Penerapan Kecerdasan Buatan Dalam Menyelesaikan Permainan Pergeseran Angka Pada Bintang David Dengan Metode Pencarian Breadth-First Dan Pencarian Heuristic Menggunakan Bahasa Pemrograman Visual

Diana Astria Gultom
Program Studi Teknik Informatika
Fakultas Ilmu Komputer, Universitas Pelita Harapan
E-mail: diana.gultom@gmail.com

ABSTRACT

This study aims to developt software enginerring applications and Artificial Intelligent play game on computer. Software Engineering had a game move number in the model david star. Where the user before not be find solution in this game because a long time or disability to solve this game. This research was done by design, and give depth to implement method breadth first search (unblind search) and heuristic search (search with gathering information). The blind search are search will not used information to find trace, therefore BFS maybe not a step by leave but limited resource memory made BFS solution incompleted. To Heuristic Search Method, search do with a make stock or information resource. After information made to BFS, then heuristic will direct guide to final state, therefore solution will incomplety if gived depth information limited. The results show software made to function properly and it can be to solve solution or goal and can be expanded to a larger scale and can it be sell to all people using PC and like this brain game.

Keywords*: artificial intelligent, software enginerring, BFS*

ABSTRAK

Penelitian ini bertujuan untuk mengembangkan aplikasi kecerdasan buatan pada perangkat lunak. Perangkat lunak berupa permainan pergeseran angka dengan model bintang David, dimana para pemain sebelumnya tidak menemukan solusi dalam permainan tersebut karena masalah waktu ataupun ketidakmampuan pemain tersebut. Penelitian ini dilakukan dengan merancang, membuat dan mengimplementasikan metode pencarian buta (BFS) dan pencarian berbekal informasi (heuristic). Pencarian buta merupakan pencarian yang tidak menggunakan informasi dalam mencari jejak, walaupun BFS tidak mungkin ada langkah yang tertinggal tetapi keterbatasan memori membuat solusi BFS tidak selalu terpenuhi. Pada metode pencarian heuristik, pencarian dilakukan dengan cara membuat bekal atau informasi, setelah informasi dibuat dengan metode BFS, maka heuristik akan langsung membimbing ke tujuan akhir, akan tetapi solusi tidak dapat terpenuhi kalau kedalaman informasi yang diberikan terbatas. Hasil penelitian menunjukkan perangkat lunak yang dibuat dapat berfungsi dengan baik dan dapat menemukan solusi atau tujuan serta dapat dikembangkan untuk skala yang lebih besar dan lebih berguna baik masyarakat pengguna komputer sekaligus menyukai permainan perangkat lunak tersebut.

Kata Kunci: kecerdasan buatan, perangkat lunak, BFS

PENDAHULUAN

Perkembangan kecerdasan buatan atau diistilahkan Artificial Intelligence sangat pesat, sehingga masalah dapat diselesaikan secepat mungkin dan dapat diterapkan dalam berbagai jenis permainan angka. Keunikan dari permainan angka ini menjadi sangat mengasyikkan, dan sekaligus dapat digunakan untuk melatih kecerdasan. Permainan pergeseran angka biasanya dimainkan dalam bentuk segi empat. Jenis permainan ini cenderung lebih mudah untuk dimainkan dan diselesaikan. Permainan ini akan menjadi jauh lebih rumit dan susah apabila dimainkan dalam model yang berbentuk bintang. Bentuk model ini menyebabkan arah proses pergeseran angka menjadi terbatas.

Permainan pergeseran angka dalam bintang David ini dapat diselesaikan dengan menggunakan bantuan struktur pohon pencarian (search tree). Pohon pencarian adalah suatu pohon (tree), dimana akar dari pohon berupa keadaan awal dan cabang berupa keadaan-keadaan yang mungkin terjadi dari keadaan sebelumnya serta daun merupakan keadaan akhir, yang dapat dijadikan sebagai solusi dari permasalahan. Namun, tidak semua daun dapat dijadikan sebagai solusi, dalam beberapa contoh kasus, ada beberapa atau semua daun bukan merupakan solusi dari permasalahan.

