Demonstration for Implementation of Basic Operational Blocks Related to Viterbi Algorithm in Software by Using VB.NET Programming Language

1. Thobius Joseph
Department of Information Technology, Faculty of Sceince and Education, University of Iringa, Tanzania
josephthobius@gmail.com
2. Shangweli Vyosena Kituma
Department of Information Technology, Faculty of Sceince and Education, University of Iringa, Tanzania
Shangwelikituma93@gmail.com
3. Frank Samson
Department of Information Technology, Faculty of Sceince and Education, University of Iringa, Tanzania
Franksam018@gmail.com

Abstract— this study aimed to provide source code and knowledge's on how to implement operational blocks found in ordinary Viterbi algorithm through high level programming languages. In high level programming languages paradigms a VB.NET programming language fall under event programming languages and this study provides easy features for coding Viterbi algorithm operational blocks. The study methodology involved generating randomly bits pattern and mark them on sheet of paper and computing corresponding outputs mathematically by using theory described in literature which clarifying viterbi operational blocks. The trial and error technique were used to migrate the formulas into a block of codes, then the generated bits patterns were passed as inputs to the blocks of codes through textboxes controls to compare the obtained results. The results of this study indicate that same results were obtained by using either mathematical tactic or code implemented by the study.

Index Terms— Convolutional Codes, Software Programming, Trellis Diagram, VB.NET, Viterbi Algorithm

1 INTRODUCTION

1.1 Background

Transmission of data in a communication channel may suffer distortion due to channel impairments and may result to a failure in recovering of such information at receiver([1], [2]). In a typical communication system, a sub system known as channel coding is incorporated to deal with channel impairments which may arises [3]. In a channel coding there are techniques known as error control techniques whose aim is to detect and/or correcting introduced errors by the transmission medium. Forward Error Correction (FEC) is one of approach used in implementing error control by introducing extra bits in a transmitted bits stream for a purpose of detecting and correcting errors automatically at receiver without retransmission of data ([4], [5]).

There are two basic types of FEC which are block codes and convolutional codes [6]. Block codes take a fixed number of input data bits to a complex matrix to generate outputs for transmission and at receiver inverse of such matrix is applied to a received data bit to recover the information's. Convolutional codes work on stream of data bits, it just takes certain length of bits on input data stream and using designed encoder to generate outputs for transmission and at receiver Viterbi Algorithm (VA) is used to recover original information. The VA is an optimal mechanism in removing introduced errors during transmission when binary convolutional codes were employed as FEC technique [8]. VA accomplishes this by comparing the likelihoods of a set of possible state transitions that can occur, and selected one with high possibility of occurence.

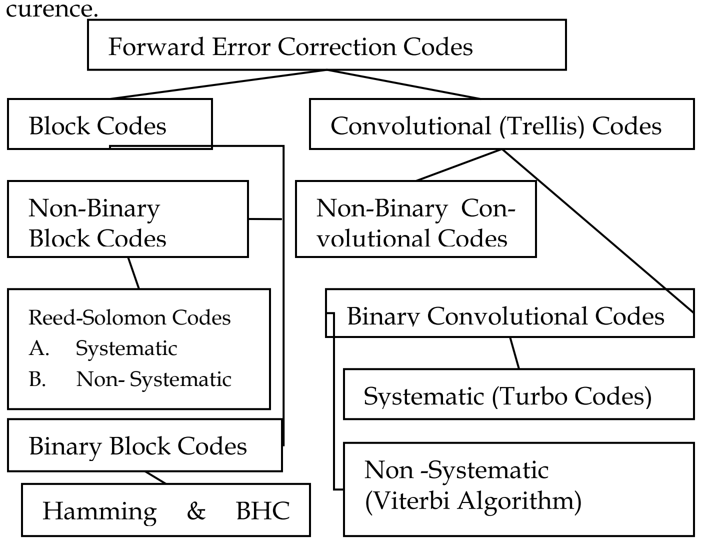
Forward Error Correction Codes
Block Codes
Convolutional (Trellis) Codes
Non-Binary Block Codes
Non-Binary Convolutional Codes
Reed-Solomon Codes
A. Systematic
B. Non- Systematic
Binary Convolutional Codes
Systematic (Turbo Codes)
Binary Block Codes
Non -Systematic (Viterbi Algorithm)
Hamming & BHC

Fig.1. Classifications of FEC codes. The Viterbi Algorithm fall under categories of binary convolutional codes. Convolutional codes also called trellis codes because trellis diagram clearly elaboration of the process [7].

