A Visual C++ Based Software Tool for Visually Teaching Discrete Convolution from the Perspective of the Input Signal in Digital Signal Processing

S. Easwaran

Department of Computer Sciences and Computer Engineering
Xavier University of Louisiana

Abstract

This paper describes an approach and a novel software tool that was developed and used by the author of this paper to visually teach discrete convolution to students encountering it for the first time. In order to accomplish this to aid understanding of concepts without losing rigor or completeness, two novel software tools each viewing discrete convolution from a different perspective were designed and implemented by the author in Microsoft Visual C++. This paper describes one of these two software tools. The other is described by the author in a companion paper[1]. The software tool described in this paper was designed and implemented to visually teach discrete convolution by interpreting the discrete convolution equation from the perspective of the input signal in Linear Shift-Invariant systems. There are no other software tools to visually teach digital convolution from this perspective. By using the approach and software tool described in this paper, it was possible to visually teach discrete convolution from the perspective of the input signal very clearly and thoroughly in a shorter amount of time than was otherwise possible.

Introduction

Discrete (or digital) convolution is a fundamental operation and concept that is used extensively in the field of Digital Signal Processing (DSP). It is the basis of many DSP techniques, and it provides a mathematical framework for DSP[3-8]. However, it is also one of the most difficult techniques to understand and implement especially for those encountering it for the first time[7].

In order to provide a complete and proper understanding of this fundamental concept to students encountering it for the first time, a systematic approach of teaching this concept and topic was adapted and developed from various sources by the author[3-8]. The approach used was by first providing discrete convolution as a mathematical operation. It was then interpreted and explained as an input-output relationship for Linear Shift-Invariant (LSI) systems. Based on this, discrete convolution was explained as an operation that can be interpreted from two different perspectives of LSI systems – namely, from the perspective of the input signal, or from the perspective of the output signal.

Further, in order to visually teach these interpretations to aid understanding of concepts without losing rigor or completeness, two novel software tools each viewing discrete convolution from a different perspective were designed and implemented in Microsoft Visual C++ by the author of this paper. This paper describes one of these two software tools. The other is described by the author in a complementary, companion paper[1]. The software tool described in this paper was designed and implemented to visually teach discrete convolution by interpreting the discrete convolution equation from the perspective of the input signal in Linear Shift-Invariant (LSI) systems.

Using this software tool, it was relatively easy to very clearly explain discrete convolution from the perspective of the input signal to the students encountering it for the first time. It was also possible to teach this topic in a shorter time while enabling the students to master its concepts. The approach that was adapted and implemented by the author is as given below. The software tool that was designed and implemented by the author is described and discussed thereafter.

Discrete Convolution

Discrete Convolution as a Mathematical Operation

One of the most basic mathematical operations in any field is the operation of addition. The operation of addition (denoted by an operation symbol +) is a mathematical operation that takes any two numbers (a and b) and produces a third number (c = a + b). Similarly, the operation of multiplication (denoted by an operation symbol x) is another mathematical operation that takes any two numbers (a and b) and produces a third number (c = a x b).

Likewise, discrete convolution (denoted by an operation symbol *) is defined as a mathematical operation that takes any two digital signals or sequences (represented as {x[n]} and {h[n]}) and produces a third digital signal or sequence ({y[n]} = {x[n]} * {h[n] }). Here, in general terms, any digital signal or sequence is represented by {s[n]}. It is a collection of sequentially indexed and ordered set of numbers, i.e., {s[n]} = {…, s[-1], s[0], s[1], s[2], ..., s[k], …} where s[k] is the value of the discrete signal or sequence at index "k". The operation of discrete convolution is denoted and defined by the equation[3-8]

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{k=+\infty} x[k]h[n-k] : \forall n . \qquad \text{(Equation 1)}$$

Discrete Convolution as an Input-Output Relationship for LSI Systems

There is a large and important class of very useful digital systems known as Linear Shift-Invariant (LSI) systems. As its name implies, an LSI system is linear and shift-invariant with respect to its operations on input signals or sequences. Linearity implying that if the output for an input $\{x_1[n]\}$ is $\{y_1[n]\}$, and the output for an input $\{x_2[n]\}$ is $\{y_2[n]\}$, then the output for an input $\{ax_1[n] + bx_2[n]\}$ (which is a linear combination of the two independent, original inputs; and a and b are constants) is $\{ay_1[n] + by_2[n]\}$ (i.e., the same linear combination of the corresponding independent, original outputs). Shift-invariance implying that if the output for an input {x[n]} is {y[n]}, then the output for an index-shifted input {x[n-k]} is {y[n-k]} (i.e., the

new output is the original output index-shifted by the same amount as the index-shift in the original input, the index-shift being "k").