Permainan pergeseran angka dalam bintang David ini membutuhkan waktu yang lama untuk diselesaikan secara manual. Oleh karena itu, penulis berusaha untuk merancang sebuah program yang dapat memberikan solusi bagi permainan ini dengan menggunakan bantuan pohon pencarian.

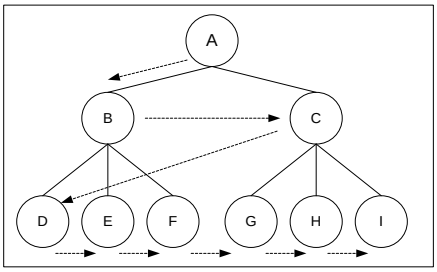
Kecerdasan Buatan

Kecerdasan Buatan (bahasa Inggris: Artificial Intelligence atau AI) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (games),logika kabur, jaringan syaraf tiruan dan robotika.

Tujuan dari AI adalah untuk memecahkan persoalan dunia nyata (bersifat praktis) dan memahami intelijensia (bersifat memahami). AI merupakan salah satu bagian ilmu komputer yang mempelajari tentang bagaimana cara membuat agar komputer dapat melakukan pekerjaan seperti yang dilakukan oleh manusia. Pada awal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan zaman, maka peran komputer semakin mendominasi kehidupan umat manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, lebih dari itu, komputer diharapkan untuk dapat diberdayakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia.

Pencarian Melebar Pertama (Breadth First Search)

Pada metode pencarian ini, semua *node* pada level n akan dikunjungi terlebih dahulu sebelum mengunjungi *node-node* pada level n+1. Pencarian dimulai dari *node* akar terus ke level ke-1 dari kiri ke kanan, kemudian berpindah ke level berikutnya.



Gambar 1. Pencarian melebar pertama (*Breadth First Search*)

Karena proses *breadth first search* mengamati setiap *node* di setiap level graf sebelum bergerak menuju ruang yang lebih dalam, maka mula-mula semua keadaan akan dicapai lewat lintasan yang terpendek dari keadaan awal. Oleh sebab itu, proses ini menjamin ditemukannya lintasan terpendek dari keadaan awal ke keadaan tujuan. Lebih jauh karena mula-mula semua keadaan ditemukan melalui lintasan terpendek sehingga setiap keadaan yang ditemui pada kali kedua didapati pada sepanjang sebuah lintasan yang sama atau lebih panjang. Kemudian, jika tidak ada kesempatan ditemukannya keadaan yang identik pada sepanjang lintasan yang lebih baik maka algoritma akan menghapusnya, sehingga keuntungan dari metode pencarian ini adalah:

1. Tidak akan menemui jalan buntu.
2. Jika ada satu solusi, maka *breadth-first search* akan menemukannya. Dan jika ada lebih dari satu solusi, maka solusi minimum akan ditemukan.

Namun demikian, ada tiga persoalan utama berkenaan dengan metode pencarian ini, yaitu:

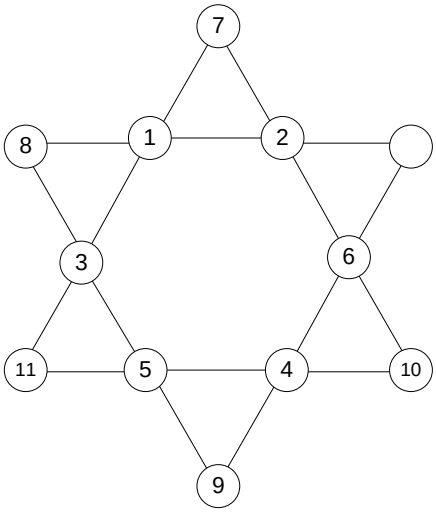
1. Membutuhkan memori yang besar, karena menyimpan semua *node* dalam satu pohon. Jumlah *node* di setiap tingkat dari pohon bertambah secara eksponensial terhadap jumlah tingkat, dan semuanya ini harus disimpan sekaligus.
2. Membutuhkan sejumlah besar pekerjaan, khususnya jika lintasan solusi terpendek cukup panjang, karena jumlah *node* yang perlu diperiksa bertambah secara eksponensial terhadap panjang lintasan.
3. Tidak relevannya operator akan menambah jumlah *node* yang harus diperiksa sangat besar.
4. Relatif membutuhkan waktu yang cukup lama, karena akan menguji semua *node* pada level ke-*n* untuk mendapatkan solusi pada level ke- (*n* + 1). **[(Anita Desiani & Muhammad Arhami) (Konsep Kecerdasan Buatan)]**

