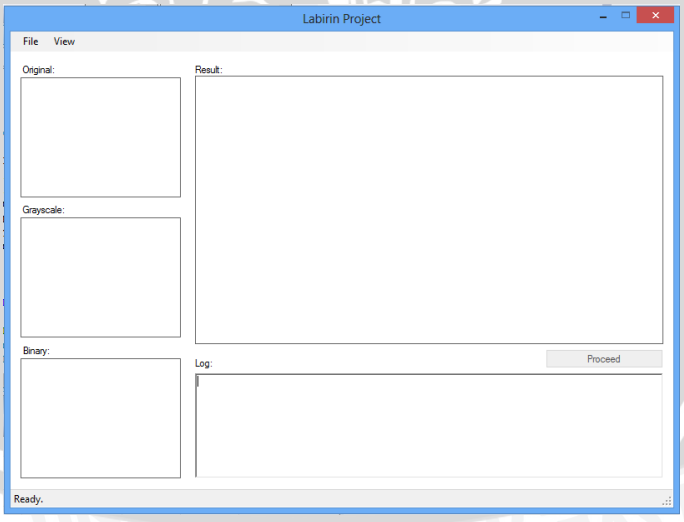
BAB V
PENGUJIAN

Rancangan yang telah diimplementasikan dalam bentuk nyata memerlukan suatu pengujian. Uji coba ini adalah untuk mengetahui hasil dari percobaan yang dilakukan sekaligus sebagi sarana pemunculan ide ide bagi proses pengembangan selanjutnya. Pengujian yang dilakukan dalam bab ini adalah sebagai berikut :

1. Pengujian terhadap beberapa jenis labirin dengan ukuran dan resolusi yang berbeda-beda.
2. Melakukan perhitungan node secara manual dan membandingkannya dengan jumlah *node* yang terdeteksi oleh aplikasi.
3. Melakukan *test* terhadap hasil keluaran aplikasi dengan mencoba melakukan l*oad* kembali untuk melihat apakah aplikasi yang dibuat sesuai.

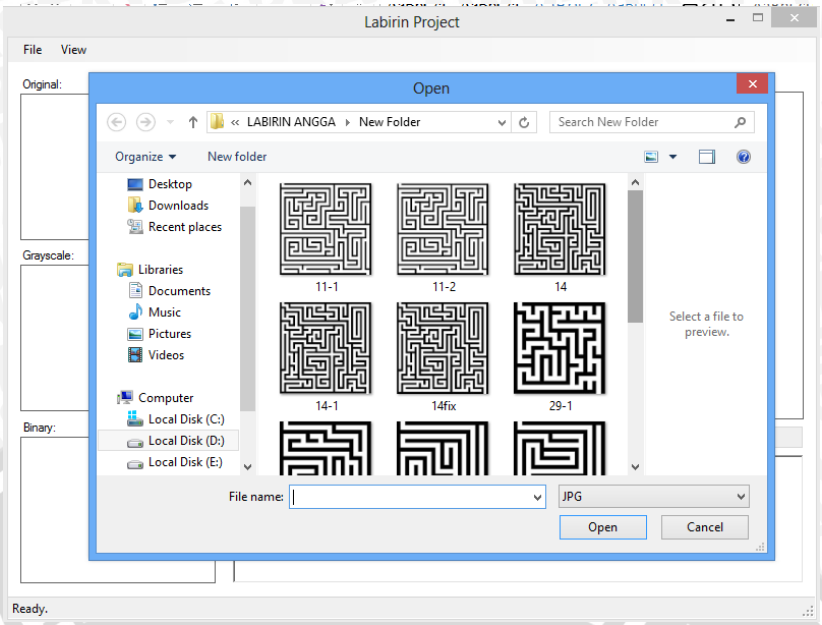
5.1 Prosedur Pengujian

1. Dengan menghitung jumlah *node* secara manual pada labirin, tandai persimpangan dan jalan buntu yang ditemukan, serta mencatat jumlah keseluruhan *node*.
2. Menjalankan aplikasi yang telah dibuat, seperti ditunjukkan oleh gambar 5.1.



