Pencarian Rute Terpendek pada Citra Labirin Menggunakan Algoritma Dijkstra sebagai Pemandu Gerak Micromouse Robot

[1)]Dwijaya Santoso [2)]Harianto [3)]Ihyauddin

1)Program Studi Sistem Komputer STIKOM Surabaya. Email: dwijaya.santoso@gmail.com

2)Program Studi Sistem Komputer STIKOM Surabaya. Email: hari@stikom.edu

3)Program Studi Sistem Komputer STIKOM Surabaya. Email: ihya@stikom.edu

Abstract

Micromouse Robot has a goal to complete the maze that is the path to find a route from start point to finish with the shortest route. In search of the shortest route, Micromouse robot must perform scanning area of the maze path directly to determine the weights as input a shortest route algorithm.

Processing on the image of the labyrinth and the Dijkstra algorithm to be the main discussion in this study. In practice, create an image processing algorithm to detect the nodes on the image of the labyrinth that the result becomes the input to the Dijkstra algorithm in order to get Micromouse Robot motion directions.

These Applications on computers have been able to detect the nodes on the image of the labyrinth if the track width of a cell evenly, and experienced an error if there is a point that the width of the tracks is uneven in the amount of two cells. The shortest route formed by the application of the computer is in accordance with the route established by the program on Micromouse Robot.

Keyword: Dijkstra Algorithm, Image Processing, Microcontroller.

Micromouse robot merupakan salah satu *mobile robot* yang memiliki tujuan untuk menyelesaikan lintasan berupa labirin, sehingga robot dapat menemukan tempat yang dituju melalui rute dengan jarak yang terpendek. Penelitian terhadap *Micromouse Robot* untuk mencari jalur terpendek pada lintasan labirin telah dilakukan Rusmini Setiawardhana dan M. Iqbal Nugrah dengan menggunakan algoritma *backtracking*. Pada penelitian Anita Nur Syafidtri, *Micromouse Robot* untuk mencari jalur terpendek dari titik start ke finish pada lintasan labirin menggunakan algoritma *Depth-First Search*.

Pada penelitian tersebut dalam menyelesaikan lintasan labirin *Micromouse Robot* harus melakukan *scanning area* secara langsung untuk mendapatkan data-data bobot sebagai input algoritma pencari jalur terpendek dengan cara melalui semua jalan di dalam lintasan labirin tersebut.

Pengembangan selanjutnya pada *Micromouse Robot* dapat dilakukan dengan mengambil citra dari area labirin, melakukan pengolahan citra tersebut pada PC (*Personal Computer*) sehingga data-data bobot yang diperlukan untuk penyelesaian rute terpendek dengan menggunakan algoritma Dijkstra dapat diketahui tanpa menentukan data-data bobot tersebut sebelumnya, dan mengirim hasil pengolahan ke *Micromouse Robot*, sehingga robot tidak perlu lagi melakukan *scanning* area pada lintasan labirin.

Algoritma Dijkstra

Algoritma Dijkstra ditemukan oleh Edsger W. Dijkstra yang merupakan salah satu varian bentuk algoritma popular dalam pemecahan persoalan yang terkait dengan masalah optimisasi dan bersifat sederhana. Algoritma ini menyelesaikan masalah mencari sebuah lintasan terpendek (sebuah lintasan yang mempunyai panjang minimum) dari *vertex* a ke *vertex* z dalam *graph* berbobot, bobot tersebut adalah bilangan positif jadi tidak dapat dilalui

oleh *node* negatif, namun jika terjadi demikian, maka penyelesaian yang diberikan adalah *infinity*. (Lubis, 2009).

Dengan simpul awal adalah *a*, dan dengan jarak dari simpul *i* diartikan sebagai jarak antara simpul *a* dan *i*, maka algoritma Djikstra akan menginisialisasikan nilai jarak awal dan memperbaikinya tahap demi tahap.

