III
METODE PENELITIAN

Dalam penyusunan skripsi ini, dirancang suatu aplikasi pengolahan citra *digital* labirin yang dapat mengubah citra *digital* labirin menjadi *edge* dan *vertex*. Metode penelitian yang digunakan pada penyusunan skripsi ini adalah :

3.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi:

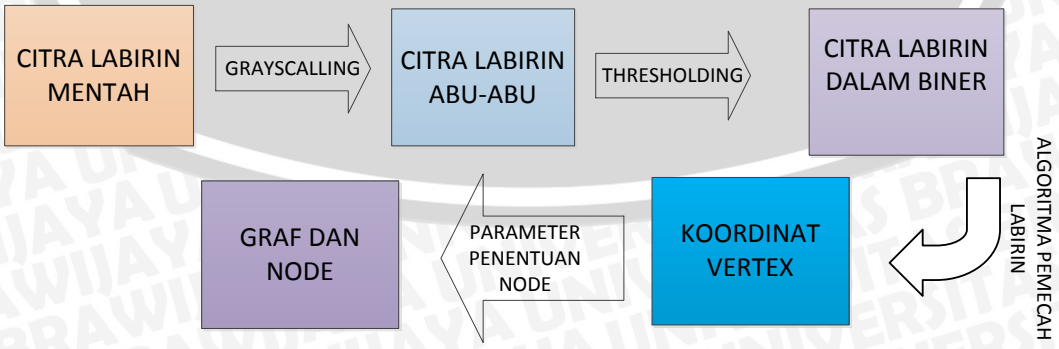
1. Membaca dan mempelajari buku – buku yang berhubungan dengan *image processing* dan *Maze Solution.*
2. Mempelajari algoritma pengolahan citra *digital* dan algoritma pendukung untuk memecah labirin.
3. Mempelajari teknik – teknik dasar pemrograman dengan menggunakan *Microsoft Visual Studio C#.*

3.2 Analisa Kebutuhan

Untuk menentukan sistem yang dapat digunakan untuk menjalankanaplikasi konversi citra labirin ke dalam *edge* dan *vertex*, antara lain:

a. Perangkat Keras :
    1. PC atau Laptop.
b. Perangkat Lunak :
    1. *Operating system Windows* 7.
    2. *Software Microsoft Visual C#* 2008.

3.3 Blok Diagram Sistem



Gambar 3.1 Blok Diagram Sistem

Keterangan :

CITRA LABIRIN MENTAH

Citra labirin adalah citra yang nantinya akan diolah sesuai proses pengolahan citra. Citra labirin berasal dari *file* atau hasil *scanner*, dapat berupa citra labirin berwarna atau hitam putih.

CITRA LABIRIN ABU-ABU

Citra labirin keabu-abu an adalah bentuk citra labirin hasil dari proses *grayscale*.

CITRA LABIRIN DALAM BINER

Citra labirin dalam biner adalah bentuk citra dalam bentuk biner 0 (hitam) dan 1 (putih). Hasil *output* ini diperoleh dari proses *binaryzation* atau *threshold*. Dalam proses *threshold* citra *digital* dianalisis untuk mendapatkan *value* tiap *pixel*. Setelah didapat *value* tiap *pixel*nya lalu dibandingkan dengan *threshold value*. Jika nilai *value* setiap *pixel*nya dibawah *threshold value* maka nilai *pixel* itu akan dirubah menjadi 0 (hitam) dan begitu juga sebaliknya.