1.2 Viterbi Algorithm

1.2.1 Background of Viterbi Algorithm

An ordinary Viterbi Algorithm (VA) was invented by Viterbi to recover transmitted data at decoder of a wireless communication system when in encoder side a convolutional code was implemented ([8], [33]). In 1955, Elias developed class of FEC known as convolutional codes as an advancement of FEC hamming codes with idea of having infinity constraint length instead of fixed length technique of hamming codes [34]. The constraint of the encoder indicates maximum number of bits upon which the output depends. The memory of the encoder is implemented by using linear shift registers. The convolutional codes has a structure of (n, k, m) meaning it take k input bits' and mapping them into n output bits for transmission by using interconnected registers whose maximum count per single input (m) is given as m = K-1 [33].

In ICT-centric diagrams have being a greater method for describing concepts; this includes use case diagrams, sequence diagrams, class diagrams, tree diagrams, state transition diagrams, trellis diagrams and block diagrams. In channel coding, a block diagram known as generator representation has being mostly used to explain convolutional encoders while trellis diagrams have being used to explain VA decoder [33]. Consider a convolutional encoder of structure (2, 1, 2) with (7, 5) generator polynomials P_0 and P_1 [33], generator representation is given in Fig. 2.

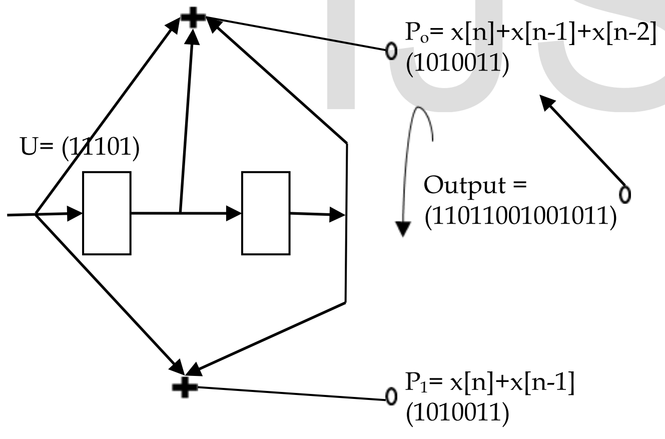

Fig.2. Generator representation of (2, 1, 2) convolutional code with generators P_0 = [111] and P_1 = [101]. The output of encoder depends on connections or taps from encoder memory.

Trellis diagram is graph where finite number of nodes represents states are drawing in vertical axis and repeated along a time axis whereby connections between adjacent axes represent transitions of triggering events [35]. In our case triggering event will be reception of incoming transmitted data. The transitions are denoted such as for binary bits, when event is for bit zero then a solid line is used while when event is for bit one a dashed line is used. The trellis diagram for VA with (2, 1, 2) convolutional encoder is shown in Fig. 4.

1.2.2 Viterbi Algorithm

Akash Thakur *et al.* [30] provide basic architecture of the VA decoder which consists of four main components namely Branch Metric Unit (BMU), Path Metric Unit (PMU), Add Compare and Select Unit (ACSU) and Survivor Management Unit (SMU). Thobius Joseph [33] provides similar list of components involved in VA decoder, whereby SMU was named as traceback block, PMU and ACSU was combined together to form path metric calculation block and BMU remained as branch metric calculation.

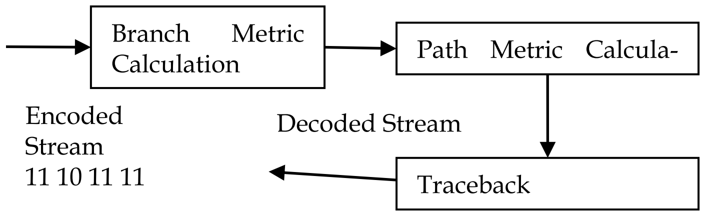

Fig.3. Block Diagram of Ordinary Viterbi Algorithm Decoder

The BMU is a unit for calculation of the metrics between each nodes transition in the trellis diagram when event occurred. The metrics are subject of the design can be Hamming distance or Euclidean distance. Path metric calculation block accumulates the branch metrics per transition between nodes by compare path metrics for each state and store one with either lower or highest value, also its transition is selected as survivor path. Traceback unit first adds tail bits to refresh decoder to a zero state position then, stored path metric are used to recover transmitted data, the process starts from trellis end towards the first stage of the trellis.