Pergeseran Angka pada Bintang David

Permainan pergeseran angka pada bintang David mirip dengan permainan pergeseran angka dalam kotak berbentuk persegi. Perbedaannya adalah bentuk wadah yang digunakan dalam permainan ini berbentuk bintang David. Setiap titik perpotongan dalam bintang David merupakan titik (*node*) penempatan angka. Seperti halnya, permainan pergeseran angka lainnya, dalam permainan ini juga disediakan satu tempat kosong sebagai ruang untuk menggeser posisi angka. Aturan permainannya adalah sebagai berikut:

1. Tentukan keadaan awal (*initial state*) pada bintang David.
2. Tentukan keadaan tujuan (*goal state*) pada bintang David.
3. Aturan pergeseran: Setiap angka dalam bintang David hanya dapat digeser ke suatu titik yang kosong atau tidak ditempati.

Permasalahan dari permainan ini adalah bagaimana mencapai *goal state* dari *initial state* dengan mengikuti aturan pergeseran yang telah ditetapkan.

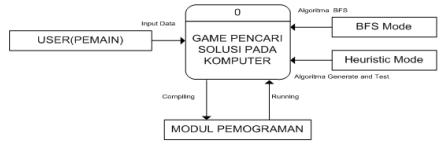


Gambar 2. Model bintang David

Permainan pergeseran angka dalam bintang David ini dapat diselesaikan dengan menggunakan bantuan struktur pohon pelacakan (*search tree*). Pohon pelacakan adalah suatu pohon (*tree*), dimana akar dari pohon berupa keadaan awal dan cabang berupa keadaan-keadaan yang mungkin terjadi dari keadaan sebelumnya serta daun merupakan keadaan akhir, yang dapat dijadikan sebagai solusi dari permasalahan. Namun, tidak semua daun dapat dijadikan sebagai solusi, dalam beberapa contoh kasus, ada beberapa atau semua daun bukan merupakan solusi dari permasalahan. Algoritma pencarian yang akan digunakan adalah algoritma *breadth-first search* dan algoritma pencarian heuristik**. [(John I.Bigg, 2001),(Permainan untuk IQ Super)]**.

METODE PENELITIAN

Dalam hal ini context diagram berfungsi sebagai media yang terdiri dari suatu proses dan beberapa komponen eksternal entitas. Adapun *context diagram* yang dimaksud dapat dilihat pada gambar berikut :



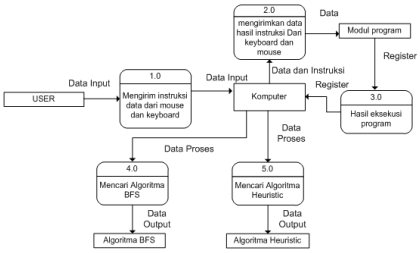
Gambar 3. Context Diagram

Pada *context diagram* diatas terdiri dari satu proses yang disebut dengan permainan pencarian solusi. Sistem ini berinteraksi dengan beberapa *entity* yaitu User (pemain), BFS *mode*, *Heuristic mode*, Modul Pemrograman, Selanjutnya entity-entity tersebut akan dibahas dibawah ini sebagai berikut :

1. *User*(Pemain) : Pemain disini untuk menginputkan data dan instruksi *initial state* dan *goal state* baik secara manual maupun acak, serta memilih metode tertentu.
2. BFS *Mode* : Metode algoritma pencarian buta atau tanpa berbekal informasi
3. *Heuristic Mode* : Metode algoritma dengan panduan atau berbekal informasi, metode yang digunakan mirip dengan BFS tetapi hanya saja menggunakan penyimpanan database.
4. Modul Pemrograman : Sarana pengolahan data dari input operator atau tempat user menginputkan data beserta metode. Dalam hal ini program sebagai *compiler* dan *running* menggunakan bahasa pemrograman Visual Basic 6.0. Jadi seluruh proses input/output dikendalikan oleh program.

