VISUALISASI KINERJA PENGKODEAN MENGGUNAKAN ALGORITMA VITERBI

Aslam mahyadi[1], Arifin,MT
[1]Politeknik Elektronika Negeri Surabaya, Jurusan Teknik Telekomunikasi
Kampus ITS, Surabaya 60111
e-mail : meaninglife@yahoo.com

ABSTRAK

Dalam perkembangan teknologi informasi dan telekomunikasi yang begitu pesat khususnya pada sistem komunikasi, proses pertukaran data yang terbagi pada dua sisi utama yaitu pada sisi pengirim dan pada sisi penerima, masing masing memiliki mekanisme kerja tersendiri dan memiliki ruang lingkup yang sangat luas. Sangat tidak mungkin untuk mengetahui dan memahami seluruh mekanisme yang ada tanpa mengetahui setidaknya salah satu bagian dari mekanisme tersebut atau prinsip kerjanya, untuk itulah pada proyek akhir ini dibuat suatu sistem perangkat lunak untuk sebuah visualisasi kinerja pengkodean menggunakan algoritma viterbi.

Prinsip kerja dari algoritma viterbi adalah mengkodekan kembali bit - bit data yang telah dikodekan oleh convolutional code dengan prinsip *maximum likelihood* atau mencari kemungkinan bit bit yang paling mirip antara bit yang diterima pada sisi decoder dengan kemungkinan bit - bit yang muncul sesuai hasil proses pada convolutional code pada tiap - tiap percabangan trellis.

Dengan mengetahui bagaimana kinerja algoritma viterbi kita dapat mengetahui setidaknya satu mekanisme pembentuk suatu sistem komunikasi yang komplek, sehingga dapat menjadi media pembelajaran yang baik dalam sistem komunikasi.

Kata kunci : Algoritma Viterbi, Convolutinal Code, Pemrograman C

1. Pendahuluan

Dalam sstem komunikasi proses pertukaran data terbagi pada dua sisi utama yaitu pada sisi pengirim dan pada sisi penerima yang masing masing memiliki mekanisme kerja tersendiri. Dalam perkembangan teknologi informasi dan telekomunikasi yang begitu pesat banyak metode yang tersedia yang membentuk mekanisme pada sisi pengirim dan penerima yang masing masing metode memiliki ruang lingkup tersendiri yang digunakan sesuai ruang linkupnya. Pada pertukaran data digital khususnya banyak sekali metode pada sisi penerima dan pengirim yang mendukung yang dalam istilahnya disebut metode encoding dan decoding. Untuk mengetahui bagaimana mekanisme pertukaran data maka mekanisme dari salah satu metode baik encoding maupun decoding harus diketahui pula. Viterbi decoding algoritma pada sisi penerima sebagai metode decoding dan binary convolutional coding pada sisi pengirim dipilih sebagai metode yang akan diuji sehingga dapat diketahui bagaimana mekanisme suatu sistem pertukaran data khususnya pada sisi penerima.

2. Teori Penunjang

2.1 Bahasa C dan OpenGl

2.1.1 Sejarah dan Perkembangan C

Bahasa C pertama kali disusun oleh Dennis Ritchie pada tahun 1972 di laboratorium AT&T BELL, suatu laboratorium tempat merancang sistem operasi UNIX.

Sejarah perkembangan C dimulai pada saat Ken Thompson membuat bahasa B untuk menyusun sistem operasi UNIX. Kemudian bahasa B dikembangkan menjadi bahasa C dengan penambahan beberapa tipe data dan sintaksis baru.

Dalam perkembangan metode atau cara menyususn suatu program, dari teknik pemrograman terstruktur berkembang menjadi apa yang disebut pemrograman berorientasi objek (*object oriented programming* atau *OOP*).

Pada tahun 1986 Bjarne Stroustrup dari laboratorium AT&T BELL, mengembangkan bahasa C menjadi bahasa C++ yang dapat digunakan untuk pemrograman berorientasi object. Fasilitas tambahan yang diberikan pada bahasa C++ adalah tipe data class.