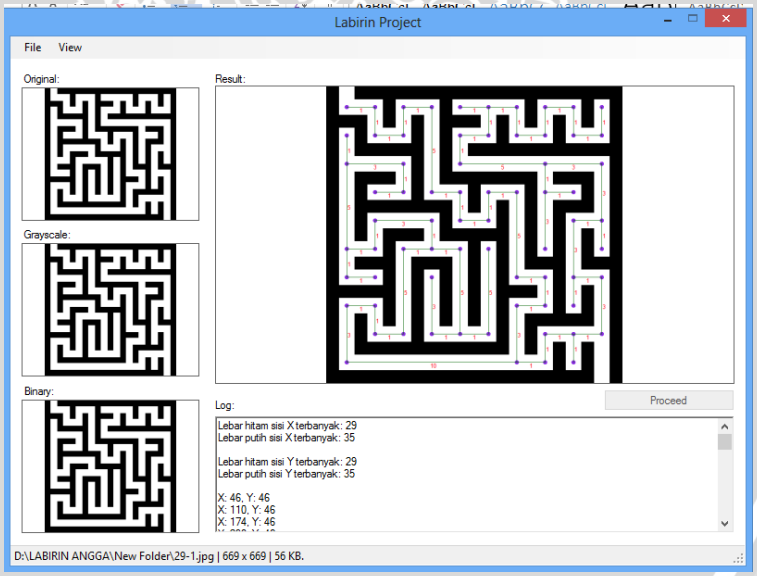
Gambar 5.1 Tampilan *user interface* program

3. Pilih *menu strip file* kemudian pilih pada *tool open* untuk memilih citra labirin yang akan di proses. Seperti ditunjukkan oleh gambar 5.2.



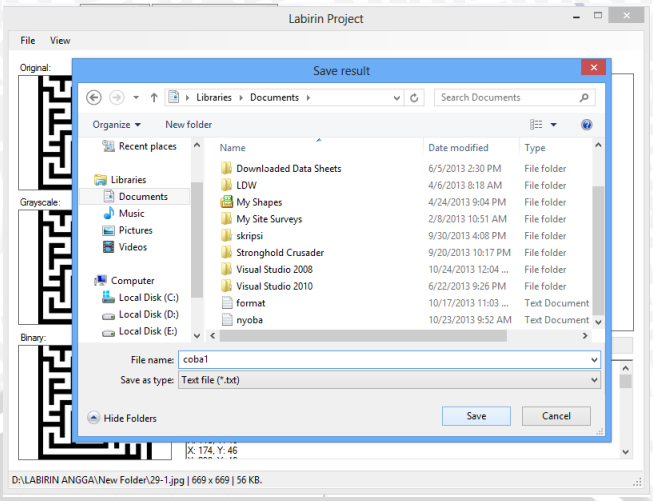
Gambar 5.2 Tampilan *user interface* saat melakukan proses *load* citra labirin.

4. Pilih *button* proses untuk memulai melakukan pengolahan citra. Seperti ditunjukkan oleh gambar 5.3.



Gambar 5.3 Tampilan *user interface* saat proses pengolahan telah selesai.

5. Simpan hasil dari pengolahan citra labirin kedalam *plaintext*, serta melakukan percobaan terhadap jenis labirin yang lainnya. Seperti ditunjukkan oleh gambar 5.4.



Gambar 5.4 Tampilan *user interface* saat proses penyimpanan hasil pengolahan program.