Untuk memberikan penjelasan lebih mendetail mengenai perangkat lunak yang sedang dibangun ini bisa dilihat pada *data flow diagram*. Dimana didalam DFD jumlah *entity*nya harus sama dengan jumlah *entity context diagram*. Pada gambar dibawah terlihat bahwa *eksternal entity* pada *context diagram* sama dengan *entity* pada DFD

level 0, sedangkan proses dinaikkan dari 1 proses menjadi 5 sub proses. Adapun proses tersebut dapat dilihat pada gambar berikut ini.



Gambar 4. Data Flow Diagram (DFD) Level 0

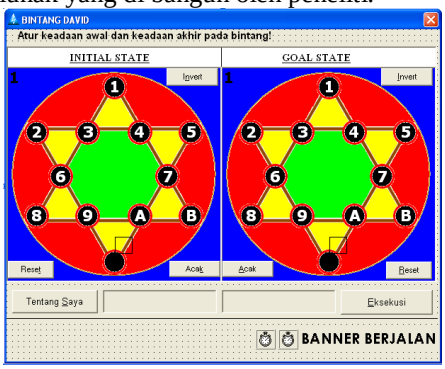
Setelah *User* memerintah instruksi *keyboard* dan *mouse* terpenuhi, maka keyboard dan mouse akan mengirim instruksi data input (1.0). Komputer akan mengirim data hasil instruksi dari keyboard dan mouse (2.0). Modul program akan menghasilkan hasil eksekusi (3.0). Komputer akan mengirim pengolahan data pada modus BFS atau Heuristic yang hasil berupa data proses yang menghasilkan data output berupa informasi (4.0)(5.0).

Database digunakan sebagai tempat penyimpanan pengetahuan heuristik. Dalam *database* tersebut terdapat sebuah tabel 'Heuristik'. Perancangan tabel ini adalah sebagai berikut:

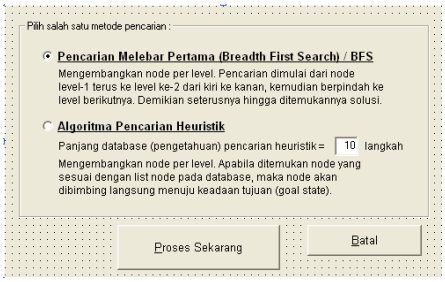
Tabel 1. Rancangan Tabel 'Heuristik'

No.	Field Name	Data Type	Field Size
1.	GoalState	Text	12
2.	Isi	Text	12
3.	Indeks	Number	Long Integer
4.	LevelS	Number	Long Integer
5.	ParentNode	Number	Long Integer
6.	Kosong	Number	Long Integer
7.	Pergeseran	Text	50

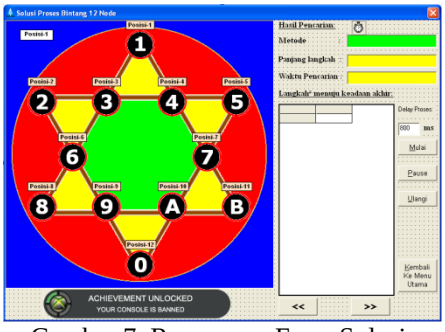
Berikut beberapa perancangan perangkat lunak yang di bangun oleh peneliti.



Gambar 5. Rancangan Form Set Keadaan Bintang David



Gambar 6. Rancangan Form Pilih Metode Pencarian

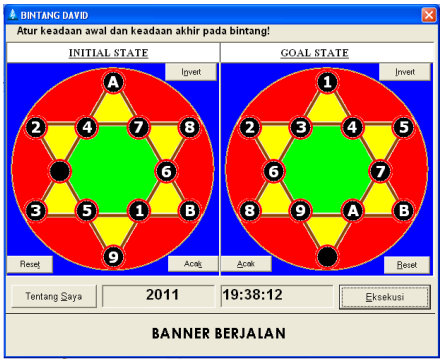


Gambar 7. Rancangan Form Solusi Bintang David

HASIL DAN PEMBAHASAN

Untuk menguji solusi yang dihasilkan oleh perangkat lunak, diambil contoh

kasus seperti terlihat pada gambar 8 berikut.