.2.1.2 OpenGl

OpenGL adalah sebuah software antarmuka pada hardware grafik. *OpenGL* tersusun atas lebih dari 700 *command* yang berbeda-beda (sekitar 670 *command* atau perintah yang dispesifikasikan untuk OpenGL versi 3.0 dan sisanya 50 dalam *OpenGL Utility Library*) yang akan digunakan untuk menspesifikasi atau menetapkan obyek dan operasi yang dibutuhkan untuk menghasilkan aplikasi dua atau tiga demensi yang interaktif.

OpenGl yang kta gunakan menggunakan tiga set grafik library utama yaitu :

- **Core OpenGL (GL)**
- **OpenGL Utility Library (GLU)**
- **OpenGL Utility Toolkit (GLUT)**

2.2 Convolutional Code

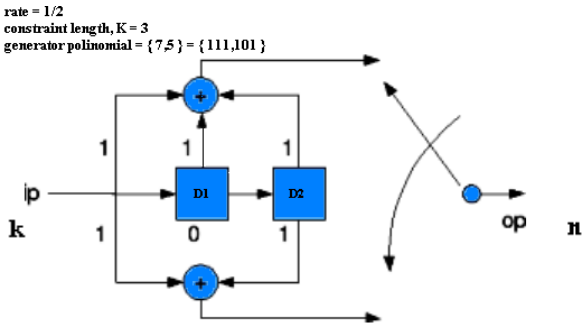
Convolutional code secara garis besar dapat didefinisikan sebagai sebuah sistem kode yang memiliki informasi informasi bit masukan yang akan dikodekan menjadi beberapa bit terkode dengan syarat informasi bit masukan harus lebih kecil daripada keluaran bit terkode dan yang paling penting convolutional code memiliki memori yang menyebabkan terciptanya sebuah aturan untuk mengkodekan setiap informasi bit masukan menjadi beberapa bit terkode berdasarkan bit bit informasi masukan sebelumnya. Untuk lebih ringkasnya convolutional code dapat disebut kode dengan struktur kode (n, k, m).

n = jumlah keluaran bit

k = jumlah masukan bit **(2.1)**

m = jumlah memori (jumlah shift register)

Tetapi di lain sisi convolutional code juga dapat didefinisikan oleh beberapa generator polinomial, ini dikarenakan setiap elemen pada generator polinomial mendefinisikan tiga tiang utama dari convolutional code yaitu jumlah shift register (jumlah memori) atau panjang kode convolutional code (constraint length), jumlah keluaran bit (jumlah modulo-2 adder) dan hubungan koneksi antara shift register dan modulo-2 adder dan ini tampak jelas pada gambar rangkaian di bawah ini.



Gambar 2.1 Rate ½ convolutional encoder

2.2.1 Convolutional Code Parameter

Berdasarkan pendefinisian bahwa convolutional code adalah kode (n, k, m) kita dapat memperoleh parameter parameter utama yang menjadi pilar utama pembangun convolutional code yaitu :

1.Rate
2.Constraint Length
3.generator polinomial

Dimana :

Rate

Rate merupakan ratio antara masukan informasi bit dengan keluaran bit terkode dan mempunyai persamaan sebagai berikut :

$$R = k / n \qquad (2.2)$$

keterangan :

R = laju kode konvolusi
k = jumlah bit masukan convolutional code
n = jumlah bit keluaran convolutional code

Constraint Length

Constraint length adalah jumlah delay elemen dalam convolutional code yaitu memori dengan masukan bit sekarang pada convolutional code atau dapat disebut juga panjang kode dari convolutional code. Constraint length dapat didefinisikan sebagai berikut :

$$K = m + 1 \qquad (2.3)$$

keterangan :

K = constraint length
m = memori

Generator Polinomial

Generator polinomial sangat dibutuhkan untuk merangkai suatu convolutional code berdasarkan jumlah memori yang digunakan dalam suatu convolutional code selain itu setiap elemen pada generator polinomial serta jumlah dari fungsi generator polinomial mempengaruhi :

1. Jumlah output (jumlah modulo-2 adder)
2. Panjang convolutional code (jumlah shift register + input)
3. hubungan koneksi antara shift register dan modulo-2 adder

Berikut adalah tabel untuk generator polinomial yang digunakan untuk convolutional code yang baik dengan rate ½.

