

IMPLEMENTASI ALGORITMA PHYSICAL-A* (PHA*) UNTUK MENEMUKAN LINTASAN TERPENDEK

Kadek Ayu Yanti Pawitri¹, Joko Purwadi, M.Kom²

Program Studi Teknik Informatika Fakultas Teknik

Universitas Kristen Duta Wacana Yogyakarta 2007

Abstrak

Artikel ini membahas tentang implementasi algoritma Physical-A (PHA*) untuk menemukan lintasan terpendek. Lintasan terpendek (shortest path) adalah lintasan minimum yang diperlukan untuk mencapai suatu titik dari titik tertentu. Pencarian lintasan terpendek (shortest path problem), pada dasarnya adalah penyelesaian masalah untuk menentukan lintasan mana yang akan dilalui sehingga ditemukan lintasan terpendek dari titik awal ke tujuan.*

Permasalahan ini direpresentasikan dalam bentuk graf berbobot dimana setiap verteks digambarkan dengan koordinat 2 dimensi, dan bobot edgenya adalah jarak eucladian diantara dua verteks yang bersesuaian. Seorang mobile agent akan berpindah dari satu verteks ke verteks lainnya untuk menemukan lintasan terpendek dari keadaan awal ke tujuan.

Algoritma PHA* merupakan pengembangan dari algoritma A* dengan memodifikasi fungsi heuristiknya. Algoritma ini ditampilkan sebagai algoritma dua tingkat (two level algorithm) yaitu level atas (high-level) dan level bawah (low-level). Level atas akan memilih verteks yang akan diekspansi dan level bawahnya akan mengarahkan agent menuju verteks tersebut dengan tujuan mengeksplorasinya.

Kata Kunci : *graf, algoritma physical A* (PHA*), lintasan terpendek.*

1. Pendahuluan

Persoalan lintasan terpendek banyak dijumpai di kehidupan sehari-hari. Aplikasi yang paling sering ditemui adalah pada bidang transportasi dan komunikasi, seperti pada pencarian rute terbaik untuk menempuh dua kota atau untuk mengetahui dan menelusuri proses pengiriman paket data komunikasi dalam suatu jaringan komunikasi agar dihasilkan suatu proses yang paling cepat.

Tujuan penelitian ini adalah menerapkan salah satu algoritma yang digunakan untuk mencari lintasan terpendek, yaitu algoritma Physical A* (PHA*).

¹ Mahasiswa Teknik Informatika, Fakultas Teknik, Universitas Kristen Duta Wacana

² Dosen Teknik Informatika, Fakultas Teknik, Universitas Kristen Duta Wacana

Permasalahan utama dalam penelitian yaitu bagaimana hasil penerapan metode Physical A* dalam pencarian lintasan/jalur terpendek. Sebagai studi kasus dipergunakan beberapa daerah-daerah di sekitar kota Denpasar – Bali.

2. Landasan Teori

2.1 Kecerdasan Buatan (*Artificial Intelligence*)

Kecerdasan Buatan atau *Artificial Intelligence* merupakan salah satu bagian ilmu komputer yang membuat atau memprogram komputer agar dapat melakukan pekerjaan seperti yang dilakukan oleh manusia.

Sistem yang menggunakan kecerdasan buatan, memiliki input berupa masalah dan menghasilkan output berupa solusi dari masalah yang diinputkan. Untuk dapat menghasilkan solusi, sistem harus dilengkapi dengan sekumpulan pengetahuan yang ada pada basis pengetahuan, selain itu sistem juga harus memiliki motor inferensi (*inference engine*) agar mampu mengambil keputusan berdasarkan fakta atau pengetahuan.

2.2 Algoritma Physical A*

Algoritma PHA* merupakan pengembangan dari algoritma A* dengan memodifikasi fungsi heuristiknya. Untuk memperluas verteks dengan algoritma PHA*, verteks terlebih dahulu harus dieksplorasi oleh agent dengan tujuan mendapatkan data-data relevan yang berhubungan dengan verteks tersebut yaitu verteks tetangga (*neighbouring nodes*) dan edge yang menghubungkannya (*incident edge*). Verteks dalam graf dapat digolongkan menjadi 2, yaitu verteks yang sudah dieksplorasi dan verteks yang belum dieksplorasi (*explored and unexplored nodes*).

