a + t_i) - y(nT_a) \Rightarrow \varepsilon_i' < \varepsilon_i
\end{aligned} \;. \tag{4}$$

The dotted line in Fig. 1 shows the resulting interpolated signal. The replacement process described above can be recursively applied in order to estimate additional intermediate points. The generated intermediate points belong to the piece of line of a first-order hold. If the input signal does not change monotonically between successive sample values, a lower approximation error is not guaranteed. This may occur when the successive ADC outputs are retrieved using excessively long sampling intervals, compared to the signal frequency.

The error improvement that is achieved by recursive interpolations is gradually decreasing since they are applied on estimated intermediate values that are not identical to the real ones. For this reason, it is not worth using more than 3-4 interpolation stages because the cost increases for a negligible gain in the linearity. In [11] a three-stage interpolator has been tested with sinusoidal signals improving their linearity (expressed as SNDR) by up to 100% if the initial resolution is too low. Multiple stage interpolation can also be beneficial in the case of sensor signals that change with a slow rate. However, in the case of signals representing acoustic sound and images multiple interpolation stages do not show a significant error improvement and are not studied in this work.

The architecture of a circuit implementing the interpolator described above is shown in Fig. 2. The digital input has an M-bit resolution and is connected to a pair of latches (L1 and L2). The latch L1 uses the interpolator clock and when the input changes, the digital M-bit comparator CMP generates a short inequality pulse indicating that the successive samples appearing at the L1 input and output are different. The comparator output controls the reset of a counter that counts the sample appearance (C) through the Control Signal Generator. This logic is also responsible for updating the output of L2 (through the signal Enable) so that the previous different input is displayed by the L2. In this way L1 and L2 always hold the successive different values (e.g., y_n and y_{n+1}).

The Adder of Fig. 2 is used to estimate the average of y_n and y_{n+1}. If $|y_n-y_{n+1}|\leq 1$ then the interpolated value would be equal to y_n and the method would not have achieved any improvement. For this reason, the output of the interpolator has 1-bit higher resolution than the input (M+1 bits) and what is available at the output V is the double of the values y_n and y_{n+1} and their sum (y_n+y_{n+1}) instead of their average. The multiplexor outputs either the previous ADC value (shifted left by one bit or simply padded with a 0 as its Least Significant Bit - LSB), or the adder output and is controlled by the Control Signal Generator too. An external system can sample the (V, C) pairs at the appropriate time indicated by the Control Signal Generator and these pairs form the compressed output of the interpolator.

III. Error Correction Rules

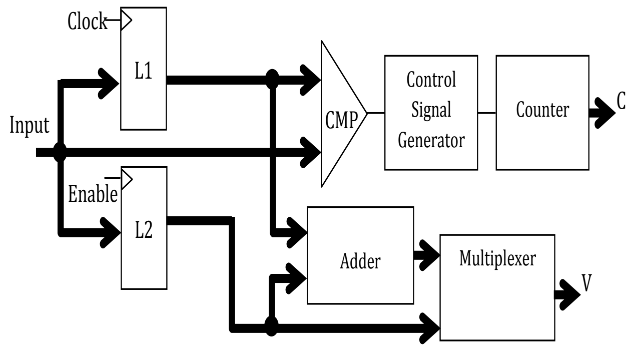


Fig. 2. The proposed interpolation method with no error correcting rules.

The error correction rules described here have been incorporated in a VHDL description implemented with reconfigurable hardware. Some specific features have to be taken into consideration in the case of acoustic and image signals. For example, in an acoustic signal with a long pause (with value 0) interval that is interrupted by a loud sound (with value y_l>>0) the interpolator would replace a large number of pause samples with the value $y_l/2$. This would cause significant distortion. Similarly wide dark areas in an image interrupted by bright islands (or vice versa) will suffer from distortion since a lot of stripes will appear by the replacement of several dark pixels in a row with pixels having intermediate brightness.

For this reason the interpolation is deactivated when either of the following conditions hold since a sudden, not smooth transition is recognized in the sound or image signal (the parameters Th1 and Th2 are application specific thresholds):

$$|y_n - y_{n+1}| > Th1 . \tag{5}$$

$$|N_n - N_{n+1}| > Th2 . \tag{6}$$

In the case of exponential or logarithmic input signals some different correction rules can be adopted that are simplified in order to be implemented with comparisons, additions and subtractions only. The following rules are based on the fact that the interpolator generates a new intermediate value between two successive known samples. If three successive known samples at regular intervals are considered the type of curvature of the input signal can be recognized as well as its slope.