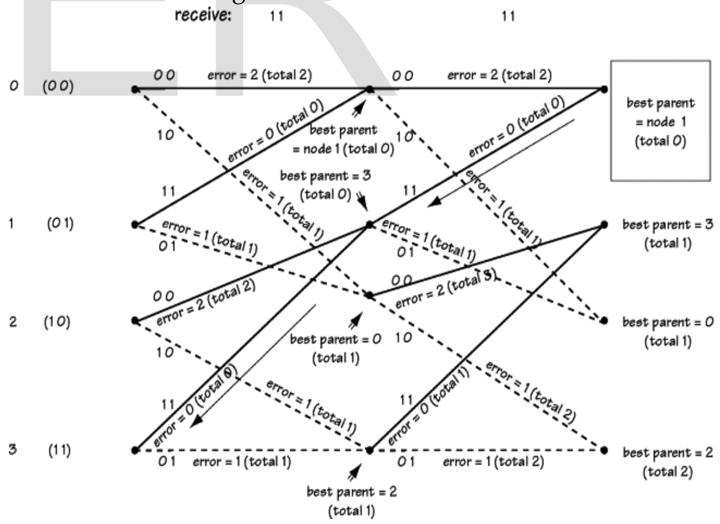

Fig.4. Trellis diagram of VA for corresponding (2, 1, 2) convolutional encoder with generators P_0 = [111] and P_1 = [101]. The diagram shows number of errors when received pattern is 1111 at input of the VA decoder and backtracking to determine the path through the trellis [36].

1.3 Visual Basic Programming Language

Visual Basic (VB) is a language improved from Microsoft Beginners' All-purpose Symbolic Instruction Code (BASIC) language which started in early of 1975 by adding visual forms builder ([19], [20]). VB comes in industry during 1991 with the concept of Component Object Model (COM). In 2002 Microsoft release a VB version under .NET framework, and from it the term VB.NET arises, in order to distinguish classical VB such as VB2 and VB3 from those released from 2002 like VB10 and visual basic 2015 ([21], [22]).

Visual Basic with Networks Enhanced Technology (VB.NET) provides easy programming syntax and it is object –oriented programming language ([9]). The language has been implemented in many studies for creating simulations due to the fact it's a proprietary language for well-known windows based Operating Systems (OS) and therefore, it provides a quickly and easy way of developing more powerful windows programs ([10], [11], [12], [13], [14], [15]). Additional advantageous of VB.NET of other high-level programming languages comes from its features of automatically memory management and intellisense ([16], [17], [18]).

1.4 Problem Statement

Viterbi Algorithm has been employed in many applications includes satellite communication, image recognition techniques, speech recognitions techniques and 5G mobile communications ([23], [24], [25], [26]). This powerful algorithm has been explored by several studied but in hardware level languages such as Very High-speed integrated circuit hardware Description Language (VHDL) for Field Programmable Gate Arrays ([27], [28], [29],[30]) and Very Large-Scale Integration (VLSI) [31]. The exploration in hardware level is complex and increases difficultness for non-electronics scholar to exploit its powerful functionality. Numerous aptitude technologist understanding in easy when concepts are in high level programming languages than complex low-level hardware related programming languages.

In Malaysian VB.NET programming language has become one of compulsory course in computer technology, hence it favored by major of technologists in their country [9]. In addition, this study has conducted a survey for 10 Universities in Tanzania to find preferred Operating System (OS) among scholars and it was observed that about 95% of students own computers are using windows-based OS. In [32] windows OSs are still dominant in the market, hence then, a program developed in VB.NET language would result in efficient and adaptableness can be in large scale.

Therefore, this study comes up with source code and elaborative knowledge's of how to implement Viterbi Algorithm in VB.NET high level programming language. The usage of VB.NET will enable even less experienced programmers to easy understand the source code due to its simple syntax. In addition, when the concepts would become clearly, more input would be provided into the Software- Defined Radio field.

1.5 Paper Organization

The remainder of the paper is organized as follows, Section 2 explain design and implementation of VA in VB.net programming language, also explain study settings. Section 3 provides conclusion and future work, finally section 5 gives acknowledgement.

2 Design and Implementation of Viterbi algorihtm in Visual Basic Programming Language

2.1 Experimental Settings

Visual Studio 2010 Ultimate was used as Integrated Development Environment (IDE). The (2, 1, 2) convolutional encoder from Fig. 2 will be used for elaboration. For convolutional and Viterbi decoder, bits are process serially and continuoulys but in this study 24-bits' pattern were generated randomly and inserted into developed software encoder to generate output. About 256 pattern of 24 bit length were tested, few are given in Table 1. From obtained output some bits were inversed to justifying errors may be introduced due to communication channel impairments. Mathematically for 24 bits input in encoder of ½ code rate the corresponding output will be 48 bits length. Viterbi decoder assumes metric is hamming distance between the received pair of bits and the output bits for each state.