An LSI system is completely determined by its impulse response {h[n]} which is different for different LSI systems[3-8]. Here, the impulse response {h[n]} is the response (output) of an LSI system to a unit-impulse input signal or sequence, {δ(n)}. It can be shown[3-8] that for an LSI system defined by its impulse response {h[n]}, the output signal or sequence {y[n]} for an input signal or sequence {x[n]} is given by the equation,

$$y[n] = \sum_{k=-\infty}^{k=+\infty} x[k]\, h[n-k] : \forall n \qquad \text{(Equation 2)}$$

Therefore, based on the definition of discrete convolution (Equation 1) and the above equation for the computation of the output of an LSI system (Equation 2), it is evident that the output signal or sequence ({y[n]}) of an LSI system is obtained by the discrete convolution of the input signal or sequence with the system's impulse response sequence (i.e., {y[n]} = {x[n]} * {h[n]}).

Thus, the operation of discrete convolution can be viewed as a fundamental input-output relationship for LSI systems. It describes how the LSI system changes an input signal (or sequence) into an output signal (or sequence). To compute the output signal for such an LSI system, it is necessary to merely perform a discrete convolution operation of the input signal {x[n]} with the impulse response {h[n]} of the system.

Performing a Discrete Convolution

The Input-Side LSI System Interpretation of the Discrete Convolution Equation

The discrete convolution equation (Equation 2) can be interpreted and understood from two different perspectives for LSI systems[3-8]. They are from the perspective of the input signal, or from the perspective of the output signal[1]. The two perspectives (interpretations) are merely two different visualizations (interpretations) of the same mathematical operation and equation.

In the input side interpretation of the discrete convolution equation (Equation 1 or 2), the input signal or sequence **S** = {x[n]} is decomposed into a collection of independent and additive constituent sequences, … $\mathbf{S}_{-2}$, $\mathbf{S}_{-1}$, $\mathbf{S}_0$, $\mathbf{S}_1$, $\mathbf{S}_2$, …; -- i.e., **S** = … + $\mathbf{S}_{-2}$ + $\mathbf{S}_{-1}$ + $\mathbf{S}_0$ + $\mathbf{S}_1$ + $\mathbf{S}_2$ + …, where sequence $\mathbf{S}_k$ = {x[k] δ[n-k]} : k = …-2, -1, 0, 1, 2, …. This input decomposition into additive component sequences is possible because of linearity. In the above, each component sequence $\mathbf{S}_k$ (k = …-2, -1, 0, 1, 2, …) is composed of a single constituent element (x[k] δ[n-k]) of the original composite sequence {x[n]} with the remaining elements in that sequence being zero. That is, each constituent sequence $\mathbf{S}_k$ (k = …-2, -1, 0, 1, 2, …) is composed of a single scaled (weighted) and shifted, unit-impulse (i.e., a unit-impulse scaled or weighted by x[k] and shifted by "k") with the remaining elements in that sequence being zero.

Each of the above constituent (component) sequence $\mathbf{S}_k$: (k = …-2, -1, 0, 1, 2, ….) of the input sequence "**S**" is then independently passed through the LSI system, and the overall output y[n] is

synthesized (added) from the corresponding component outputs. Again, this additive synthesis of the output from the output components is possible because of linearity.