The general form of a logarithmic signal is y_k=αlog(k)+ γ where k≥1. If γ is 0 and α is 1 its form is the one shown in Fig. 3. The interpolated value y_k is estimated as the sum of y_{k-1} and y_{k+1}. The ratio of the estimation error E1 between the real value of y_k and the estimated one: $\hat{y}_k$, to the error E2 between the real value of the sample y_{k-1} and the one that would have been estimated ($\hat{y}_{k-1}$), had the samples y_{k-3} and y_{k+1} been used is:

$$\begin{aligned}\frac{E2}{E1} &= \frac{y_{k-3} + y_{k+1} - 2y_{k-1}}{y_{k-1} + y_{k+1} - 2y_k} = \\ &\frac{\log(k-3) + \log(k+1) - 2\log(k-1)}{\log(k-1) + \log(k+1) - 2\log k} = \\ &= \frac{\log((k-3)(k+1)/(k-1)^2)}{\log((k^2-1)/k^2)}\end{aligned} \tag{7}$$

Equation (7) shows that the ratio E2/E1 for a logarithmic function does not depend on the factors α and γ, but depends on the index k. The ratio E2/E1 in (7) is not defined for k<4 and approaches 4 when k→∞ as can be shown by the L' Hopital theorem for the derivatives of a ratio.

An exponential signal is $y_k=\beta e^{\alpha k}+\gamma$. In a similar manner:

$$\begin{aligned}\frac{E2}{E1} &= \frac{e^{a(k-3)} + e^{a(k+1)} - 2e^{a(k-1)}}{e^{a(k-1)} + e^{a(k+1)} - 2e^{ak}} = \\ &\frac{1 + e^{4a} - 2e^{2a}}{e^{2a}(1 + e^{2a} - 2e^{a})} = (1 + e^{-a})^2\end{aligned} \tag{8}$$

If α>0 and β>0, then y_k is monotonically increasing and the ratio E2/E1 is between the values 1 and 2, thus E1≤E2≤2E1.

The same holds if $\beta<0$ but in this case y_k is monotonically decreasing. If $\alpha<0$, then E2>2E1. These relations between E1 and E2 depend only on the parameter α and not the index k.

By comparing the values of the known samples y_{k-3}, y_{k-1} and y_{k+1}, it can be determined whether the input signal is monotonically rising or falling (provided of course that the three known samples are close enough) and whether it is increasing (or decreasing) in an exponential or logarithmic-like way. The latter can be determined by the differences $y_{k-1}-y_{k-3}$ and $y_{k+1}-y_{k-1}$. Using this information the relation between E1 and E2 is determined and the calculated $\hat{y}_k$ is shifted towards y_k. For example, if $y_{k+1}\geq y_{k-1} \geq y_{k-3}$ and $y_{k+1}-y_{k-1}\geq y_{k-1}-y_{k-3}$ then the signal is recognized as a monotonically increasing exponential and the relation $E1\leq E2\leq 2E1$ should hold.

$E2= y_{k-3}+y_{k+1}-2y_{k-1}$ is estimated from known sample values. In the calculation of $E1=y_{k-1}+y_{k+1}-2y_k$, the estimated y_k sample value is modified appropriately in order to fulfill the closest of the relations: $1\leq E2/E1$ or $1\leq 2E1/E2$.

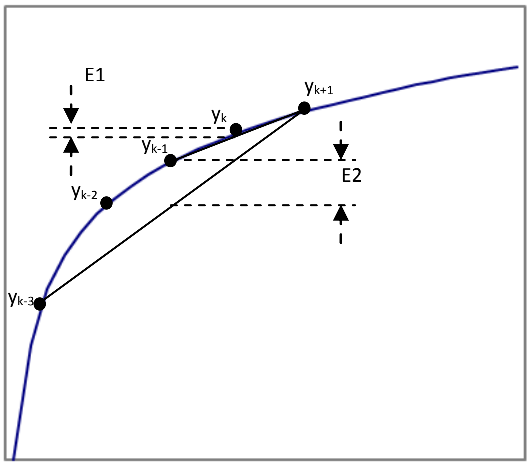


Fig. 3. Interpolation of a logarithmic signal.

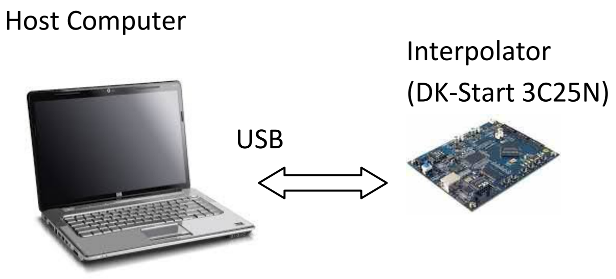


Fig. 4. Experimental setup.