Tabel 2.2 Best rate ½ convolutional code

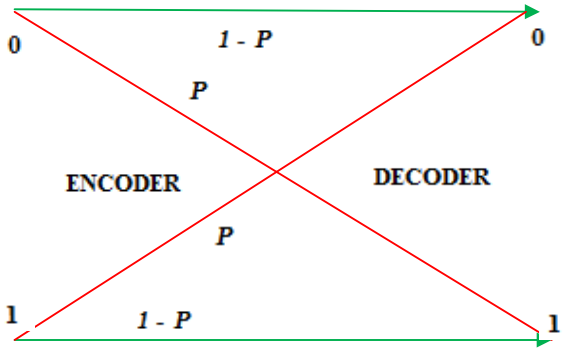
Constraint Length, K	Code Generator polynomials (Octal)
3	(7,5)
4	(17, 15)
5	(35, 23)
6	(75, 53)
7	(171, 133)
8	(371, 247)
9	(753, 561)

2.3 Kanal Transmisi

Kanal transmisi merupakan media yang memiliki sejumlah efek pada proses pentransmisian yaitu seperti attenuasi, distorsi, interferensi dan noise yang membuat keadaan tak tentu pada bit bit data akan diterima dengan baik atau tidak di sisi receiver. Dalam kasus ini yang kita pakai adalah **Binary Symmetric Channel**.

2.3.1 Binary Symmetric Channel

A binary symmetric channel (BSC) adalah sebuah kanal komunikasi yang memiliki masukan dan keluaran berupa bit biner dengan probabilitas error saling silang serta hanya dapat memproses satu dari dua bit simbol keluaran encoder. Dalam model gambar 2.2 diharapkan sebuah encoder untuk mengirim sebuah bit (nol atau satu) dan pada sisi decoder menerima sebuat bit yang sama pula. Diasumsikan bahwa secara umum bit yang ditransmisikan setelah mengalami proses pengkodean pada encoder akan dibalik nilainya dengan nilai probabilitas yang kecil (probabilitas saling silang).



Gambar 2.2 Binary Symmetric Channel

Sebuah binary symmetric channel dengan probabilitas saling silang **P** adalah sebuah kanal dengan masukan dan keluaran dalam bentuk biner serta probabilitas error yang mana jika pada sisi **encoder** dimisalkan **X** adalah bit keluaran encoder dan **Y** dimisalkan sisi pada decoder adalah bit keluaran dari kanal, maka kanal dapat dikondisikan dengan kondisional probabilitas sebagai berikut :

$$\begin{aligned} &Pr(Y = 0 \mid X = 0) = 1 - P \\ &Pr(Y = 0 \mid X = 1) = P \\ &Pr(Y = 1 \mid X = 0) = P \\ &Pr(Y = 1 \mid X = 1) = 1 - P \end{aligned} \qquad (2.4)$$

Diasumsikan range probabilitas adalah $0 \leq P \leq 1/2$. Jika $P > 1/2$ maka proses reverse bit kemungkinannya akan semakin besar dibandingkan asumsi awal probabilitas yang telah ditentukan di atas.