KOORDINAT VERTEX

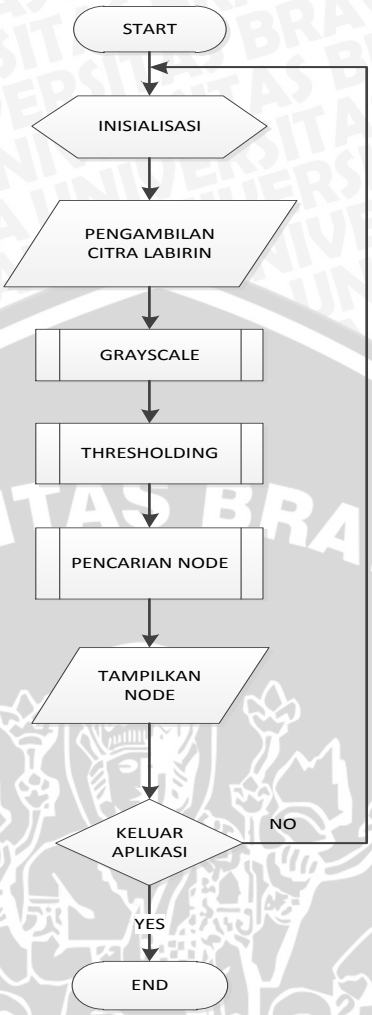
Setelah didapatkan citra labirin dalam bentuk biner, nantinya citra tersebut dipecah menggunakan algoritma yang telah dibentuk untuk mendapatkan nilai pada koordinat titik tengah pada lorong.

GRAF DAN NODE

Graph dan *Node* adalah hasil dari proses pencarian *node*. *Graph* berisi koordinat masing-masing node dan jarak antara *node*nya dengan struktur *graph* tidak berarah, berbobot Hasil *node* yang ditemukan, nantinya akan di *overlay* pada citra labirin dan disimpan dalam bentuk *plaintext*.

3.4 Abstraksi Sistem

Untuk membantu memahami abstraksi sistem, Berikut ini merupakan diagram alir dari proses pengolahan citra labirin menjadi *edge* dan *vertex*.



Gambar 3.2 *Flowchart* Keseluruhan Sistem

Pada tahap awal PC mengambil citra labirin dari *file* atau hasil *scanning* dan mem*filter*nya menjadi keabuan dengan proses *grayscale*. Setelah citra berwarna keabuan selanjutnya di *binaryzation* dengan proses *threshold*. Citra labirin hitam putih hasil *threshold* nantinya akan dipecah menggunakan algoritma yang telah dibuat untuk memperoleh *data vertex*, menghitung jarak antara *vertex* satu ke *vertex* yang lain dan menyimpannya pada media penyimpanan menggunakan struktur *data graph* dari sebuah citra labirin.

Pemecahan labirin dari citra hitam putih sampai kepada berbentuk *node* dan *graph* memerlukan beberapa tahap :

1. Mencari nilai lebar dinding dan jalan atau lorong citra labirin.

Dalam mencari nilai lebar dinding dan lorong dari citra labirin dimulai dari sisi sebelah kiri labirin. Pertama, *height* labirin dibagi dengan titik bagi yang

telah ditentukan . Pengecekan dimulai dari titik tadi dengan asumsi bahwa dinding labirin adalah hitam. Pengecekan dilakukan tiap *pixel* dari sisi sebelah kiri ke arah kanan, jika telah ketemu *pixel* warna putih maka nilai *pixel* hitam pertama akan disimpan dan pengecekan dilakukan pada *pixel* putih hingga bertemu *pixel* hitam. Pengecekan akan berakhir saat titik pencari nilai tadi bernilai hitam semua.

Setelah proses pertama maka akan ditemukan nilai dari lebar *pixel* hitam (dinding) dan nilai dari lebar *pixel* putih (lorong). Dari hasil masing-masing *pixel* hitam dan putih tersebut diambil nilai yang paling sering muncul dan digunakan sebagai lebar dinding dan lebar lorong dengan asumsi nilai lebar masing-masing dinding ataupun lorong sama.

2. Menentukan koordinat *vertex*.

Setelah didapatkan lebar dari masing-masing dinding dan lorong, selanjutnya adalah mencari letak *vertex* pada lorong labirin. Pencarian *vertex* adalah dengan mencari titik potong antara sumbu *x* dan *y* pada lorong labirin, dengan cara membuat garis acuan awal berdasarkan nilai lebar dinding ditambah setengah nilai lorong, setelah itu untuk garis selanjutnya adalah dengan menambahkan kombinasi nilai lebar dinding ditambah dengan nilai lebar lorong. Proses tersebut berakhir saat koordinat koordinat *width* maks = (*width* maks – (lebar dinding + ½ lebar lorong). Dan saat koordinat *height* maks = (*height* maks – (lebar dinding + ½ lebar lorong).