Table 1
Randomly generated 24 Bits' pattern

24 bits' encoder Input	48 bits' encoder Output	X flipped bits denoted by line
0000000000 00100000100	000000000000 000000000000 111011000000 111011111011	000000000000 000000000000 11101$\underline{0}$000000 111011111011
000100100 000100100 000100	000000111011 111011000000 111011111011 000000111011	00$\underline{1}$000111011 111$\underline{1}$11000000 111011111011 000000111011
100100000 100000000 100100	111011111011 000000111011 000000000000 111011111011	111011$\underline{0}$11011 00000011$\underline{0}$011 0000$\underline{1}$0000000 111011011011

The flipped bits act as bit inversion occurred due to medium noises. The patterns were successfully recovered by presented VA code in this paper

2.2 Code for Convolutional Encoder

The textbox control was used as place to enter randomly generated bits pattern, its name TxtInput. The following code should be under button for performing encode task.

```
    '' left most is the first signafance bit, encode start fron
MSB, temp will hold input from TxtInput
Dim temp As String
    Try
      temp = TxtInput.text
   '' Declare codewords C1, C2 and total output c of the en-
coder
      Dim c1, c2, c As String
      For i As Integer = 0 To temp.Length - 3 Step 1
        ' Extract single bit from temp bits array.
        c1 = (Val(temp(i + 2)) Xor Val(temp(i + 1)) Xor Val(temp(i))).ToString ' give C' codeword due to u(n-2) xor u(n-1) xor u(n), n= string.legnth-1 to 0

        c2 = (Val(temp(i + 2)) Xor Val(temp(i))).ToString ''Overall c codeword due to u(n-2) xor u(n)
        c = c1 + c2
      Next
      '' Display the encoded bits patterns
      MessageBox.Show (c.ToString)
    Catch ex As Exception
    '' Display any msg if error occured such textbox takes character O instead of zero 0.
     MessageBox.Show ("Error occured")
    End Try
```

2.3 Code for Branch Metric Calculation

First step is to encode legend trellis diagram for (2, 1, 2) convolultional code in order to hold all possible output for each transition. The legend trellis diagram is shown in Fig. 5.

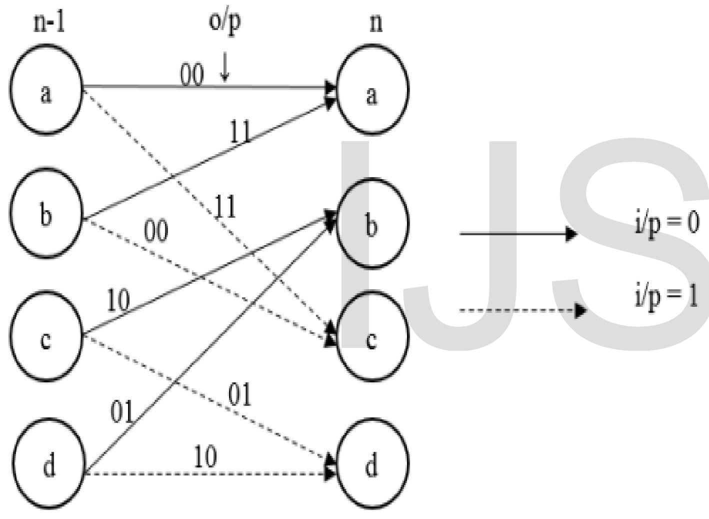

Fig.5. Legend Trellis diagram of VA for corresponding (2, 1, 2) convolutional code

From the diagram there are 8 possible outputs. The outputs will be hold in state array.

```
'Declare the possible output array for each state and ep= temp pm=emission prob= bm + pm, bm = branch metric, pm path metric
   Dim st_relst(0 To 7) As String
     For i As Int16 = 0 To 0
        st_relst(0) = "00"
        st_relst(1) = "11"
        st_relst(2) = "11"
        st_relst(3) = "00"
        st_relst(4) = "10"
        st_relst(5) = "01"
        st_relst(6) = "01"
        st_relst(7) = "10"
     Next
```