Mengeksplorasi suatu verteks berarti agent secara fisik mengunjungi verteks tersebut, mempelajari lokasinya, serta lokasi verteks tetangganya. Algoritma ini membutuhkan dua antrian yang digunakan untuk menyimpan verteks-verteks yang ada, yaitu:

1. Open, berisi verteks-verteks yang sudah dibangkitkan dan memiliki fungsi heuristic namun belum dieksplorasi.
2. Closed, berisi verteks-verteks yang sudah dieksplorasi.

Algoritma PHA* ditampilkan sebagai algoritma dua tingkat (*two level algorithm*) yaitu level Atas (*high-level*) dan level bawah (*low-level*). Level atas pada prinsipnya bekerja seperti algoritma A*. Level ini akan memilih satu verteks di tiap-tiap lingkaran (*cycle*) yang berada pada Open untuk diekspansi. Fungsi heuristic $h(n)$ yang digunakan adalah jarak eucladian diantara verteks n dan verteks tujuan.

Jika verteks yang telah dipilih belum dieksplorasi oleh agent, maka level bawahnya, yang berupa algoritma navigasi, akan diaktifkan untuk mengarahkan agent menuju verteks tersebut dan mengeksplorasinya. Setelah verteks dieksplorasi oleh level bawah maka verteks dapat diperluas oleh level atas. Jika verteks yang dipilih telah dieksplorasi, atau verteks tetangganya telah diketahui, maka verteks tersebut akan dengan mudah dapat diperluas tanpa perlu mengirim agent untuk mengunjunginya.

Urutan langkah yang digunakan untuk mencari lintasan terpendek dengan menggunakan *Physical A** adalah:

1. Mulai
2. Tentukan verteks awal dan verteks tujuan
3. Set: OPEN={S}, dan CLOSED={}, dengan S adalah verteks yang dipilih sebagai keadaan awal.
4. Selama verteks tujuan belum berada dalam CLOSED (*closed list*), kerjakan langkah 5-9

5. Pilih verteks n dari OPEN

Untuk iterasi pertama, n = S

Untuk iterasi selanjutnya, verteks yang diekspansi adalah verteks yang memiliki nilai $f(n)$ dan $c(n)$ yang terkecil.

6. Bangkitkan semua cabang n.
7. Kerjakan untuk setiap anak n, yaitu n'. Hitung nilai fungsi:

- a. $g(n')$
- b. $h(n')$
- c. $f(n') = g(n') + h(n')$
- d. $c(n') = (g(n') + h(n')) \cdot dist(curr, n')$

8. Masukkan n' ke antrian OPEN
9. Jika semua anak n sudah diekspansi, maka masukkan n yang memiliki nilai $f(n)$ terkecil ke CLOSED
10. Selesai.

2.3 Teori Graf

Graf linier atau graf $G = (V, E)$ terdiri dari himpunan objek $V = \{v_1, v_2, \dots\}$ yang disebut verteks dan himpunan lain $E = \{e_1, e_2, \dots\}$ yang elemen-elemennya disebut edge, sedemikian

sehingga tiap-tiap edge e_k didefinisikan dengan pasangan verteks tak urut (*Unordered*)^[3]. Verteks v_i, v_j yang berasosiasi dengan edge e_k disebut verteks ujung (*end-verteks*) dari e_k .

Terdapat beberapa istilah penting yang berkaitan dengan graf. Berikut ini didefinisikan beberapa terminologi yang sering digunakan:

1. Graf tak berarah didefinisikan sebagai suatu pasangan berurutan $G = (V, E)$, yang mempunyai sifat-sifat sebagai berikut^[4]:
 - Komponen pertama, V adalah terbatas (*finite*), himpunan yang tidak kosong. Elemen V ini disebut verteks dari G .
 - Komponen kedua, E , adalah himpunan terbatas dari suatu himpunan. Setiap elemen E , adalah suatu himpunan yang terdiri dari tepat dua verteks yang berbeda. Elemen E disebut edge dari G .
2. Graf G dikatakan berbobot (*weighted*) bila terdapat bilangan riil yang berasosiasi dengan masing-masing edge G ^[5]. Bobot-bobot itu sendiri biasanya berfungsi pada saat dibuat suatu lintasan antar verteks-verteks pada graf tersebut.