Since the impulse response h[n] of an LSI system is defined as the output sequence for a unit-impulse input δ[n], the output due to a "k" shifted impulse input δ[n-k] is h[n-k]. This is because of shift-invariance. Thus, the output $y_k[n]$ at any index "n" due to a single input sample x[k] (i.e., due to an x[k] weighted impulse given by x[k] δ[n-k] at "k") is x[k] h[n-k]. That is, $y_k[n]$ = x[k] h[n-k]. This is because of linearity (hence, scalability) and shift-invariance. This output x[k] h[n-k] is the output due to the independent, constituent set or sequence S_k. Thus, the overall composite output y[n] at any index "n" due to the entire input signal (i.e., $\mathbf{S} = \ldots + \mathbf{S}_{-2} + \mathbf{S}_{-1} + \mathbf{S}_0 + \mathbf{S}_1 + \mathbf{S}_2 + \ldots$) is obtained by summing the independent component outputs at index "n". It is given by,

$$y[n] = \sum_{k=-\infty}^{k=+\infty} y_k[n] = \sum_{k=-\infty}^{k=+\infty} (x[k]\, h[n-k]) : \forall n .$$

This equation is the same as the defining equation for discrete convolution of {x[n]} and {h[n]} (Equation 1).

Thus, discrete convolution can be interpreted from this input decomposition and output synthesis perspective. This perspective of interpretation of the discrete convolution equation enables one to analyze how each sample of the input signal or sequence impacts many sample components of the output signal or sequence. Each sample x[k] in the input signal will contribute a scaled (scaled or weighted by x[k]) and shifted version of the impulse response (x[k].h[n-k]) to the output signal y[n].

A Novel Software Tool for Teaching Discrete Convolution from the Perspective of the Input Signal

Discrete convolution is one of the most difficult techniques in DSP to visualize, understand, interpret and implement especially for those encountering it for the first time[7]. This is because the concept of discrete convolution and its effect on discrete-time signals are not intuitive from its defining equation (Equation 1) whose definition and mathematics seems confusing at first.

Because of this, discrete convolution was taught by the author along the above systematic presentation and approach, and by providing visual explanations of its interpretations. This was accomplished by specifically designing and implementing two novel software tools. Each software tool was designed to visually explain the interpretation of the discrete convolution equation from one of two different perspectives – namely, from the perspective of the input signal, or from the perspective of the output signal. In this paper, the software tool that was designed and implemented to teach discrete convolution from the perspective of the input signal of LSI systems is described. The software tool that was designed and implemented to teach discrete convolution from the perspective of the output signal of LSI systems is described by the author in a complementary, companion paper[1].

The User Interface and Features of the Software Tool

The user interface of the software tool that was designed and implemented to teach discrete convolution from the perspective of the input signal of LSI systems is shown in Figure 1. This user interface was designed and implemented to be as simple and intuitive as possible with regard to teaching and learning without sacrificing its powerful pedagogical features.