Additional two zero bits are added as flush bits, which will make corresponding trellis to have26 transitions. The next step is to generate a trellis matrix with four nodes and 26 transitions.

```
' First apped flush bits into received pattern, here copy what was encoded to a TextBox2
    Dim stri_receiver As String = TextBox2.Text.Trim(" ")
    ' Declare trellis matrix
    ' hold accumalated error metric for each time interval
        Dim PathTable(0 To 3, 0 To 26) As Int16
    ' hold predecessor- successor history for each state
        Dim Pred_Succe_Table(0 To 3, 0 To 26) As Int16
      """ first level decoding, t= 1
        For i As Int16 = 0 To 0
          PathTable(0, 1) = (1 - InStr(st_relst(0)(i), stri_receiver(0)) + (1 - InStr(st_relst(0)(i + 1), stri_receiver(1)))) ' calculate branch metric
          Pred_Succe_Table(0, 1) = 0
        Next
        For i As Int16 = 0 To 0
          PathTable(2, 1) = (1 - InStr(st_relst(1)(i), stri_receiver(0)) + (1 - InStr(st_relst(1)(i + 1), stri_receiver(1)))) ' calculate branch metric
          Pred_Succe_Table(2, 1) = 0
        Next
        """ second level decoding, t = 2
        For i As Int16 = 0 To 0
          PathTable(0, 2) = PathTable(0, 1) + (1 - InStr(st_relst(0)(i), stri_receiver(2)) + (1 - InStr(st_relst(0)(i + 1), stri_receiver(3))))
          Pred_Succe_Table(0, 2) = 0
          ' calculate branch metric
          PathTable(2, 2) = PathTable(0, 1) + (1 - InStr(st_relst(1)(i), stri_receiver(2)) + (1 - InStr(st_relst(1)(i + 1), stri_receiver(3))))
          Pred_Succe_Table(2, 2) = 0
          PathTable(1, 2) = PathTable(2, 1) + (1 - InStr(st_relst(4)(i), stri_receiver(2)) + (1 - InStr(st_relst(4)(i + 1), stri_receiver(3))))
          Pred_Succe_Table(1, 2) = 2
          PathTable(3, 2) = PathTable(2, 1) + (1 - InStr(st_relst(5)(i), stri_receiver(2)) + (1 - InStr(st_relst(5)(i + 1), stri_receiver(3))))
          Pred_Succe_Table(3, 2) = 2

        Next
    ' Show path taken
        MessageBox.Show (PathTable(0, 2).ToString + "  " + PathTable(1, 2).ToString + " " + PathTable(2, 2).ToString + "  " + PathTable(3, 2).ToString)
```

2.4 Code for Path Metric Calculation

The path metric is calculated when trellis is start to repeate itself. The trellis will start to repeate when transition is at third level (t=2).

```
""" enters in the stable state where the graphy is repeating
      Dim temp1, temp2 As Int16
      Dim indexupdate As Int16 = 0
      '' for state 0, each state two transisitions are compared
      For j As Int16 = 2 To 6
      For i As Int16 = 0 To 0
        temp1 = PathTable(0, j) + (1 - InStr(st_relst(0)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(0)(i + 1), stri_receiver(j + 3 + indexupdate))))


        temp2 = PathTable(1, j) + (1 - InStr(st_relst(2)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(2)(i + 1), stri_receiver(j + 3 + indexupdate))))

      Next

      If temp1 > temp2 Then
        PathTable(0, j + 1) = temp2
        Pred_Succe_Table(0, j + 1) = 1
      Else

        PathTable(0, j + 1) = temp1
        Pred_Succe_Table(0, j + 1) = 0
      End If
      MessageBox.Show(PathTable(0, j + 1).ToString) 'Show path taken
      '''' for state 1
      For i As Int16 = 0 To 0
        temp1 = PathTable(2, j) + (1 - InStr(st_relst(4)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(4)(i + 1), stri_receiver(j + 3 + indexupdate))))
        temp2 = PathTable(3, j) + (1 - InStr(st_relst(6)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(6)(i + 1), stri_receiver(j + 3 + indexupdate))))

      Next
      If temp1 > temp2 Then
        PathTable(1, j + 1) = temp2
        Pred_Succe_Table(1, j + 1) = 3
      Else

        PathTable(1, j + 1) = temp1
        Pred_Succe_Table(1, j + 1) = 2
      End If

      '''' for state 2
      For i As Int16 = 0 To 0
        temp1 = PathTable(0, j) + (1 - InStr(st_relst(1)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(1)(i + 1), stri_receiver(j + 3 + indexupdate))))
        temp2 = PathTable(1, j) + (1 - InStr(st_relst(3)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(3)(i + 1), stri_receiver(j + 3 + indexupdate))))
      Next
      If temp1 > temp2 Then
        PathTable(2, j + 1) = temp2
        Pred_Succe_Table(2, j + 1) = 1
      Else

        PathTable(2, j + 1) = temp1
        Pred_Succe_Table(2, j + 1) = 0
      End If
      '' for state 3
      For i As Int16 = 0 To 0
        temp1 = PathTable(2, j) + (1 - InStr(st_relst(6)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(6)(i + 1), stri_receiver(j + 3 + indexupdate))))
        temp2 = PathTable(3, j) + (1 - InStr(st_relst(7)(i), stri_receiver(j + 2 + indexupdate)) + (1 - InStr(st_relst(7)(i + 1), stri_receiver(j + 3 + indexupdate))))
      Next
      If temp1 > temp2 Then
        PathTable(3, j + 1) = temp2
        Pred_Succe_Table(3, j + 1) = 3
      Else

        PathTable(3, j + 1) = temp1
        Pred_Succe_Table(3, j + 1) = 2
      End If
      indexupdate = indexupdate + 1
      ' Show path that was taken
      MessageBox.Show(PathTable(0, j + 1).ToString + " " + PathTable(1, j + 1).ToString + "  " + PathTable(2, j + 1).ToString + " " + PathTable(3, j + 1).ToString)
    Next
    '''''' terminating the trellis by flush bits, only 0 transition are considered here
    For ji As Int16 = 7 To 8
      If ji = 7 Then
        For i As Int16 = 0 To 0
          temp1 = PathTable(0, ji) + (1 - InStr(st_relst(0)(i), stri_receiver(ji + 3)) + (1 - InStr(st_relst(0)(i + 1), stri_receiver(ji + 4))))
          temp2 = PathTable(1, ji) + (1 - InStr(st_relst(2)(i), stri_receiver(ji + 3)) + (1 - InStr(st_relst(2)(i + 1), stri_receiver(ji + 4))))
        Next
        If temp1 > temp2 Then
          PathTable(0, ji + 1) = temp2
          Pred_Succe_Table(0, ji + 1) = 1
        Else

          PathTable(0, ji + 1) = temp1
          Pred_Succe_Table(0, ji + 1) = 0
        End If
        '''' for state 1
        For i As Int16 = 0 To 0
          temp1 = PathTable(2, ji) + (1 - InStr(st_relst(4)(i), stri_receiver(ji + 3)) + (1 - InStr(st_relst(4)(i + 1), stri_receiver(ji + 4))))
          temp2 = PathTable(3, ji) + (1 - InStr(st_relst(5)(i), stri_receiver(ji + 3)) + (1 - InStr(st_relst(5)(i + 1), stri_receiver(ji + 4))))
        Next
        If temp1 > temp2 Then
          PathTable(1, ji + 1) = temp2
          Pred_Succe_Table(1, ji + 1) = 3
        Else
```

```
            PathTable(1, ji + 1) = temp1
            Pred_Succe_Table(1, ji + 1) = 2
          End If
          ' Show path taken
          MessageBox.Show(PathTable(0, j + 1).ToString + " " +
PathTable(1, j + 1).ToString)
        Else
          "" for last state zero only

          For i As Int16 = 0 To 0
            temp1 = PathTable(0, ji) + (1 - InStr(st_relst(0)(i),
stri_receiver(stri_receiver.Length - 2)) + (1 - InStr(st_relst(0)(i +
1), stri_receiver(stri_receiver.Length - 1))))
            temp2 = PathTable(1, ji) + (1 - InStr(st_relst(2)(i),
stri_receiver(stri_receiver.Length - 2)) + (1 - InStr(st_relst(2)(i +
1), stri_receiver(stri_receiver.Length - 1))))

          Next
          If temp1 > temp2 Then
            PathTable(0, ji + 1) = temp2
            Pred_Succe_Table(0, ji + 1) = 1
          Else

            PathTable(0, ji + 1) = temp1
            Pred_Succe_Table(0, ji + 1) = 0
          End If
          ' Show path taken
          MessageBox.Show(PathTable(0, j + 1).ToString)
        End If
      Next
```