3. Hasil Implementasi Algoritma Physical A* (PHA*) Untuk Menemukan Lintasan Terpendek

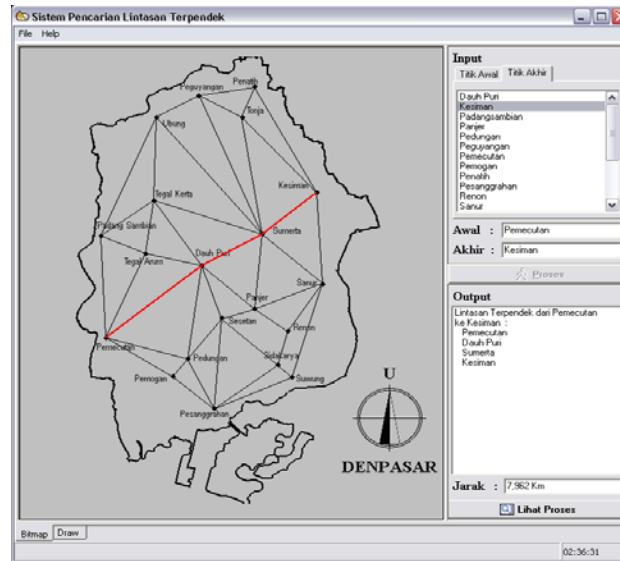
Pada bagian ini disajikan hasil implementasi algoritma PHA* untuk menemukan lintasan terpendek. Tampilan utama dari sistem ini adalah:

- Tab Bitmap

^[3]Narsingh Deo, Graph Theory With Application to Engineering and Computer Science (Prentice Hall), hlm. 1.

^[4]Bruno R. Preiss, P.Eng (1997), *Data Structures and Algorithms with Object-Oriented Design Patterns in C++* (<http://www.brpreiss.com/books/opus4/html/page529.html>)

^[5]L. Allison (1999), *Graph* (<http://www.csse.monash.edu.au/~llyoyd/tildeAlgDS/Graph/>)



Gambar 3.1

Form Pencarian Lintasan Terpendek - Tab BitMap

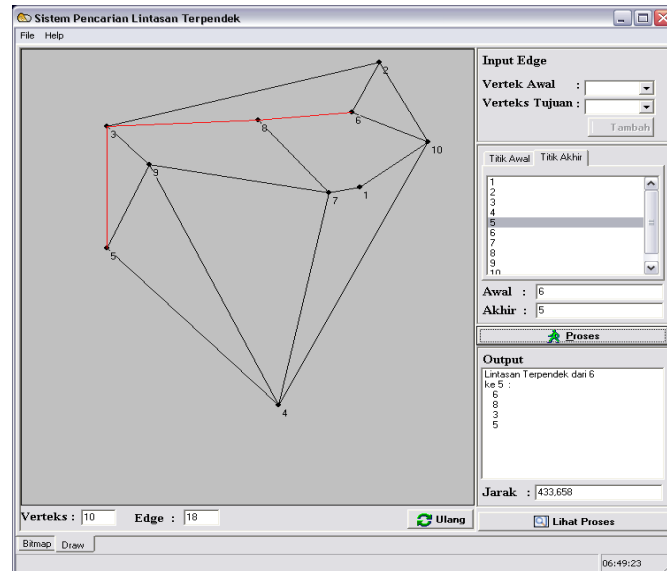
Gambar 3.1 menunjukkan tampilan Form Pencarian Lintasan setelah dilakukan proses pencarian lintasan terpendek. Sistem akan menampilkan titik-titik yang harus dilewati serta total jarak tempuhnya. Sistem juga memvisualisasikan lintasan yang terbentuk dengan memberi warna merah pada lintasan yang bersesuaian. Tombol Lihat Proses, berfungsi untuk menampilkan Form Proses. Form ini berisi perhitungan yang dilakukan oleh algoritma Physical A* untuk menemukan lintasan terpendek. Nilai fungsi $g(n)$, $h(n)$, $f(n)$ dan $c(n)$ untuk masing-masing verteks juga ditampilkan pada Form Proses. Gambar 3.2 menunjukkan tampilan Form Proses.