Gambar 8. Contoh kasus Perangkat Lunak

Untuk kasus pada gambar 8, solusi yang dihasilkan perangkat lunak dapat dilihat pada tabel 1 dan waktu pencarian yang dibutuhkan dapat dilihat pada tabel 2.

Tabel 2. Solusi Contoh Kasus Pada Bintang David

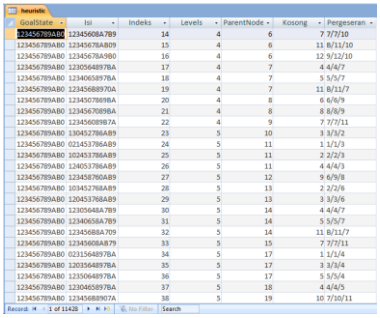
No.	Langkah
0.	Keadaan Awal.
1.	Geser angka ’5’ dari posisi 9 ke posisi 6.
2.	Geser angka ’1’ dari posisi 10 ke posisi 9.
3.	Geser angka ’6’ dari posisi 7 ke posisi 10.
4.	Geser angka ’8’ dari posisi 5 ke posisi 7.
5.	Geser angka ’7’ dari posisi 4 ke posisi 5.
6.	Geser angka ’A’ dari posisi 1 ke posisi 4.
7.	Geser angka ’4’ dari posisi 3 ke posisi 1.
8.	Geser angka ’5’ dari posisi 6 ke posisi 3.
9.	Geser angka ’1’ dari posisi 9 ke posisi 6.
10.	Geser angka ’6’ dari posisi 10 ke posisi 9.
11.	Geser angka ’8’ dari posisi 7 ke posisi 10.
12.	Geser angka ’A’ dari posisi 4 ke posisi 7.
13.	Geser angka ’5’ dari posisi 3 ke posisi 4.
14.	Geser angka ’1’ dari posisi 6 ke posisi 3.
15.	Geser angka ’3’ dari posisi 8 ke posisi 6.
16.	Geser angka ’6’ dari posisi 9 ke posisi 8.
17.	Geser angka ’8’ dari posisi 10 ke posisi 9.
18.	Geser angka ’A’ dari posisi 7 ke posisi 1.
19.	Geser angka ’7’ dari posisi 5 ke posisi 7.
20.	Geser angka ’5’ dari posisi 4 ke posisi 5.
21.	Geser angka ’4’ dari posisi 1 ke posisi 4.
22.	Geser angka ’1’ dari posisi 3 ke posisi 1.
23.	Geser angka ’3’ dari posisi 6 ke posisi 3.
24.	Geser angka ’6’ dari posisi 8 ke posisi 6.
25.	Geser angka ’8’ dari posisi 9 ke posisi 8.
26.	Geser angka ’9’ dari posisi 12 ke posisi 9.

Tabel 3. Waktu Pencarian Solusi

No	Metode Pencarian	Waktu Pencarian
1.	Pencarian BFS.	17.86025 detik.
2.	Pencarian Heuristik.	*Generate Time Varian* + 3.343875 detik.

Pada tabel 2 dapat dilihat bahwa pencarian heuristik jauh lebih singkat dan menghemat waktu pencarian ketimbang pencarian BFS karena metode pencarian heuristik metodenya lebih menyerupai *Depth-First Search* walaupun algoritma *generate* pencarian heuristik menggunakan metode *Breadth-*

First Search. Sedangkan Untuk melihat pemakaian database pada metode heuristic search dapat dilihat pada tabel gambar berikut.