Pseudo code algoritma Dijkstra dapat ditunjukkan sebagai berikut:

```
function Dijkstra(Graph, source):
for each vertex v in Graph:
// Initializations
dist[v] := infinity ;
// Unknown distance function from
source to v
previous[v] := undefined ;
// Previous node in optimal path from
source
end for ;
dist[source] := 0 ;
// Distance from source to source
Q := the set of all nodes in
Graph ;
// All nodes in the graph are
unoptimized - thus are in Q
while Q is not empty:
// The main loop
u := vertex in Q with
smallest dist[] ;
if dist[u] = infinity:
break ;
// all remaining vertices are
inaccessible from source
fi ;
remove u from Q ;
for each neighbor v of u:
// where v has not yet been removed
from Q.
alt := dist[u] +
dist_between(u, v) ;
if alt < dist[v]:
// Relax (u,v,a)
dist[v] := alt ;
previous[v] := u ;
fi ;
end for ;
end while ;
return dist[] ;
end Dijkstra.
```

(Handaka, 2010).

Kecerahan Gambar

Pengubahan kecerahan gambar bertujuan untuk membuat citra menjadi lebih terang atau lebih gelap. Kecerahan gambar dapat diperbaiki dengan menambah atau mengurangi setiap pixel pada citra dengan sebuah nilai konstan.

Secara matematis operasi ini dapat ditunjukkan seperti pada persamaan berikut (Munir, 2004):

$$fx,y' = fx,y + b$$

Algoritma pengubahan kecerahan gambar dapat ditunjukkan sebagai berikut:

```
void ImageBrightness(citra Image, int N, int M, int b)
```

```
/* Mengubah kecerahan citra image yang berukuran N x M dengan penambahan
intensitas sebesar b.
*/
{     int i, j, n;
      for(i=0;i<=N-1;i++)
            for(j=0;j<=M-1;j++)
                  Image[i][j]+=b;
}
```
(Munir, 2004).

Konversi Citra *True Color* ke *Grayscale*

Citra *true color* dapat dikonversi menjadi *grayscale* menggunakan operasi titik. Rumus yang digunakan untuk konversi ini dapat ditunjukkan seperti berikut (Basuki, 2005) :

$$Ko=R+G+B3$$

Modifikasi rumus di atas maka didapatkan seperti berikut (Basuki, 2005):

$$Ko=wr.R+wg.G+wb.B$$

Bobot-bobot wr, wg, dan wb merupakan bobot untuk elemen warna merah, hijau, dan biru. *National Television System Committee* (NTSC) mendefinisikan bobot untuk konversi citra *true color* ke *grayscale* ini adalah sebagai berikut: wr = 0.299, wg = 0.587, dan wb = 0.144.

Thresholding

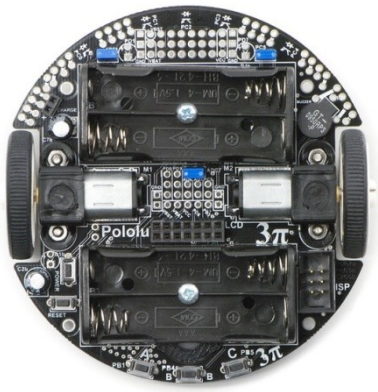
Contoh operasi titik berdasarkan intensitas adalah operasi pengambangan (*thresholding*). Pada operasi pengambangan, nilai intensitas pixel dipetakan ke salah satu dari dua nilai, α_1 atau α_2, berdasarkan nilai ambang (*threshold*) T dapat ditunjukkan seperti berikut:

$$fx,y'=\alpha_1,\ fx,y<T\alpha_2,\ fx,y\geq T$$

Jika $\alpha_1 = 0$ dan $\alpha_2 = 1$, maka operasi pengambangan mentransformasikan citra hitam-putih ke citra biner. Dengan kata lain, nilai intensitas *pixel* semula dipetakan ke dua nilai saja: hitam dan putih. Nilai ambang yang dipakai dapat berlaku untuk keseluruhan *pixel* atau untuk wilayah tertentu saja (berdasarkan penyebaran nilai intensitas pada wilayah tersebut).
(Munir, 2004).