Proses Perhitungan								
Proses Algoritma Physical A* (PHA*)								
Titik Awal dan Titik Akhir								
Titik Awal : Pemecutan Titik Akhir : Kesiman								
Verteks Ekspansi	Cabang VertekEks	Fungsi g(n) High	Fungsi g(n) Low	Fungsi h(n) High	Fungsi h(n) Low	Fungsi f(n) High	Fungsi f(n) Low	Fungsi c(n) Low
Pemecutan	Tegal Arum	2729,46881279124	2729,46881279124	5756,7351858497	8486,20399864094	8486,20399864094	11215,6728114322	30612829,1532747
	Dauh Puri	3827,53184180093	3827,53184180093	4118,2520563948	7945,78389819573	7945,78389819573	11773,3157399967	45062740,8784132
	Pedungan	2816,02956806574	2816,02956806574	6389,05313798531	9205,07870605106	9205,07870605106	12021,1042741168	33851736,9922973
	Pemogan	2955,38646783613	2955,38646783613	7093,65914038728	9649,04560822341	9649,04560822341	12204,4320760595	31187040,5747877
Dauh Puri	Sumerta	2193,17121994613	6020,70306174706	1941,64878389476	7962,35184564182	7962,35184564182	13983,0549073889	30667233,5898118
	Tegal Kerta	2262,74189979695	6090,27354158788	5215,36192416212	11305,635465676	11305,635465676	17395,9090073579	39362448,7168221
	Sesetan	1708,80074906351	5536,33259086443	4686,14980554399	10222,4823964084	10222,4823964084	15758,8149872729	26928674,8546051
	Tegal Arum	1923,53840616713	5751,07024796806	5756,7351858497	11507,8054338178	9071,36827695126	14822,4385249193	28511529,7757336
	Pemecutan	3827,53184180093	7655,06368360185	7940,40301244213	15595,466896044	15595,466896044	23250,5303796458	88992145,3668543
	Pedungan	2942,78779391243	6770,31963571336	6389,05313798531	13159,3727736987	10885,7793916758	17656,0990273891	51958152,7059099
	Kesiman	1941,64878389476	7962,35184564182	0	7962,35184564182	7962,35184564182	15924,7036912836	30920181,5560653
Lintasan Terpendek: Pemecutan -> Dauh Puri -> Sumerta -> Kesiman								

Gambar 3.2

Tampilan Form Proses

- Tab Draw

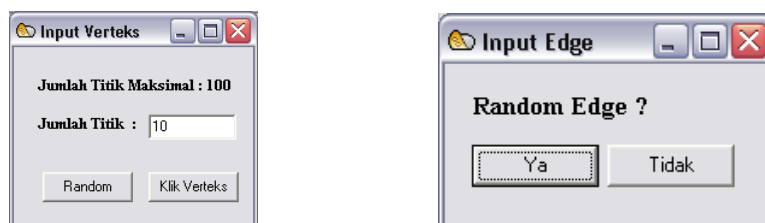


Gambar 3.3

Form Lintasan Terpendek –Tab Draw

Pada Tab Draw terdapat 4 pilihan menu yaitu: New, Open, Save, dan Exit. Ketika user memilih New maka akan muncul Form Input Verteks. Pada form ini, user diminta untuk memasukkan jumlah verteks. Nilai minimal yang boleh dimasukkan adalah 2 dan nilai maksimal adalah 100. User dapat memilih tombol Klik Verteks jika ingin menentukan sendiri posisi dari setiap titik yang akan digambar dengan cara menekan tombol mouse pada kotak image atau sebaliknya, memilih tombol Random.

Jika tombol Random yang dipilih, maka sistem akan membangkitkan sejumlah bilangan random untuk menggambar verteks. Posisi dari titik-titik tersebut juga akan di random oleh sistem. Ketika user memilih tombol random, maka sistem akan memberikan pilihan apakah user ingin menggambar sendiri edge-edgenya atau tidak. Jika pilihannya ya, maka sistem kembali akan membangkitkan bilangan random untuk menggambarkan edge-edge sehingga terbentuk suatu graf. Akan tetapi, jika pilihannya adalah Tidak, maka user sendiri yang harus menggambar edgenya.