IV. Experimental Setup

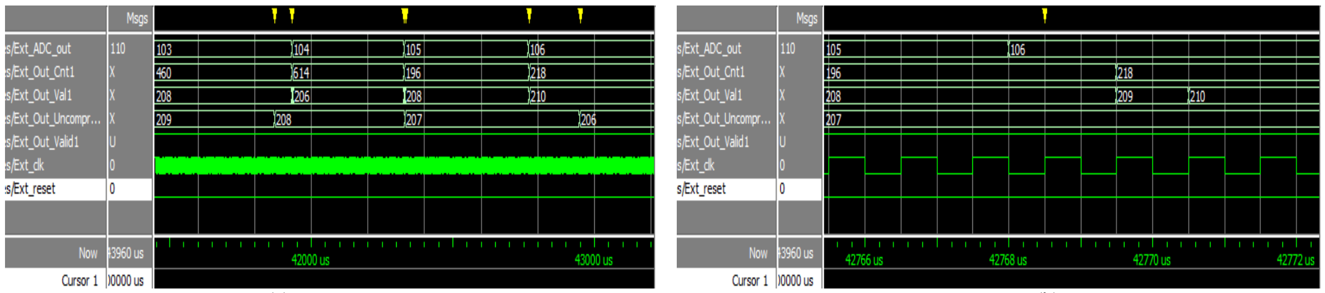
The experimental setup shown in Fig. 4 was used in order to test the developed interpolator that has been described in VHDL and implemented using an Altera DK-Start 3C25N evaluation board with Altera Cyclone III EP3C25N FPGA. The audio and image input stimuli as well as the interpolator outputs were transferred from/to the Host Computer through USB. The results were processed in MATLAB in order to measure the MSE/NMSE errors. The maximum interpolator clock is determined by the on-board oscillator of the DK-Start 3C25N (50 MHz) and was selected as 1 usec. As already mentioned an 8-bit input and 9-bit output interpolator required 583 LEs of the specific FPGA.

Fig. 5a shows the input/output of the interpolator signals during simulation at Modelsim. Fig. 5b shows a zoomed-in window of Fig. 5a where the generated interpolator output is shown more clearly. The top signal (Ext_ADC_out) of Fig. 5 is actually the input of the interpolator and its 8-bit values are displayed in decimal. The 2nd and 3rd signals (Ext_Out_Cnt1, Ext_Out_Val1) are the compressed outputs (C, V) of the interpolator. The 4th signal is the uncompressed interpolator output generated by a decompressor that has also been developed to test multiple interpolator stages. For example, the values 104 and 105 at the input are doubled at the compressed output (values 208 and 210) and the intermediate value (209) is generated. This value appears clearly at Fig. 5b along with the value 218 at the Ext_Out_Cnt1 signal that indicates that 209 (as well as 210) have to appear for 218 interpolator clock pulses at the uncompressed output.

A portion of a reconstructed violin G-tone appears at Fig. 6. The correction rules described in Section 3, were applied in Fig. 6a while in Fig. 6b no logarithmic or exponential correction was applied. However the rules that deactivate the interpolator if (5) or (6) hold, were applied, otherwise the MSE would be much worse. An 8-bit resolution signal is used as input to the interpolator while a 10-bit resolution version of the same signal is used as a reference. The 9-bit interpolator output also appears in this figure. A best-fit to the 10-bit reference signal has been applied for both the interpolator input and output in order to find the minimum MSE which was 15.3 for the interpolator input and 13.42 for its uncorrected output and 13.29 for the corrected output. Thus, the interpolator achieved a 12.28-13.13% improvement in the MSE.