3. Mencari *node* dan jarak antara *node* atau *vertex*.

Setelah ditemukan koordinat masing-masing *vertex* maka hal selanjutnya yang dilakukan adalah dengan mengecek tiap *vertex*, apakah *vertex* tersebut termasuk *node* atau bukan. Jarak antara *vertex* satu dengan *vertex* yang lain dihitung dan disimpan jika masing-masing termasuk *node* dan saling berhubungan.

Pencarian *node* dimulai dari sisi sebelah kiri dan atas. Saat masuk koordinat lorong atau dengan *pixel* berwarna putih dilakukan pengecekan terhadap batas-batas dari koordinat tersebut dengan parameter seperti ditunjukkan pada tabel 3.1.

Tabel 3.1 Parameter Penentuan *Node*

Nomor Vertex	Batas Kiri	Batas Atas	Batas Kanan	Batas Bawah	Keterangan
1	Ada dinding	Ada dinding	Tidak	Tidak	Tikungan
2	Ada dinding	Ada dinding	Ada dinding	Tidak	Jalan buntu
3	Ada dinding	Ada dinding	Tidak	Ada dinding	Jalan buntu
4	Ada dinding	Tidak	Ada dinding	Ada dinding	Jalan buntu
5	Ada dinding	Tidak	Tidak	Ada dinding	Tikungan
6	Ada dinding	Tidak	Tidak	Tidak	Simpang Tiga
7	Tidak	Ada dinding	Ada dinding	Ada dinding	Jalan buntu
8	Tidak	Ada dinding	Ada dinding	Tidak	Tikungan
9	Tidak	Ada dinding	Tidak	Tidak	Simpang Tiga
10	Tidak	Tidak	Ada dinding	Ada dinding	Tikungan
11	Tidak	Tidak	Ada dinding	Tidak	Simpang Tiga
12	Tidak	Tidak	Tidak	Ada dinding	Simpang Tiga
13	Tidak	Tidak	Tidak	Tidak	Perempatan

Hasil dari pencarian *node* disimpan dalam bentuk *graf* dan *node – node* yang telah ditemukan di *overlay* kedalam citra labirin dengan warna *pixel* yang berbeda.

3.5 Pengujian Sistem

Pengujian dilakukan untuk menjamin dan memastikan bahwa aplikasi yang telah dirancang memiliki tingkat kesalahan yang kecil. Untuk mengetahui apakah aplikasi bekerja dengan baik dan sesuai dengan perancangan, maka diperlukan serangkaian pengujian. Pengujian aplikasi konversi citra labirin ke dalam *edge* dan *vertex* dilakukan terhadap beberapa sample citra labirin, baik dari file atau dari hasil *scanner*.

Hasil pengujian dari proses konversi citra nantinya akan dibandingkan dengan citra awal atau mentah yang belum diproses. Parameter yang menjadi tolak ukur adalah kualitas citra hasil pengujian, waktu yang dibutuhkan untuk memproses suatu citra labirin (dengan melakukan pengujian terhadap citra yang memiliki resolusi dan besar kapasitas *file* nya), serta hasil *edge* dan *vertex* dari proses pengolahan citra labirin.

Pengujian terhadap keluaran dari aplikasi konversi citra labirin kedalam *edge* dan *vertex* juga dilakukan secara *manual*, yaitu dengan membandingkan jumlah dan letak *vertex* yang dihasilkan oleh aplikasi pengolahan citra labirin dengan perhitungan jumlah dan letak *vertex* secara manual oleh user.

3.6 Kesimpulan dan Saran

Pada tahap ini, diambil dari hasil pengujian dan analisa terhadap Aplikasi Konversi Citra Labirin kedalam *Edge* dan *Vertex*. Tahap Selanjutnya adalah membuat saran untuk perbaikan terhadap penelitian selanjutnya sehinga dapat menyempurnakan kekurangan-kekurangan yang ada.