2.5 Code for Traceback

At this stage we are end of trellis (t=9) and it required to turn back to t=0 so that we can be able to recover original bits from received codewords.

```
' start of traceback of the code
    Dim tracebackarray(0 To 26) As Int16 ' hold traceback
current state 2 its predecessor state value
    tracebackarray(26) = 0 ' dis position always is zero
    ' MessageBox.Show("trace " + traceback-
array(26).ToString)
    tracebackarray(25) = Pred_Succe_Table(0, 26)
    ' MessageBox.Show("trace " + traceback-
array(25).ToString)
    Dim valuehold As Int16 = tracebackarray(25)
    For cv As Int16 = 25 To 1 Step -1
      tracebackarray(cv - 1) = Pred_Succe_Table(valuehold,
cv)
      valuehold = tracebackarray(cv - 1)
      ' MessageBox.Show("trace " + valuehold.ToString)
    Next
    stri_receiver = ""
    Dim err As String = ""
    Dim jj As Int16 = 1
    Dim V() As Int16 = {0, 1, 2, 3, 4}

    For el As Int16 = 0 To tracebackarray.Length - 2
      ' MessageBox.Show(indexarray(jj).ToString)
      If (tracebackarray(el) = V(0) Or tracebackarray(el) =
V(1)) And (tracebackarray(jj) = V(0)) Then
        stri_receiver = stri_receiver + "0"
      ElseIf (tracebackarray(el) = V(0) Or tracebackarray(el) =
V(1)) And (tracebackarray(jj) = V(2)) Then
        stri_receiver = stri_receiver + "1"
      ElseIf (tracebackarray(el) = V(2) Or tracebackarray(el) =
V(3)) And (tracebackarray(jj) = V(1)) Then
        stri_receiver = stri_receiver + "0"
      ElseIf (tracebackarray(el) = V(2) Or tracebackarray(el) =
V(3)) And (tracebackarray(jj) = V(3)) Then
        stri_receiver = stri_receiver + "1"
      Else
        err = err + "x" '' count number of errors in VA algo-
rithm
        stri_receiver = stri_receiver + "0"

        V(4) = 0
      End If
      jj = jj + 1
    Next

    If V(4) = V(0) Then '' it will be show number of error oc-
cured
      MessageBox.Show ("Amount of error occured is " + err)
    End If
' Display recovered bits in Txtrecoverd
    Txtrecovered.Text = stri_receiver, Remove(24, 2)
```