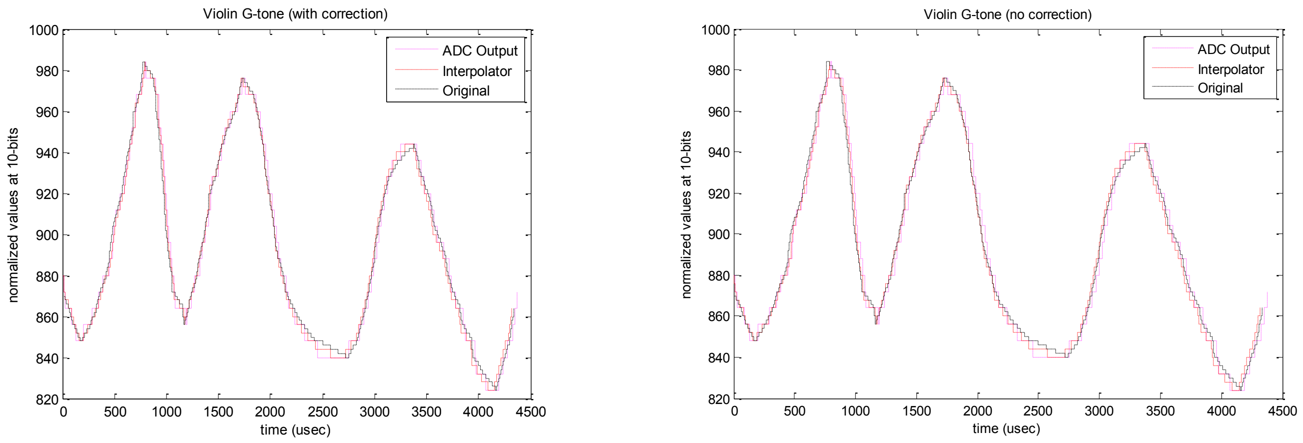
Fig. 7a shows an infrared image from a night-watch camera in grayscale. A 3-bit resolution version of this image is shown in Fig. 7b, while the corrected interpolator output is shown in Fig. 7c. The MSE of the images in Fig. 7b and 7c are 170 and 150 respectively achieving 11.76% improvement in the MSE.

A 2MHz and a 5MHz sinusoidal signal with 8-bit resolution were also tested as inputs of the developed interpolator. The initial SNDR that was 22dB and 13dB respectively was increased at 28dB and 18dB at the output of the developed interpolator. The SFDR was also increased from 27dB to 33dB for the 2MHz signal and from 18dB to 24dB for the 5MHz signal.



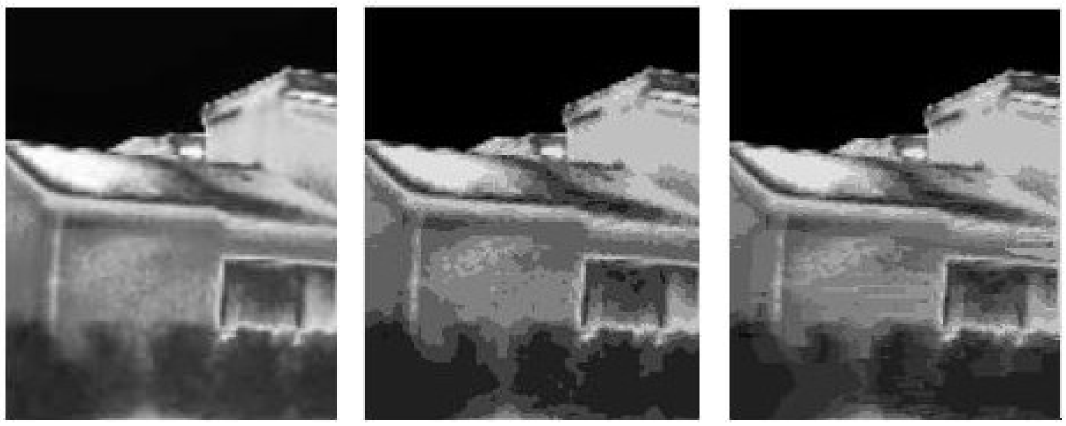
(a) (b)

Fig. 5. Simulation of the interpolator in Modelsim (a) and zoom-in (b).



(a) (b)

Fig. 6. Reconstructing a part of a violin G-tone with correction rules (a) and without correction rules (b).



(a) (b) (c)

Fig. 7. Infrared night-watch. Original image (a), 3-bit resolution (b) and interpolator output (c).

Table 1, describes the improvements achieved by the referenced approaches. The measures used in each reference are not strictly comparable but as was shown in [11] the proposed interpolator implemented with 3-stages can achieve significant improvement in the SNDR of up to 100% which is comparable or higher than the referenced approaches. Although the improvement in the MSE of real audio signals and images is lower, the fact that the interpolator implementation requires extremely lower hardware resources compared to the referenced approaches, along with its capability to perform compression, makes it valuable for several applications in the IoT domain.