Gambar 3.4

Form Input Verteks dan Input Edge

Procedure utama yang terdapat pada form pencarian lintasan adalah **Procedure btnProsesClick(Sender: TObject)**. Jika procedure ini dijalankan maka sistem akan melakukan proses pencarian lintasan terpendek.

- Pernyataan

```
SetLength(arrWindow,1);

for j := 1 to 2 do

begin

if arrEks[j].Nama <> " then

begin

verteksEks := arrEks[j].Nama;

jarak := arrEks[j].g_n;

Table2.First;

while not Table2.eof do

begin

if Table2['Titik'] = verteksEks then

begin

w := Length(arrWindow)+1;

SetLength(arrWindow,w);

arrWindow[w-2].Nama := Table2.FieldName('Tetangga').AsString;

arrWindow[w-2].parent:= verteksEks;

arrWindow[w-2].g_n := jarak;

end;

Table2.Next;

end;
```

end;

end;

digunakan untuk membangkitkan semua cabang dari verteks ekspansi. Cabang-cabang tersebut kemudian disimpan ke *arrWindow*.

- Pernyataan

for k := 0 to Length(arrWindow)-2 do

begin

Table1.First;

Table1.IndexFieldNames := 'Nama';

Table1.FindKey([arrWindow[k].parent]);

xEks := Table1.FieldByName('KoorX').AsFloat;

yEks := Table1.FieldByName('KoorY').AsFloat;

Table1.First;

Table1.IndexFieldNames := 'Nama';

Table1.FindKey([arrWindow[k].Nama]);

arrWindow[k].X := Table1.FieldByName('KoorX').AsFloat;

arrWindow[k].Y := Table1.FieldByName('KoorY').AsFloat;

g_n1 := Sqrt(sqr(xEks - arrWindow[k].X) + sqr(yEks - arrWindow[k].Y));

g_n := g_n1 + arrWindow[k].g_n;

h_n := Sqrt(sqr(arrWindow[k].X - xGoal) + sqr(arrWindow[k].Y - yGoal));

f_n := g_n + h_n;

if arrWindow[k].open = 'yes' then

*h_n := f_n * (1 - 0.25 * Power(f_t/arrWindow[k].f_n, 2.5))*

else

h_n := f_n;

f_n := g_n + h_n;

$$c_n := f_n * g_{nI};$$

digunakan untuk menghitung nilai $g(n)$, $h(n)$, $f(n)$, dan $c(n)$ untuk semua verteks yang ada pada arrWindow.

4. Analisis Hasil Penelitian

Pada sub bab ini akan diuraikan analisis dari implementasi algoritma Physical A* untuk menemukan lintasan terpendek. Analisis diuraikan dengan proses perhitungan yang dilakukan oleh sistem untuk menemukan lintasan terpendek.

Verteks awal : Penatih (0305700, 9045900)

Verteks akhir : Sumerta (0305000, 9041900)

1. Set OPEN {Penatih} dan CLOSED { }

VerteksEks : Penatih (0305700, 9045900)

Cabang : Kesiman (0306600, 9043000), Peguyangan (0303000, 9045800), Tonja (0304100, 9045200)

TABEL 4.1

Hasil perhitungan nilai $g(n)$, $h(n)$, $f(n)$ dan $c(n)$ untuk setiap cabang Penatih

Cabang V_e	High Level				Low Level		
	$g(n)$	$g(n')$	$h(n')$	$f(n')$	$h(n')$	$f(n')$	$c(n')$
Peguyangan	2,7018	2,7018	4,3829	7,0847	7,0847	9,7866	26,4420
Tonja	1,7464	1,7464	3,4205	5,1669	5,1669	6,9133	12,0736
Kesiman	3,0364	3,0364	1,9416	4,9780	4,9780	8,0145	24,3357

2. OPEN{Peguyangan, Tonja, Kesiman} CLOSED{Penatih}

VerteksEks : Kesiman (0306600, 9043000), Tonja (0304100, 9045200)

Cabang : Penatih (0305700,9045900) , Peguyangan (0303000,9045800) , Kesiman (0306600,9043000) , Sumerta (0305000,9041900)