Thobius Joseph own BSc. In Telecommunications Engineering (2014) and MSc. In Telecommunications Engineering (2016) from Unicersity of Dodoma, Tanzania. E-mail: josephthobius@gmail.com

3 Conlusion and Future work

In this paper, source codes for implementing convolutional encoder and Viterbi decoder in VB. Net were developed, tested and successfully reproduce the desire results. To get overall function code for VA decoder the code for BMU, PMU and traceback should be combined together. The conversion of source code from VB.NET to other high-level language like java is easy as there are many free online converters.

The future work will be enabling dynamic constraint length and developing open program which could increase community skills and expand software defined radio field.

Acknowledgment

The authors wish to thank Vijana Labs. Co. limited and University of Iringa.

References

[1] I. B. Djordjevic, *Advanced Optical and Wireless Communications Systems*, Springer, Cham, 2018 https://doi.org/10.1007/978-3-319-63151-6_6.

[2] S. Matin and L. Milstein, "Impacts of Channel Impairments and Imperfections on OFDM System Performances," *2018 IEEE Global Con-*

ference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 2018, pp. 1209-1213, doi: 10.1109/GlobalSIP.2018.8646520.

[3] Ouyang, Feng, *Digital Communication For Practicing Engineers*, The Institute of Electrical and Electronics Engineers, Inc, pp.163-240, 2019.

[4] A. Mohr, E. Riskin, R. Ladner, "Unequal loss protection: Graceful degradation over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Communication*, vol. 18, pp.819–828, 2000.

[5] Pradeep Kumar Shaga et al, "A New Approach of Forward Error Correction For Packet Loss Recovery," *International Journal of Computer Science and Mobile Computing*, Vol.3 Issue.9, pg. 670-674, 2014.

[6] UKEssays, "Types Of Forward Error Correction," https://www.ukessays.com/essays/information-technology/types-of-forward-error-correction-information-technology-essay.php?vref=1, 2018.

[7] Atlanta RF, "Link Budget Analysis: Error Control & Detection",unpublished, 2013.

[8] Akkidas, Davidrichesemmanuel, "Viterbi Algorithm," *Journal of Experimental Algorithmics*, 2015.

[9] Chowdhury, Dwaipayan, "Design Automation Methodology of a Cube using Solidworks and VB.net," *International Journal of Engineering Research*, 2020, V9. 10.17577/IJERTV9IS060352.

[10] H. Lauren, "Programmer's toolchest: examining VB.NET," *Dr. Dobb's Journal*, 27. 65-69, 2002.

[11] N. Dharshinni, S. Amir, A. Fadhillah, and I. Fawwaz, "Design of Simulation Definite Integral Application learning Using Trapezoid Method based on VB.Net," *Journal of informatics and telecommunication engineering*, 4. 193-202, 2020, 10.31289/jite.v4i1.3880.

[12] H. John, H. Elizabeth, T. Anne, J. Scott, and B. Moe, "Using Assignments in a VB.Net Class to Create Simulations for Use by Teachers," *Conference: Society for Information Technology & Teacher Education International Conference*, 2010.

[13] H. Wen-Qing and Z. Jie-min., "Mixed-language programming with Fortran and VB.NET," *Zhongshan Daxue Xuebao/Acta Scientiarum Natralium Universitatis Sunyatseni*, 56. 1-8, 2017, 10.13471/j.cnki.acta.snus.2017.04.001

[14] Prastianto, Fayyadh & Rostiani, Yeny, "Komputerisasi Akuntansi Penyusutan Aktiva Tetap Metode Garis Lurus Berbasis Vb.Net Pada Pt Alam Makmur Karawang," *Jurnal Interkom*, 2020, 15. 10.35969/interkom.v15i1.68.

[15] Bakhtiar Md Shaari1 and Wan Roslina Wan Musa, "Visual Basic Program With Hardware: Teaching Of Interfacing Method Approach For Serial Communication,"*Advanced Journal of Technical and Vocational Education* 1 (1): 97-106, 2017.

[16] Appleman, Dan, *Moving to VB.NET: Strategies, Concepts, and Code* , pp.103-121, 2001.

