1 Appendix A

2 Appendix B - BIBO Gain Analysis (alternate approach)

To determine the best fixed point representation for the hardware implementation of the lifting structure, the gain of each lifting step had to be found. Knowing this, it would be possible to know where to keep the 'virtual' decimal point to ensure correct implementation and avoid overflow. It was tempting to use conventional methods of BIBO analysis whereby the absolute sum of the filter coefficients is found and this is considered the upper bound. However this would not provide us with insight into the internal working of the lifting architecture, not reflect its unique structure.

To calculate the BIBO gain of the lifting implementation the system is treated as a linear open loop control system, which has two inputs, four states and two outputs, Figure 1.

The analysis becomes quite simple when we write the system in the form:

$x(k+1) = \boldsymbol{A}x(k) + \boldsymbol{B}u(k)$

$y(k) = \boldsymbol{C}x(k) + \boldsymbol{D}u(k)$

We then write the expressions for the four state values:

$x_4(k+1) = x_3(k) + \gamma\left[x_2(k) + \beta[x_3(k) + u_2(k) + \alpha\left(x_4(k) + u_1(k)\right)]\right]$

$x_3(k+1) = x_4(k) + \beta\left[x_3(k) + u_2(k) + \alpha\left(x_4(k) + u_1(k)\right)\right]$

$x_2(k+1) = u_2(k) + \alpha\left(x_4(k) + u_1(k)\right)$

$x_1(k+1) = u_1(k)$

Next, we write the output in terms of the inputs, and state values:

$y_1(k) = x_1(k)$

$???y_2(k) = x_2(k)+\delta[x_3(k)+\gamma\left[x_2(k) + \beta[x_3(k) + u_2(k) + \alpha\left(x_4(k) + u_1(k)\right)]\right]]$

Hence, we have our system expressed as:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}(k+1) = \begin{pmatrix} 0 & \gamma & \beta\gamma+1 & \gamma\left(\alpha\beta+1\right) \\ 0 & 0 & \beta & \alpha\beta+1 \\ 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}(k) + \begin{pmatrix} \alpha\beta\gamma & \beta\gamma \\ \alpha\beta & \beta \\ \alpha & 1 \\ 1 & 0 \end{pmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}(k)$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}(k) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \delta & \gamma\delta+1 & \delta(\beta\gamma+1) & \gamma\delta(\alpha\beta+1) \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}(k) + \begin{pmatrix} 0 & 0 \\ \alpha\beta\gamma\delta & \beta\gamma\delta \end{pmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}(k)$$

Here, it is important to observe that the A matrix is nilpotent of the fourth degree, i.e.:

$\boldsymbol{A}^4 = 0$ (which is expected, because this indicates that the eigenvalues of the matrix are all zero, which is a characteristic of an FIR filter).

Hence, we can express the output as a finite sum of the state terms and the input:

$\mathbf{y}_k = \mathbf{C}(\mathbf{B}\boldsymbol{u}_{k-1} + \boldsymbol{A}(\boldsymbol{B}\boldsymbol{u}_{k-2} + \boldsymbol{A}(\boldsymbol{B}\boldsymbol{u}_{k-3} + \boldsymbol{A}(\boldsymbol{B}\boldsymbol{u}_{k-4} + \boldsymbol{A}\boldsymbol{x}_{k-4})))) + \boldsymbol{D}\boldsymbol{u}_k$

but

$\boldsymbol{A}^4 x_{k-4} = 0$

simplifying the above expression:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}(k) = \begin{pmatrix} \boldsymbol{CA^3B} & \boldsymbol{CA^2B} & \boldsymbol{CAB} & \boldsymbol{CB} & \boldsymbol{D} \end{pmatrix} \begin{bmatrix} \boldsymbol{u}_{k-4} \\ \boldsymbol{u}_{k-3} \\ \boldsymbol{u}_{k-2} \\ \boldsymbol{u}_{k-1} \\ \boldsymbol{u}_k \end{bmatrix}$$

This is a linear mapping $\Phi : \mathbb{R}^{10} \longrightarrow \mathbb{R}^2$

However, we are posed with an initial problem, namely that at first glance we see that even though our inputs are 8-bit numbers and hence the input range is: ± 127, the number of possible input combinations that we must test in order to determine the largest output is: $(256 \times 256)^5 = \left(2^{16}\right)^5 = 2^{80}$! This is a difficult task even for todays super computers. The problem can be greatly simplified if we observe an interesting characteristic of linear systems, namely that extreminities (edges or points) in $\mathbb{R}^n$ will be mapped to extreminites in $\mathbb{R}^m$. Here is a simple proof of this property:

Proof. If we have two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$in $\mathbb{R}^n$, an edge is defined as the set: $\alpha\boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2$ (as α varies between zero and one). Now, the matrix $\mathbf{T}$ maps points from $\mathbb{R}^n$to $\mathbb{R}^m$. Applying the transformantion to our two vec-

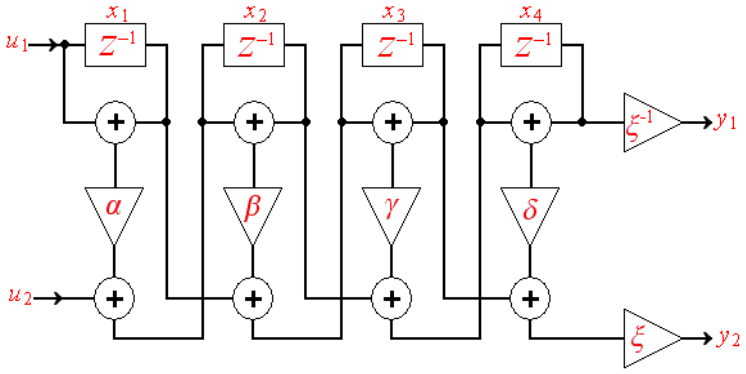

Figure 1: Lifting structure as an open-loop control system

tors: $\mathbf{T}(\alpha \boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2) = \alpha \mathbf{T}(\boldsymbol{x}_1) + (1-\alpha)\mathbf{T}(\boldsymbol{x}_2)$ (by linearity of matrix mappings) Which is just a representation of an edge in $\mathbb{R}^m$. ■

It is possible that this transformation maps edges to points, and because in our particular case we are going from $\mathbb{R}^{10} \longrightarrow \mathbb{R}^2$ it's quite possible that $\mathbf{T}(\boldsymbol{x}_1) = \mathbf{T}(\boldsymbol{x}_2)$, even if $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are different. However this does not concern us here, what is important is that we have restricted the range of possible inputs to lead to the maximum outputs from $2^{80} \longrightarrow 2^{10}$.

Assuming that the input is uniformly distributed, the derived BIBO gain will be the exact largest possible output. In addition, this method provides us with an opportunity to examine the input combinations which lead to the largest output value. Then if we know the image statistics for the possible set of used images, we can determine whether such an input combination is possible. In this way we can achieve the optimal integer-fractional format for the lifting filter.