Mobile Robot

Micromouse Robot menggunakan *mobile robot* komersil Pololu 3π seperti yang ditunjukkan pada gambar 1.



Gambar 1. Pololu 3π *robot*.

Micromouse Robot yang digunakan dilengkapi dengan sensor dinding IR (Infra Red) dan Mikrokontroler ATMega644P sebagai pengendali utama yang terhubung dengan mikrokontroler pada *Pololu 3π robot* yaitu ATMega328P sebagai pengendali motor. (Pololu).

Struktur Data

Dalam pemrograman untuk menyelesaikan permasalahan pencarian rute dengan jarak terpendek pada lintasan labirin menggunakan struktur data yang bersifat dinamis yang mempunyai kelebihan dapat menyesuaikan penggunaan tempat pada media penyimpanan sesuai dengan kebutuhan. Dalam algoritma Dijkstra struktur data yang digunakan dalam pengolahan berbentuk graf, dimana sebuah *vertex* (titik) diwakili oleh sebuah *set of data* dan sebuah *edge* (jarak dari sebuah *vertex* satu ke sebuah *vertex* yang lain) diwakili sebuah data. Dalam sebuah graf untuk mewakili area labirin maka jumlah *vertex* yang disimpan akan tergantung dari jumlah persimpangan dan jalan buntu yang ada. Secara keseluruhan sebuah graf dengan menggunakan struktur data DLL (*Double Linked List*) dapat diilustrasikan seperti yang ditunjukkan pada gambar 5.

Gambar 5 (a). Bentuk *node* suatu persimpangan pada lintasan. 5 (b) Ilustrasi bentuk struktur data graf.

Gambar 5 (b) merupakan bentuk perwujudan bentuk struktur data dari *node* suatu persimpangan seperti yang ditunjukkan pada gambar 5 (a).

Flowchart Aplikasi Pengolahan Citra pada PC

Perancangan aplikasi pada PC menggunakan bahasa pemrograman Pascal pada compiler Borland Delphi 2010. *Flowchart* dari perancangan program secara umum seperti yang ditunjukkan pada gambar 6.

Gambar 6. *Flowchart* aplikasi pada PC.

Secara umum fungsi aplikasi pada PC yaitu untuk menggolah citra lintasan labirin menjadi hitam dan putih, mencari *nodes* pada citra dan mengirimkan *nodes* tersebut ke *Micromouse Robot*.

Flowchart Pencarian Node

Flowchart dari proses pencarian *node* pada citra labirin dapat ditunjukkan seperti pada gambar 7.

Gambar 7. *Flowchart* proses pencarian *node*.

Yang menjadi *node* pada labirin yaitu persimpangan dan jalan buntu. Ada atau tidaknya sebuah *node* dideteksi dengan ada atau tidaknya dinding sisi depan, kiri, dan kanan terhadap arah hadap. Dinding atau lintasan dibedakan dari warna pixel pada citra labirin. Dinding memiliki warna hitam dan lintasan memiliki warna putih.

Flowchart Program pada ATMega644P

Perancangan perangkat lunak pada mikrokontroler menggunakan bahasa pemrograman C pada compiler Code Vision AVR 2.03.4. *Flowchart* dari perancangan program secara umum seperti yang ditunjukkan pada gambar 8.

Gambar 8. *Flowchart* program pada mikrokontroler ATMega644P.

Secara umum pada ATMega644P terdapat beberapa fungsi diantaranya penerimaan data dari PC, komunikasi antara ATMega644P dengan ATMega328P sebagai pengatur kecepatan putar motor, penampil ke layar LCD, dan sebagai pembaca sensor dinding. Data nodes yang diterima dari PC diolah menggunakan Algorima Dijkstra yang ditanamkan pada ATMega644P untuk mencari jalur terpendek dari titik start menuju finish, dan diolah kembali hingga menjadi panduan arah gerak.

HASIL DAN PEMBAHASAN

Hasil pengambilan citra oleh aplikasi pada PC dapat ditunjukkan seperti pada gambar 9.