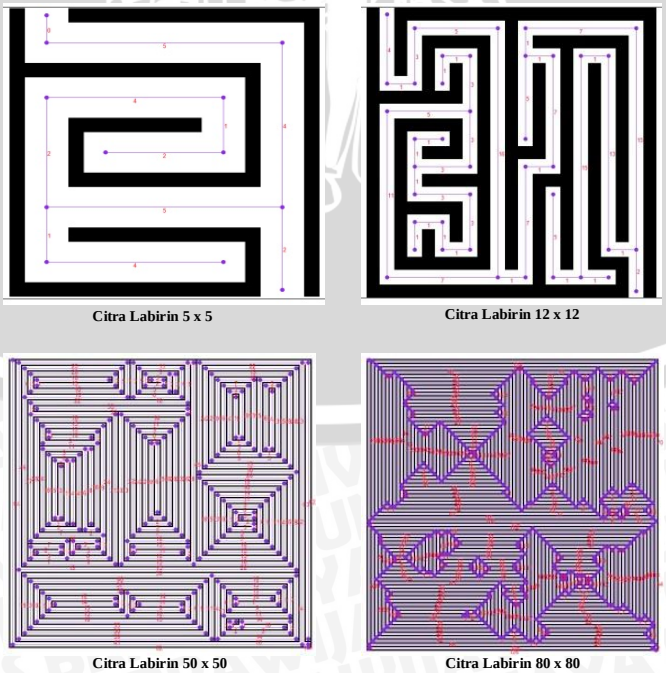
6. Melakukan proses *load* dari *file graph* hasil dari pengolahan citra sebelumnya untuk melihat apakah program berjalan sebagaimana mestinya.

5.2 Pengujian

Pengujian dilakukan terhadap beberapa tipe labirin, seperti :

A. Citra Labirin Biasa

Citra labirin biasa adalah bentuk citra labirin sempurna yang langsung dimulai dari dinding tanpa adanya *border* diluar citra labirin. Pengujian pada citra labirin biasa dilakukan kepada 20 sampel citra labirin yang mempunyai bentuk labirin dan besar *file* yang berbeda – beda. Contoh citra labirin yang diuji seperti ditunjukkan oleh gambar 5.5.



Citra Labirin 5 x 5 **Citra Labirin 12 x 12**

Citra Labirin 50 x 50 **Citra Labirin 80 x 80**

Gambar 5.5 Contoh beberapa citra labirin yang diuji

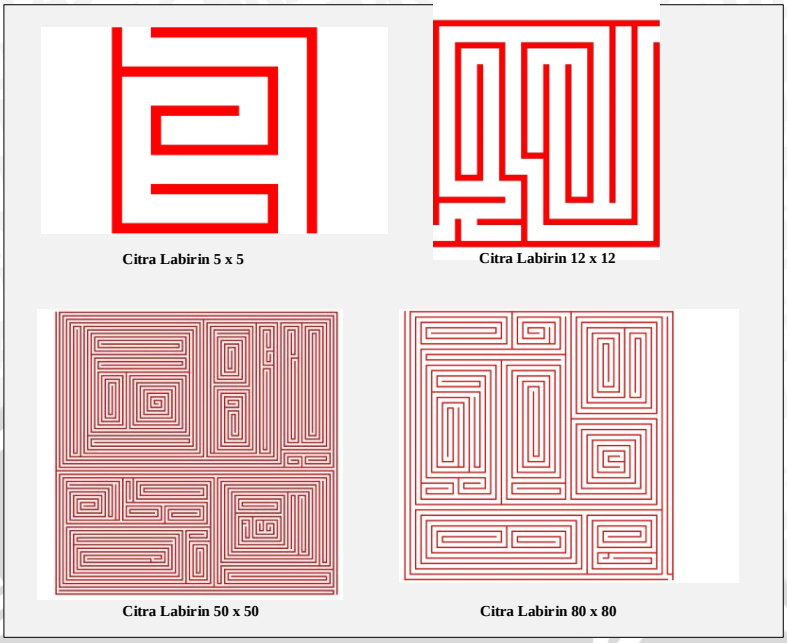
Dari hasil pengujian pada citra labirin biasa, program yang dibuat dapat mendeteksi *node* dengan baik dan sesuai dengan perhitungan manual. Tetapi saat menangani citra labirin besar dengan nilai dinding dan lorong yang sangat kecil, algoritma yang dibuat membutuhkan waktu yang cukup lama karena terlalu banyak *node* yang diseleksi serta pengecekkan nilai *edge* masing-masing *node* . Hasil pengujian seperti ditunjukkan pada tabel 5.1.