2.4 Viterbi Algoritma

Viterbi algoritma adalah metode decoding untuk mengkodekan kembali bit yang telah dikodekan oleh convolutional code dengan prinsip mencari kemungkinan bit yang paling mirip atau dapat disebut maximum likelihood. Proses decoding dapat disamakan dengan membandingkan deretan bit yang diterima dengan semua kemungkinan bit terkode, dari proses perbandingan tersebut akan dipilih bit yang paling mirip antara deretan bit yang diterima dengan kemungkinan deretan bit bit yang ada. Untuk menentukan deretan bit yang paling mirip adalah dengan menghitung nilai hamming distancenya dan dari setiap nilai hamming distance untuk tiap index kedalaman trellis akan dibandingkan dan diakumulasikan sehingga nilai akumulasi yang terkecil akan dipilih dan akan menjadi dasar dari survivor path selanjutnya.

Viterbi algoritma dapat diwakili dengan sebuah trellis diagram, dalam kasus ini metode yang kita gunakan dalam viterbi algoritma adalah hard decision decoding yaitu dengan menggunakan ukuran metric yaitu hamming distance.

2.4.1 Hard Decision Decoding

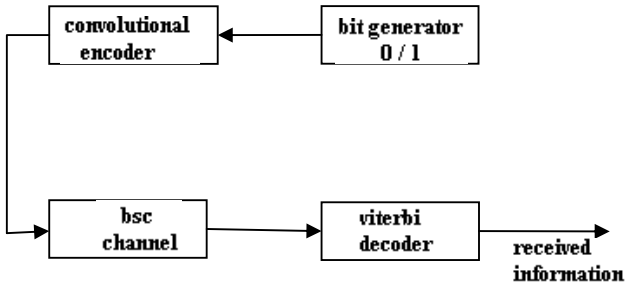
Path path yang tidak terputus merupakan survivor path yang berguna untuk menentukan decoding path dalam trellis diagram. Dalam metode viterbi algoritma hard decision decoding, prinsip maximum likelihood atau mencari kemungkinan bit yang mirip adalah mencari path dengan nilai hamming distance yang paling kecil (akumulasi terkecil nilai hamming distance) yang akan menjadi kandidat penerus survivor path

sebelumnya dalam proses decoding di dalam trellis diagram.

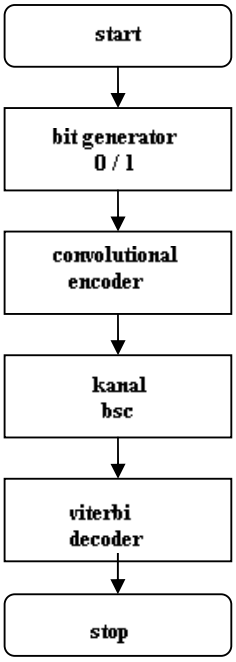
Berikut adalah contoh proses decoding viterbi algoritma menggunakan metode Hard Decision Decoding yang terlihat pada hasil.

3 Perencanaan Sistem

Untuk mengetahui step step pembuatan sistem keseluruhan setidaknya membutuhkan perencanaan dalam hal ini adalah desain sistem dan alur flowchart utama.