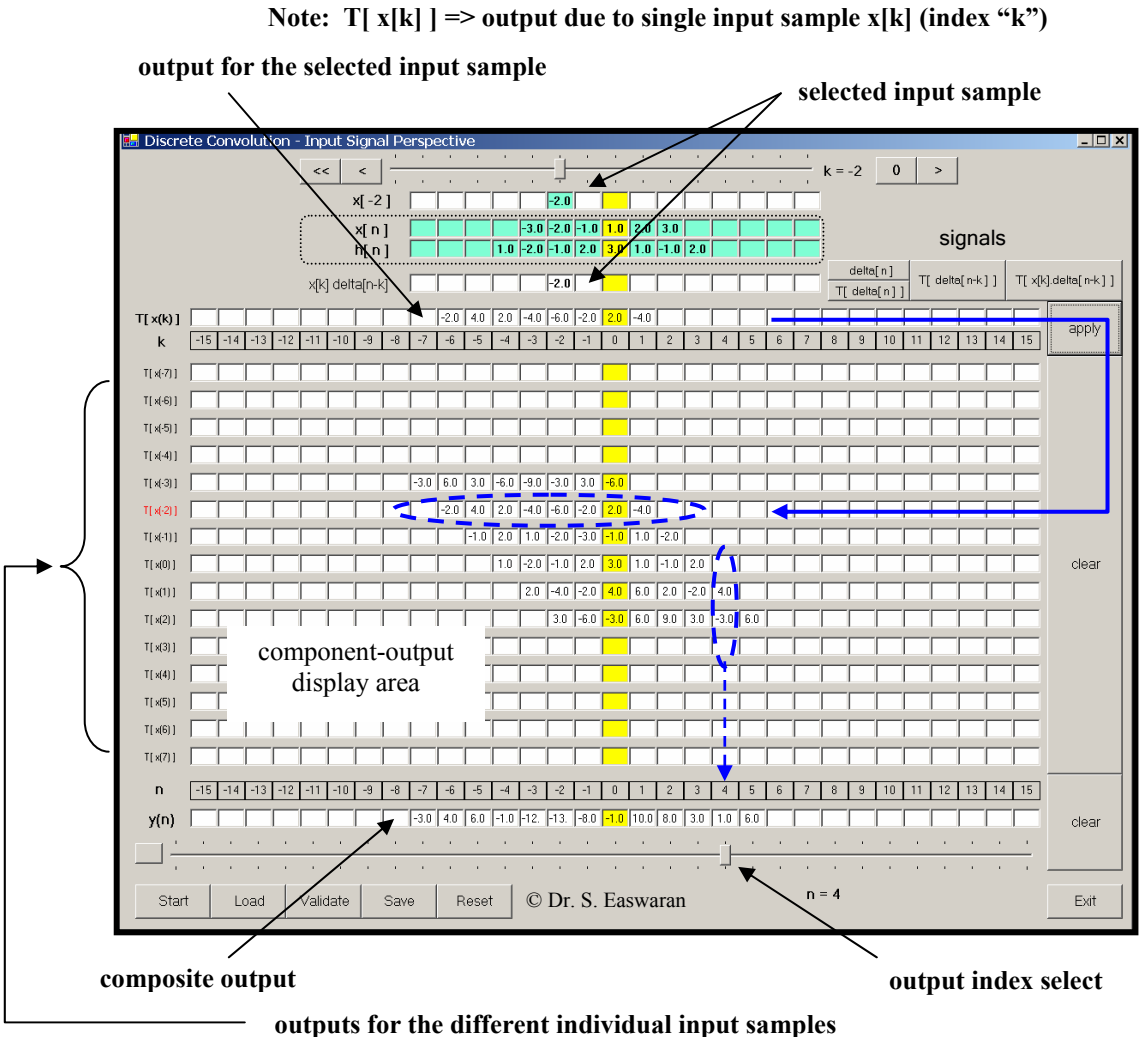

Figure 1: User Interface of Software Tool for Discrete Convolution (Input Signal Perspective)

It should be noted that prior to teaching discrete convolution, the students were taught the different forms of digital signal (or sequence) representations -- namely, the "sequence form" representation, the "graphical lollipop form" representation, and the "analytical form" representation. These representations were visually taught to the students through the use of still another software tool[2] designed and implemented by the author of this paper. It was developed to aid students in their understanding of digital signal representations, and of fundamental digital signals when encountering them for the first time.

In the software tool described in this paper, signals are diagrammatically represented using their "sequence form" representation {…, s[-2], s[-1], s[0], s[1], s[2], …} as opposed to their "graphical lollipop form" representation. This form of representation (via a diagram) was used in order to simplify the teaching and to better aid students in their understanding of the operation of discrete convolution from the perspective of the input signal. This representation was selected also because such a representation can be easily and directly mimicked on paper when there is a need to quickly perform a discrete convolution by hand ("pencil and paper" method).

The Description of the Software Tool, and its use in Teaching Discrete Convolution

Initially, the input signal or sequence {x[n]} and the system impulse-response signal or sequence {h[n]} are manually entered (by typing their values), or are loaded (using the "Load" button -- lower, left of user interface) (Figure 1) from previously stored values into their respective "bars". These "bars" are labeled, "x[n]" and "h[n]" respectively in the user interface (near top).

After the data is manually entered or loaded from previously stored values into their respective "bars", this data is validated by clicking the "Validate" button (bottom, left). Once the data is validated, their values can be stored for subsequent use by this or other programs[1] if needed. The storing of this data ({x[n]} and {h[n]}) is done using the "Save" button (bottom, left).