Gambar 9. Hasil pengambilan gambar.

Pengolahan citra dan pencarian *nodes* labirin hasil pengambilan aplikasi pada PC dapat ditunjukkan seperti pada gambar 10.

Gambar 10. Hasil pengolahan citra dan pencarian *nodes*.

Rute pada labirin, titik *start* terletak pada *node* ke-32 dan titik *finish* terletak pada *node* ke-0 yang dapat ditunjukkan seperti pada gambar 11.

Gambar 11. Rute ke-1 pada labirin bentuk ke-1.

Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 1.

Tabel 1. Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
32	25	0	0	Sesuai
25	33	3	3	Sesuai
33	27	3	3	Sesuai
27	21	0	0	Sesuai
21	14	1	1	Sesuai
14	9	3	3	Sesuai

9	6	0	0	Sesuai
6	3	0	0	Sesuai
3	1	3	3	Sesuai
1	0	3	3	Sesuai

Simpulan

Aplikasi pada PC dapat melakukan pengolahan citra labirin sehingga pada dinding labirin berwarna hitam dan pada lintasan berwarna putih. Aplikasi juga dapat mendeteksi adanya *nodes* dan jumlah *nodes* yang terdeteksi oleh aplikasi pada PC sesuai dengan perhitungan *nodes* secara manual jika pada semua lintasan memiliki lebar yang sama. Aplikasi akan mengalami *error* dalam pencarian *nodes* jika terdapat lintasan yang memiliki lebar tidak sama.

Struktur data graf dapat digunakan untuk menyatakan hubungan antar *node* yang kemudian diolah untuk mencari rute terpendek dari titik *start* menuju *finish* menggunakan algoritma Dijkstra.

Rute terpendek *output* dari algoritma Dijkstra yang dilakukan pada *Micromouse Robot* berkesuaian dengan yang dilakukan oleh aplikasi pada PC.

Pergerakan *Micromouse Robot* dari titik *start* menuju titik *finish* mengalami kegagalan pada pendeteksian simpangan disebabkan ketidakstabilan sensor oleh pengaruh cahaya dan tidak meratanya warna pada dinding labirin.

DAFTAR PUSTAKA

Basuki, A., Palandi, J.F. 2005. *Pengolahan Citra Digital Menggunakan Visual Basic*. Jogjakarta: Graha Ilmu.

Munir, Renaldi. 2004. *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung: Informatika.

Handaka, Michell Setyawati. 2010. *Perbandingan Algoritma Dijkstra (Greedy), Bellman-Ford (BFS-DFS), dan Floyd-Warshall (Dynamic Programming) dalam Pengaplikasian Lintasan Terpendek pada Link-State Routing Protocol*. (Online) (http://www.informatika.org/~rinaldi/Stmik/2010-2011/Makalah2010/MakalahStima2010-040.pdf, diakses 8 Juni 2011).

Lubis, Henny Syahriza. 2009. *Perbandingan Algoritma Greedy dan Dijkstra untuk Menentukan Lintasan Terpendek*. (Online) (http://repository.usu.ac.id/bitstream/123456789/14038/1/09E00633.pdf, diakses 8 Juni 2011).

Setiawardhana, Rusmini, M. Iqbal Nugraha. 2011. *Pencarian Jalur Terpendek untuk Robot Micromouse dengan Menggunakan Algoritma Backtracking*. (Online) (http://repo.eepis-its.edu/1459/1/[A-D206-9]_pp.41-48_Pencarian_Jalur_Terpendek_Untuk_Robot_Micromouse_Dengan.pdf, diakses 8 Juni 2011).

Nur Syafidtri, Anisa. 2010. *Robot Micromouse dengan Menggunakan Algoritma Depth-First Search*. (Online) (http://www.gunadarma.ac.id/library/articles/graduate/computer-science/2010/Artikel_21105199.pdf, diakses 8 Juni 2011).

Pololu. --. Pololu *3pi Robot User's Guide*. Texas : Pololu.