Tabel 5.1. Hasil pengujian citra labirin biasa

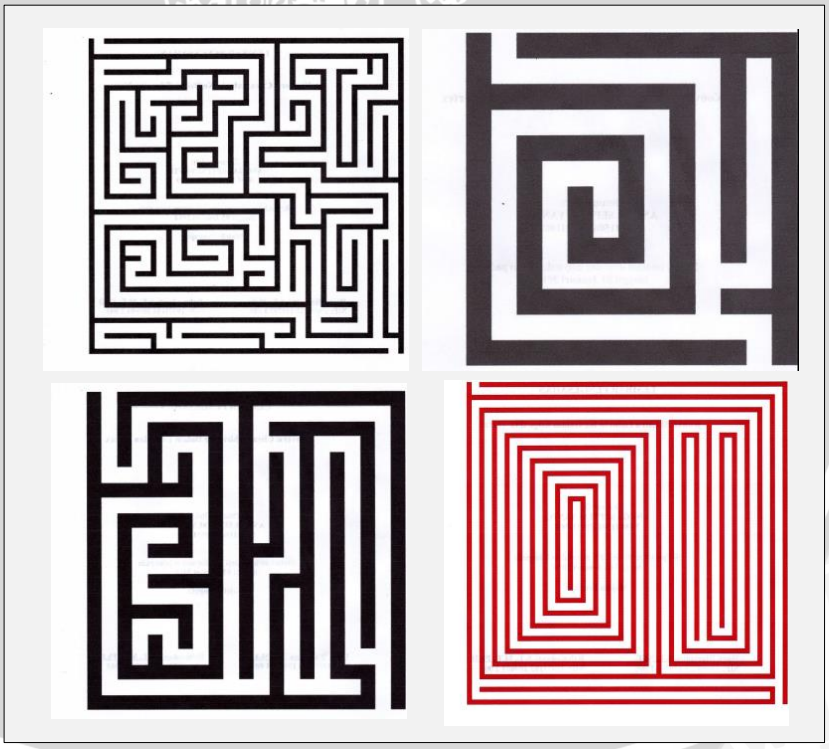
No	Labirin	Jumlah *node* program	Hasil perhitung-an manual
1	5 x 5	12	12
2	7 x 7	16	16
3	8 x 8	18	18
4	11 x 11	72	72
5	12 x 12	40	40
6	21 x 21	150	150
7	22 x 22	186	186
8	25 x 25	55	55
9	27 x 27	61	61
10	33 x 33	101	101
11	38 x 38	105	105
12	41 x 41	130	130
13	47 x 47	154	154
14	50 x 50	250	250
15	60 x 60	289	289
16	70 x 70	372	372
17	80 x 80	478	478
18	90 x 90 (3)	6251	577
19	90 x 90 (2)	8100	577
20	100 x 100	6732	535

B. Citra Labirin Biasa Dengan *Frame*

Citra labirin dengan *frame* adalah salah satu bentuk citra labirin yang memerlukan metode khusus, karena dalam penanganannya harus terlebih dahulu mendeteksi *frame* diluar citra labirin baru kemudian mulai pemecahan dinding dan jalan labirin. Pengujian dilakukan kepada 20 sampel citra labirin yang mempunyai bentuk labirin dan *frame* yang berbeda – beda, baik dari file maupun citra hasil dari *scanner*.



Gambar 5.6 Contoh beberapa citra labirin dengan *frame* yang diuji



Gambar 5.7 Contoh beberapa citra labirin hasil dari *scanner* yang diuji

Dari hasil pengujian pada citra labirin dengan *frame*, program yang dibuat dapat mendeteksi *node* dengan baik dan sesuai dengan perhitungan manual. Tetapi saat menangani citra labirin besar dengan nilai dinding dan lorong yang sangat kecil, algoritma yang dibuat membutuhkan waktu yang cukup lama

karena terlalu banyak *node* yang diseleksi serta pengecekkan nilai edge masing-masing *node*.