After the data is validated by the software, the process of performing discrete convolution from the perspective of the input signal begins. The scrollbar or "audio cassette-player" like buttons in the top area of the user interface are used to scan and select a single input-sample (x[k]) from the input signal. The signal value (x[k]) at the selected sample index ("k") is displayed in the display box below this input-sample selection scrollbar. Since discrete convolution is viewed from the input signal perspective (one input sample at a time) in the perspective of interpreting the discrete convolution equation described in this paper, this single input-sample selection (input-signal decomposition) is demonstrated and explained to the students first.

Thereafter, the students are shown a delta (unit sample or impulse) function (by displaying it in "sequence form" in the fourth bar from the top) by clicking the "delta(n)" button. Then, by clicking the "T[delta(n)]" button, the output sequence due to this delta function (as input) is displayed in the "T[x[k]]" bar (fifth bar) – it is simply the impulse-response sequence of the system (by definition). By clicking the button "T[delta(n-k)]", the output due to a shifted delta function whose unity value is shifted to the location of the previously selected input sample is displayed (fifth bar from top). This output is simply the commensurately shifted impulse-response of the system (due to shift-invariance). By clicking the button "T[x[k] delta(n-k)]", the output due to a x[k] (previously selected sample value) weighted delta-function shifted to the location of the previously selected input sample is displayed (in the fifth bar from top). This is the output of the system due to a single (pre-selected) input sample of the input sequence. It is displayed in the bar labeled "T[x[k]]" (fifth bar from the top). This concept of the output due to a single input sample is interactively explained to the students using this software tool for demonstration.

After the output due to a single input-sample is demonstrated and explained to the students, the "Apply" button (upper right) is clicked to transfer this component output sequence due to a single input sample into the component-output display area. It is transferred and displayed in the row of the component-output display area corresponding to its input sample index.

In the example figure shown (Figure 1), the sample index of the selected input signal is "k = -2" (its sample value too is "-2"). Thus, the output due to the input sample x[-2] (displayed in bar labeled T[x[k]]) is transferred to the row marked "T[x[-2]]" in the component-output display area, when the "Apply" button is clicked.

The above demonstration and explanation is repeated for each of the input samples (starting from input sample index "k = -7" through input sample index "k = 7") while explaining, transferring, and displaying the output due to each of the input samples of the input signal. At this point, the output due to each of the input samples of the input signal is separately displayed in its respective "bar" (accordingly labeled) in the component-output display area.

Once the outputs due to each of the input signal samples are displayed in the component-output display area, the process of explaining the computation of the composite output (output synthesis) to the students begins.

This is accomplished using the output index selection scrollbar located at the bottom of the user interface. As the scrollbar index is scrolled through each of the output indexes (n = -15 through 15), the composite output at the selected output index is displayed by summing the individual component-outputs at that output index. Thus, the entire output is displayed in the composite output display bar (bottom of the user interface) once the output index selection has scrolled through all the output indexes.

By using this software tool, the process and steps involved in performing discrete convolution (from the perspective of the input signal) was visually demonstrated and taught in a clear step-by-step fashion as explained, repeating any step or the whole process until the process of discrete convolution from this perspective was understood and mastered by the students.

The inputs {x[n]} and {h[n]} that were used in the discrete convolution operation can be stored using the "Save" button for performing the same discrete convolution operation later, but (this time) from the perspective of the output signal (instead of the perspective of the input signal). This was demonstrated and performed using a separate tool (discrete convolution from the perspective of the output signal) developed by the author[1] for explaining the interpretation of the discrete convolution equation from that perspective. The purpose of using the same data was for comparing the results obtained by the two different interpretations (input and output) of the discrete convolution equation, and for showing to the students that both these interpretations result in the same final output as it ought to. By using these two software tools for demonstrating discrete convolution, it was possible to elegantly and visually teach by demonstration, and make students better understand discrete convolution and its interpretations from both (input and output) perspectives. Once the students understood this process and steps, they were able to easily "reproduce" (mimic) these actions on paper, and were able to very

quickly do any discrete convolution of two signals by hand ("pencil and paper") using each of the two interpretations.

Student Performance Statistics

Data on student performance based on test scores before and after using this tool for teaching discrete convolution is limited as the Computer Engineering program at Xavier University of Louisiana is relatively new with a limited number of students. A total of twelve students were taught digital convolution using this software tool during the academic year 2004–2005 at Xavier.