TABEL 4.2

Hasil perhitungan nilai $g(n)$, $h(n)$, $f(n)$ dan $c(n)$ untuk setiap cabang Kesiman dan Tonja

Cabang V_e	High Level				Low Level		
	$g(n)$	$g(n')$	$h(n')$	$f(n')$	$h(n')$	$f(n')$	$c(n')$
Cabang Kesiman							
<u>Penatih</u>	3,0364	6,0728	4,0607	10,133	10,133	16,2065	49,2103
<u>Tonja</u>	3,3301	6,3666	3,4205	9,7871	8,4473	14,8139	49,3329
Sumerta	1,9416	4,9780	0	4,9780	4,9780	9,9561	19,3314
Sanur	2,6305	5,6670	2,5	8,1670	8,1670	13,834	36.3917
Cabang Tonja							
<u>Penatih</u>	1,7464	3,4928	4,0607	7,5536	7,5536	11,0464	19,2918
<u>Peguyangan</u>	1,2529	2,9994	4,3829	7,3823	6,9233	9,9227	12,4331
Sumerta	3,4205	5,1669	0	5,1669	5,1669	10,3339	35,3473
<u>Kesiman</u>	3,3301	5,0765	1,9416	7,0182	5,9637	11,0403	36,7662

3. OPEN{Peguyangan, Tonja, Kesiman, Sanur, Sumerta}

CLOSED{Penatih, Kesiman}

Verteks yang selanjutnya di ekspansi adalah Sumerta. Verteks ini otomatis akan dimasukkan ke CLOSED setelah proses ekspansi selesai. Karena verteks Sumerta (*goal*) telah berada di CLOSED, maka proses pencarian akan berhenti. Jadi didapatkan jalur terpendek dari Penatih ke Sumerta adalah : **Penatih – Kesiman - Sumerta**

5. Kesimpulan

Hasil analisis penelitian yang telah dilakukan, dapat disimpulkan bahwa Algoritma Physical A* (PHA*) merupakan :

1. Algoritma PHA* merupakan pengembangan dari algoritma A* dengan memodifikasi fungsi heuristiknya.
2. Algoritma PHA* ditampilkan sebagai algoritma dua tingkat (*two level algorithm*) yaitu level atas (*high-level*) dan level bawah (*low-level*). Level atas memilih verteks yang diekspansi dan level bawah mengarahkan agent menuju verteks tersebut dengan tujuan mengeksplorasinya.

3. Algoritma A* memperluas verteks-verteksnya menurut nilai fungsi f yang terbaik (*best-first order*). Dalam kenyataannya, tidak selalu efisien jika agent memperluas verteks-verteksnya hanya dengan mempertimbangkan nilai $f(n)$ -nya. Algoritma Physical A*, memperluas verteks-verteksnya tidak hanya berdasarkan nilai fungsi $f(n)$ yang terbaik tetapi juga mempertimbangkan jarak antara verteks n dengan lokasi agen saat itu ($c(n)$).

6. Daftar Pustaka

- Cantu's Marco. 2002. *Essential Delphi: A Friendly Introductory Guideto Borland Delphi*. <http://www.marcocantu.com/edephi>
- Deo, Narshingh. 1974. *Graph Theory With Application to Engineering and Computer Science*. New Jersey: Prentice Hall, Englewood Clif.
- Diestel, Reinhard. 2000. *Graph Theory: Electronic Edition 2000*. <http://www.math.uni-hamburg.de/home/diestel/books/graph>.
- Felner, Ariel, Roni Stern, Asaph Ben-Yair, Sarit Kraus, and Nathan Netanyahu. (2004). *PHA*: Finding the Shortest Path with A* in Unkown Physical Environmet*. Journal Of Artificial Intelegence Research 21. <http://www.jair.org>
- Gross, Jonathan, Jay Kellen. 1999. *Graph Theory and Its Application*. London: CRC Press.
- J.Wilson, Robin. 1985. *Introduction to Graph Theory*. United States: John Willey & Sons Inc.
- Kusumadewi, Sri. 2003. *Artificial Intelegence (Teknik dan Aplikasinya)*. Yogyakarta: Graha Ilmu.
- Prahasta, Edi. 2005. *Konsep-konsep Dasar Sistem Informasi Geografis*. Bandung: CV. Informatika.