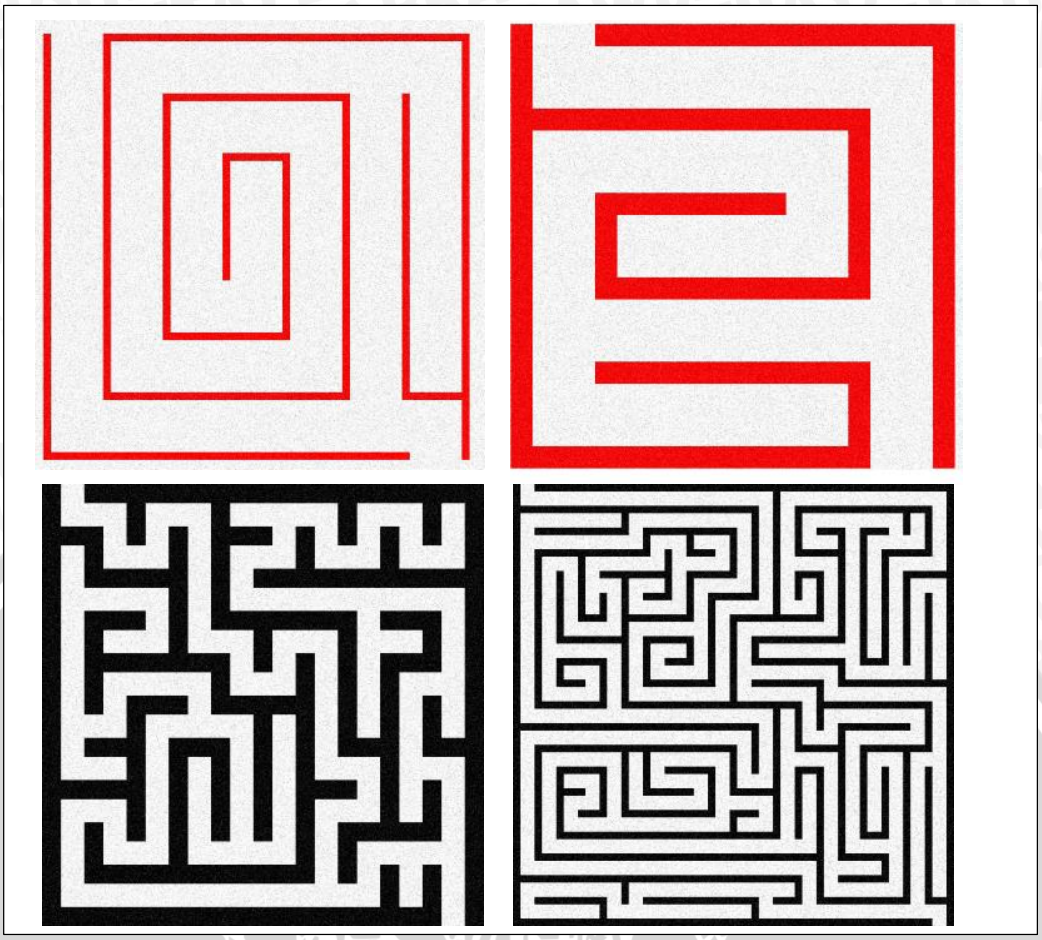
Pada pengujian terhadap citra labirin hasil dari *scanner* terdapat beberapa hasil pengolahan yang tidak sesuai dengan perhitungan manual dikarenakan pada citra labirin hasil *scanner* memiliki *noise* yang menimbulkan kesalahan dalam proses pencarian nilai dinding dan lorong, serta posisi citra hasil *scanner* yang tidak simetris dan tegak lurus. Hasil pengujian seperti ditunjukkan pada tabel 5.2.

Tabel 5.2. Hasil pengujian citra labirin dengan *frame*

No	Labirin	Jumlah *node* program	Hasil perhitung-an manual
1	5 x 5	12	12
2	7 x 7 (*scanner*)	16	16
3	8 x 8	48	48
4	11 x 11 (*scanner*)	98	96
5	12 x 12	31	31
6	21 x 21 (*scanner*)	69	69
7	22 x 22	119	119
8	25 x 25 (*scanner*)	55	55
9	27 x 27	61	61
10	33 x 33 (*scanner*)	109	101
11	38 x 38	105	105
12	41 x 41 (*scanner*)	130	130
13	47 x 47	154	154
14	50 x 50	250	250
15	60 x 60	289	289
16	70 x 70	372	372
17	80 x 80	478	478
18	90 x 90 (3)	6251	577
19	90 x 90 (2)	8100	577
20	100 x 100	6732	535