I. Conclusions

A very low complexity linear interpolator, appropriate for the improvement of sensor indications was presented in this paper. It can be implemented with less than the 3% of the resources offered by a low cost FPGA like Altera Cyclone III

EP3C25N. The improvement in the representation of audio and image signals was examined as a case study demonstrating how IoT applications can benefit from the use of such an interpolator. The MSE of the real signals tested was improved by about 15%. Real time compression can also be performed by the proposed interpolator. Future work will include the implementation of the proposed interpolator in Application Specific Integrated Circuits (ASICs).

TABLE I. COMPARISON WITH OTHER APPROACHES

Ref	Improvement	Conditions-Notes
[2]	SNDR increased from 51dB to 69dB	300kHz signal oversampled at 3MS/s.
[3]	ADC Resolution increased by from 10 to 14-bits (40%)	INL correction
[5]	MSE close to optimal when interpolation rate is 4	30% higher MSE when interpolation rate is 1. The gap is closed gradually when interpolation rate increases
[6]	PSNR of various interpolation schemes between 22.5dB and 24.5dB	Including bicubic and minmax regret interpolation methods
[7]	22.5%-35% increase of the SSIM of a low resolution image	Compared to several other techniques that achieve 12%-28% improvement
[8]	Average PSNR=26.45dB Average SSIM=0.792	Compared to Linear Filtering interpolation (LFI) (Cubic Spline-CS, Lanczos) and Edge Directional Interpolation (EDI) (NEDD, LSMD) that achieve average PSNR between 25.54dB and 26.19dB and SSIM between 0.768 and 0.778.
[11]	- SNDR increased by 20%-100% (from 34.7dB to 42.1dB or from 22dB to 40dB) - SFDR increased by 100% (from 27dB to 54dB)	Tested on sinusoidal signals. The results here retrieved by a 3-stage interpolator
This work	Audio signal MSE improved by 13% (from 15.3 to 13.29-13.42) Low res/tion image MSE improved by 11.76% (from 170 to 150)	One stage interpolator. 3% of the LEs of an Altera Cyclone III EP3C25N FPGA used for a 9-bit resolution. Compression performed with a rate of downto 11%.

ACKNOWLEDGMENT

This work is protected under the provisional patent 1008346, published by the Greek Patent Office (Nov. 3, 2014).

REFERENCES

[1] S. A. Dyer and J. S. Dyer, “Cubic Spline Interpolation,” IEEE Instrumentation and Measurement Magazine, vol. 4, no. 1, pp. 44-46, 2001.

[2] H. W. Wang, C. F. Chan and C. S. Choy, “High speed CMOS digital-to-analog converter with linear interpolator,” IEEE Transactions on Consumer Electronics, vol. 46, no. 4, pp. 1137-1142, 2000.

[3] L. Jin, D. Chen and R. Geiger, “A digital self-calibration algorithm for ADCs based on histogram test using low-linearity input signals,” Proceedings of the IEEE ISCAS’05, Kobe, Japan, pp. 1378 – 1381. May 23-26, 2005.

[4] J. J. Fuchs and B. Delyon, “Min-max interpolators and Lagrange interpolation formula,” Proceedings of the IEEE ISCAS ’02, Phoenix-Scottsdale, AR, No. 4, pp. 429-432, May 26-29, 2002.

[5] T. Michaeli and Y. Eldar, “High-Rate Interpolation of Random Signals from Nonideal Samples,” IEEE Transactions on Signal Processing, vol. 57, no. 3, pp. 977-992, Mar. 2009.

[6] Y. Eldar and T. Michaeli, “Beyond bandlimited sampling,” IEEE Signal Processing Magazine, vol. 26, no. 3, pp. 48-68, May 2009.

[7] T. Ogawa and M. Haseyama, “Adaptive example-based super-resolution using kernel PCA with a novel classification approach,” EURASIP Journal on Advances in Signal Processing, vol. 2011, no. 138, 2011.

[8] D. S. Yoo, J. Chang, C. H. Park and M. G. Kang, “Video resampling algorithm for simultaneous deinterlacing and image upscaling with reduced jagged edge artifacts,” EURASIP Journal on Advances in Signal Processing, vol. 2013, no. 188, 2013.

[9] P. Garrett, Advanced Instrumentation and Computer I/O Design, J. Wiley and Sons, 2013

[10] Z. Wang, A.C. Bovik, H.R. Sheikh and E.P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, 2004.

[11] N. Petrellis, “A Scalar Interpolator/Compressor for the Improvement of Sensor Linearity,” Proceedings of the 4th International Conference on Wireless Mobile Communication and Healthcare (Mobihealth), Athens, Greece, Nov. 3-5, 2014.