Initially, the students were taught discrete convolution without using this software tool as the software tool was under development. As their test scores on discrete convolution were not satisfactory, they were taught discrete convolution again at a later time during the same semester using this software tool once its development was completed. The students were then tested by asking them to compute the output of two separate discrete systems by hand ("pencil and paper") based on the perspective of the input signal, given the input signal and the system's impulse response for each of the two systems. The test score of each student was scaled to a total maximum of 10 points and the test statistics were computed for the 9 (of the total 12) students who had attended all the lectures when discrete convolution was taught without and when it was taught with this software tool.

The statistics for the pre and post-test were as follows: average 6.39 with standard deviation 1.51, and average 9.22 with standard deviation 0.92, respectively. Though the results tends to indicate significant improvements in student learning when taught using this software tool, more student performance data and more careful testing and comparison strategies that are also practical are needed for meaningful statistical measures, and to draw statistically strong conclusions. The students were also given a tool-evaluation form after they had been taught discrete convolution from the perspective of the input signal, and asked how strongly they agree (from "very strongly" = 5, to "strongly disagree" = 1) that their learning and understanding of discrete convolution had improved based on this software tool. All nine students who had attended all the lectures before and when discrete convolution was taught using this software tool were asked to provide their feedback. They all "very strongly" agreed that their learning and understanding of discrete convolution from the perspective of the input signal had improved when it was taught using this software tool.

Summary and Conclusion

Discrete convolution is an operation and concept that is used in the mathematics of many scientific and technical fields including the field of DSP. However, it is one of the most difficult techniques in DSP to understand and implement especially for those encountering it for the first time. Discrete convolution can be interpreted as an input-output relationship for LSI systems. It can be interpreted from the perspective of the input signal or from the perspective of the output signal in LSI systems. A software tool for teaching discrete convolution by interpreting the discrete convolution equation from the perspective of the input signal of LSI systems was designed and implemented by the author, and was discussed in this paper. This software tool

was used by the author to visually teach discrete convolution from this perspective. Using the software tool and approach described in this paper, it was relatively easy to explain this concept to the students encountering it for the first time. It was possible to teach this topic in a shorter time than was otherwise possible while enabling the students to understand this concept well.

Bibliography

1. S. Easwaran, "An Innovative Software Tool for Teaching Discrete Convolution form the Perspective of the Output Signal in Digital Signal Processing: Its Software Design and Implementation, and Usage in Teaching and Learning", American Society for Engineering Education (ASEE), Annual Conference, Portland, Oregon (June 12-15, 2005). Paper Accepted.

2. S. Easwaran, "A Suite of Software Tools Developed in Microsoft Visual C++ for Teaching Discrete-Signal Representations, and Fundamental Discrete Signals, namely -- Real and Complex Exponential Signals, Impulse Signals, Step Signals, and Sinusoidal Signals" -- (unpublished), paper preparation in progress.

3. Alan V. Oppenheim, and Ronald W. Schafer with John R. Buck, "Discrete-Time Signal Processing" (Second Edition), Prentice Hall (1999)

4. John G. Proakis, and Dimitris G. Manolakis, "Digital Signal Processing - Principles, Algorithms, and Applications" (Third Edition), Prentice Hall (1996)

5. Steven W. Smith, "Digital Signal Processing: A Practical Guide for Engineers and Scientists", Newnes (2003)

6. Sanjit K. Mitra, "Digital Signal Processing - A Computer Based Approach" (Second Edition), McGraw-Hill (2001)

7. Richard G. Lyons, "Understanding Digital Signal Processing", Addison-Wesley (2004)

8. Lonnie C. Ludeman, "Fundamentals of Digital Signal Processing", John Wiley & Sons (1986)

S. EASWARAN
Dr. Easwaran holds BS, MS, and Ph.D. degrees in Electrical Engineering, and is an Assistant Professor in the Computer Sciences and Computer Engineering department at Xavier University of Louisiana. He teaches a senior year Digital Signal Processing course among various other courses at Xavier. His research interests include Digital Signal, Speech, and Image Processing; Digital Communications; and Embedded Systems.