Gambar 3.1 Desain Sistem Utama

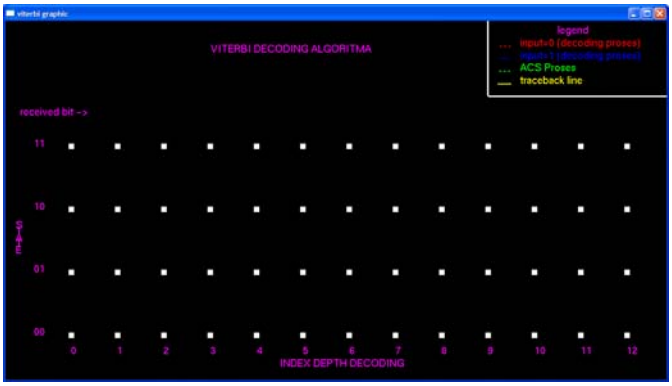


Gambar 3.2 Flowchart Sistem Utama

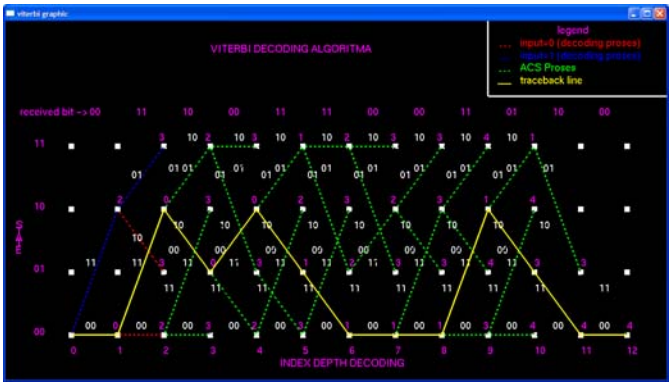
Keterangan Flowchart :

1. Start untuk memulai suatu program pada perangkat lunak.
2. Proses pembangkitan data bit.
3. Data bit yang telah dibangkkitkan dikodekan dengan prinsip convolutional code dengan struktur yang ditentukan.
4. Bit data terkode hasil proses convolutional code disimulasikan untuk mengalami distorsi.
5. Bit bit terkode keluaran kanal akan mengalami proses decoding dengan menggunakan algoritma viterbi.
6. Stop proses akhir program.

4 Hasil



Gambar 4.1 Trellis Kosong



Gambar 4.2 Proses Akhir Decoding

5 Kesimpulan

Bahwa setelah melalui proses uji dan analisa dapat disimpulkan :

1. Dengan metode eliminasi salah satu dari dua percabangan yang menuju setiap state dari *convolutional encoder* yang memiliki hamming distance yang sama, hanya ½ dari jumlah keseluruhan percabangan trellis pada setiap kedalaman decoding tertentu yang lolos seleksi, yaitu pada kedalaman decoding ke 3 sampai kedalaman decoding akhir yaitu ke 12 .
2. Bahwa proses decoding memiliki dua mekanisme berbeda untuk viterbi algoritma hard decision yaitu pada saat decoding proses dari kedalaman trellis 0 sampai 2 dan

decoding proses kedalaman 3 sampai seterusnya.

3. Bahwa kunci efisiensi dari algoritma viterbi adalah mengeliminasi salah satu dari dua percabangan trellis state pendahulu yang menuju setiap state pada kedalaman decoding berikutnya.
4. Struktur dari *convolutional encoder* sangat berpengaruh dalam menentukan proses dan pola pengkodean kembali oleh algoritma viterbi.

Daftar Pustaka

[1] Chip Fleming, A Tutorial on Convolutional Coding with Viterbi Decoding, 1999-2002.

[2] P.J.Deitel, H.M.Deitel, C How To Program Fifth Edition, Pearson International Edition.

[3] Teguh Wicakulah, Pembuatan Software Sistem Pengiriman dan Penerimaan Informasi Menggunakan Teknik Konvolusi, PENS-ITS, Surabaya, 2007.

[4] Rodger E.Ziemer, Roger L.Paterson, INTERNATIONAL EDITION INTRODUCTION TO DIGITAL COMMUNICATION SECOND EDITION, PRENTICE HALL INTERNATIONAL, INC.

[5] Dave Shreiner, OpenGL Programming Guide Seventh Edition The Official Guide to Learning OpenGL Versions 3.0 and 3.1, Addison-Wesley, 2010.

[6] http://www.dsplog.com/2009/01/04/viterbi/

[7] http://www.dsplog.com/2009/01/04/convolutional-code/

[8] http://www.codeproject.com/Articles/67933/OpenGL-String- printw.aspx

[9] http://www.dsplog.com/2009/08/21/matlab-or-c-for-viterbi-decoder/

[10] http://publications.gbdirect.co.uk/c_book/

[11] http://en.wikipedia.org/wiki/Binary_symmetric_channel

[12] http://www.complextoreal.com/chapters/convo.pdf