[17] K. Tahani, *Visual programming in VB.Net*, Dar wael, 2014, ISBN: 978-9957-91-128-7.

[18] Whiteside, Mary & Eakin, Mark,"VB Simulation for Multiple Comparisons Research,"2020.

[19] Wikipedia, "BASIC", https://en.wikipedia.org/wiki/BASIC, 2020.

[20] Haney, John & Lovely, John. .NET as a Teaching Tool. 1., 2016

[21] T. Stephen, *Subclassing and hooking with Visual Basic - harnessing the full power of VB/VB.NET: covers VB.NET*, O'Reilly, 978-0-596-00118-6, 2001.

[22] S. Devi, S. P. Ramalingam, and P.V.S.S.R., Chandra Mouli, "Challenges and Issues in Code Migration From VB 6.0 TO VB.NET," *International Journal of Computer Information Systems*, 2011.

[23] Liu, Kaiming and Peng, Zhengbo, "Service Function Chain Deployment Based on Improved Viterbi Algorithm in 5G-C-RAN," *Conference: 2019 IEEE 5th International Conference on Computer and Communications (ICCC) Project: Large Vocabulary Indonesian Automatic Speech Recognition*,1030-1035, 2019, 10.1109/ICCC47050.2019.9064366.

[24] Hatala, Zulkarnaen, "Viterbi Algorithm and its application to Indonesian Speech Speech Recognition," *Conference: 3rd International Conference on Statistics, Mathematics, Teaching, and Research (ICSMTR 2019) Makassar*, 9-10 Oct 2019.

[25] V. Fabian, F. Rubem and J. Daniel, "A FPGA-based Viterbi algorithm implementation for speech recognition systems,"*Conference: Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01)*. 2. 1217 – 1220, 2001,10.1109/ICASSP.2001.941143.

[26] Q. Xie, Q. Long, and S. Mita, "Integration of optical flow and Multi-Path-Viterbi algorithm for stereo vision," *International Journal of Wavelets, Multiresolution and Information Processing*, 2017, 15. 10.1142/S0219691317500229.

[27] V. Kakkara, K. Balasubramanian, B. Yamuna, D. Mishra, K. Lingasubramanian and S. Murugan, "A Viterbi decoder and its hardware Trojan models: an FPGA-based implementation study," *PeerJ Comput. Sci.* 6:e250, 2020, DOI 10.7717/peerj-cs.250.

[28] S. Hema, V. Babu, and R. Pushpangadan, "FPGA implementation of Viterbi decoder," *Proceedings of the 6th WSEAS Int. Conf. on Electronics, Hardware, Wireless and Optical Communications*, Corfu Island, Greece, 162-167, February 16-19, 2007.

[29] M. Mozaffari Kermani, V. Singh and R. Azarderakhsh, "Reliable Low-Latency Viterbi Algorithm Architectures Benchmarked on ASIC and FPGA," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 1, pp. 208-216, Jan. 2017, doi: 10.1109/TCSI.2016.2610187.

[30] Akash Thakur and Manju K Chattopadhyay," Design and Implementation of Viterbi Decoder Using VHDL," *3rd International Conference on Communication Systems (ICCS-2017) IOP Publishing IOP Conf. Series: Materials Science and Engineering 331*, 2018, doi:10.1088/1757-899X/331/1/012009.

[31] R. Putra, and T. Adiono, "VLSI Architecture for Configurable and Low-Complexity Design of Hard-Decision Viterbi Decoding Algorithm," *Journal Of ICT Research And Applications*, 10(1), 57-75, 2016.

[32] Patayon, Urbano and Mingoc, Nerico, "Operating Systems Usability: A Comparative Study," *JPAIR Multidisciplinary Research*, 2019, 36. 10.7719/jpair.v36i1.683.

[33] Thobius Joseph, "Prototype for Multimedia Content Delivery Based on Ntc Enhanced Viterbi Algorithm", MSc. In Telecommunications Eng., The University of Dodoma, Dodoma, Tanzania, 2016.

[34] James L. Massey, "OBITUARY Peter Elias, 1923–2001", IEEE Information Theory Society Newsletter, Vol. 52, No. 1 available at https://www.itsoc.org/publications/newsletters/itNL0302.pdf, 2001.

[35] M. Mona, A. Hossam, A. Fathi and E. Ayman, "Image Security With Different Techniques Of Cryptography And Coding: A Survey," *IOSR Journal of Computer Engineering*, 16. 39-45, 2014, 10.9790/0661-16313945.

[36] Carl Nassar, "Channel Coding and Decoding: Convolutional Coding and Decoding," *Telecommunications Demystified*, Carl Nassar, eds., Newnes, Pages 197-219, 2001, ISBN 9780080518671, https://doi.org/10.1016/B978-0-08-051867-1.50013-5