GoalState	Isi	Indeks	Levels	ParentNode	Kosong	Pengeseran
123456789AB	12345608A789	14	4	6	7	7/7/10
123456789AB0	12345678A809	15	4	6	11	8/11/10
123456789AB0	12345678A9B0	16	4	6	12	9/12/10
123456789AB0	1230564897BA	17	4	7	4	4/4/7
123456789AB0	1234065897BA	18	4	7	5	5/5/7
123456789AB0	12345688970A	19	4	7	11	8/11/7
123456789AB0	12345078098A	20	4	8	6	6/6/9
123456789AB0	12345670898A	21	4	8	8	8/8/9
123456789AB0	12345608987A	22	4	9	7	7/7/11
123456789AB0	130452786AB9	23	5	10	3	3/3/2
123456789AB0	021453786AB9	24	5	11	1	1/1/3
123456789AB0	102453786AB9	25	5	11	2	2/2/3
123456789AB0	124053786AB9	26	5	11	4	4/4/3
123456789AB0	123458760AB9	27	5	12	9	6/9/8
123456789AB0	103452768AB9	28	5	13	2	2/2/6
123456789AB0	120453768AB9	29	5	13	3	3/3/6
123456789AB0	12305648A789	30	5	14	4	4/4/7
123456789AB0	12340658A789	31	5	14	5	5/5/7
123456789AB0	123456B8A709	32	5	14	11	8/11/7
123456789AB0	12345608AB79	33	5	15	7	7/7/11
123456789AB0	0231564897BA	34	5	17	1	1/1/4
123456789AB0	1203564897BA	35	5	17	3	3/3/4
123456789AB0	1235064897BA	36	5	17	5	5/5/4
123456789AB0	1230465897BA	37	5	18	4	4/4/5
123456789AB0	12345688907A	38	5	19	10	7/10/11

Gambar 9. Tabel Pengetahuan Heuristic Search

Sebelumnya Tabel pada *database Heuristic Search* belum terisi sama sekali. Maka data yang di-*generate* menggunakan panjang 15 langkah, sedangkan record yang dihasilkan 11428, apabila langkah yang dilakukan berlebih tidak bermasalah karena solusi tetap ditemukan tetapi lama waktu yang dibutuhkan akan bertambah, sedangkan apabila kurang kemungkinan solusi tidak ditemukan karena informasi yang diberikan tidak cukup.

Pada hakekatnya komputer dapat menyelesaikan solusi pohon dalam waktu yang singkat, akan tetapi semakin dalam jumlah atau *level* akar dalam suatu pohon semakin banyak waktu yang dibutuhkan karena cabang yang banyak.

KESIMPULAN

Peneliti menarik beberapa kesimpulan yaitu sebagai berikut :

1. Penggunaan metode BFS menjamin solusi yang ditemukan adalah solusi terpendek (shortest path) atau tidak ditemukan solusi sama sekali (hang).
2. Pencarian dengan bantuan heuristik (Algoritma Generate and Test) akan jauh lebih cepat dibandingkan dengan pencarian BFS. Hal ini dikarenakan pencarian heuristik memiliki pengetahuan yang tersimpan dalam database untuk membimbing setiap node (yang dikenali dan terdapat dalam database) untuk langsung menuju solusi tanpa mengembangkan node-node lain yang tidak berguna. Metode heuristik lebih menyerupai pencarian Depth-First Search.
3. Perangkat lunak merupakan implementasi nyata penggunaan metode pencarian Breadth First Search (BFS) dan pencarian heuristik dalam mencari solusi dalam suatu permasalahan berbasis Artificial Intelligence (AI).

DAFTAR PUSTAKA

[1] Suryanto.ST.Msc, Artificial Intelligence (Searching,Reasoning,Planning,And Learning), Penerbit Informatika, Bandung, 2007.

[2] Kusumadewi.S, Artificial Intelligence (Teknik dan Aplikasinya), Edisi 2,Penerbit Graha Ilmu, 2002.

[3] Desiani.A dan Arhami.M, Konsep Kecerdasan Buatan, Penerbit Graha Ilmu, 2002.

[4] Arhani.M , Konsep Dasar Sistem Pakar, Penerbit Andi, Yogyakarta, 2005.

[5] Hartati Sri dan Iswanti Sari, Sistem Pakar Dan Pengembangannya, Penerbit Graha Ilmu.Andi Offset Yogyakarta,2003.

[6] Halvorson Michael, Microsoft Visual Basic 6.0 Proffesional, Step by Step, Microsoft Press, Jakarta, 2000.

[7] Hadi.R, Pemrograman Microsoft Visual Basic dengan menggunakan Windows API, PT. Elex Media Komputindo, Jakarta, 2001.

[8] Ramadhan.A, 36 Jam Belajar Komputer Visual Basic 6.0, PT. Elex Media Komputindo, Jakarta, 2004.

[9] Zakaria.MY, Perancangan Antarmuka Untuk Interaksi Manusia dan Komputer, Penerbit Informatika, Bandung, 2007