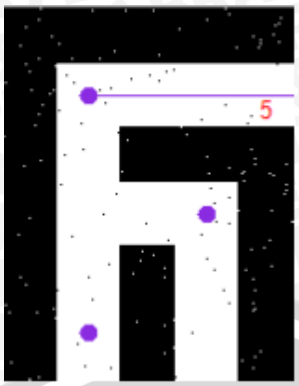
C. Citra Labirin Dengan *Noise*

Citra labirin dengan *noise* adalah salah satu bentuk citra labirin yang didalamnya terdapat gangguan – gangguan baik pada *frame*, dinding maupun lorong sehingga dalam pengolahannya sering menimbulkan kesalahan pada hasil dari pengolahan citra. Pengujian dilakukan kepada 20 citra labirin yang diberi *gaussian noise* secara acak.

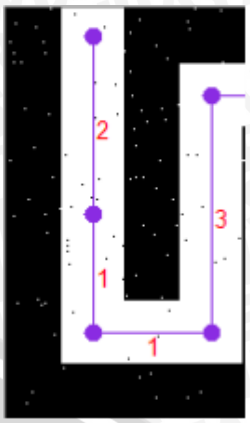


Gambar 5.8 Citra labirin dengan *Gaussian noise*

Dari hasil pengujian pada citra labirin dengan *Gaussian noise* secara acak pada citra labirin, program yang dibuat dapat menangani *noise* serta mendeteksi *node* yang *valid*. Tetapi terdapat beberapa kondisi *noise* pada lorong yang menyebabkan jumlah dan letak *node* tidak sesuai dengan yang diinginkan. Kondisi *noise* tersebut adalah saat letak *noise* berada tepat di *meetpoint* atau di titik *node* yang akan diseleksi serta saat ada *noise* yang terdapat di garis *edge* antar *vertex* pada lorong yang menyebabkan terbentuknya *node* baru dan membuat *edge* antar *node* menjadi berbeda. Kondisi *noise* pada lorong yang menyebabkan kesalahan *node* dan *edge* seperti ditunjukkan pada gambar 5.9.



***Noise* pada lorong**



Noise* pada *meetpoint

Gambar 5.9 Kondisi *noise* pada lorong yang menyebabkan kesalahan

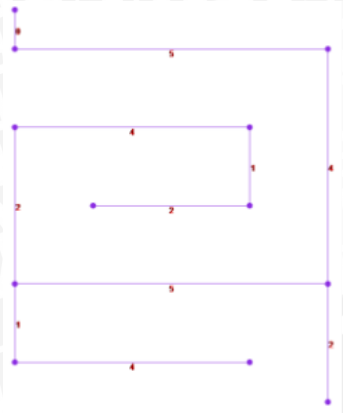
Untuk hasil pengujian citra labirin dengan *Gaussian noise* seperti ditunjukkan pada tabel 5.3.

Tabel 5.3. Hasil pengujian citra labirin dengan *gaussian noise.*

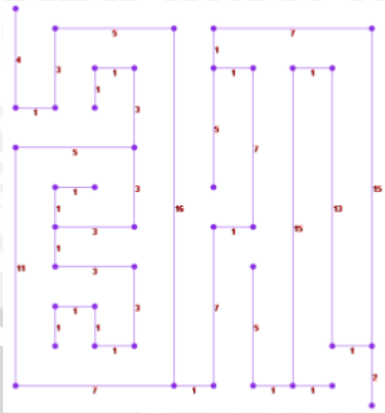
No	Labirin	Jumlah *node* program	Hasil perhitung-an manual
1	676 x 694 px	12	12
2	700 x 700 px	16	16
3	669 x 669 px	72	72
4	651 x 651 px	150	150
5	700 x 700 px	55	55
6	750 x 750 px	103	102
7	898 x 857 px	130	130
8	878 x 705 px	250	250
9	756 x 705 px	289	289
10	863 x 897 px	574	478
11	679 x 704 px	12	12
12	709 x 758 px	16	16
13	669 x 669 px	72	72
14	669 x 655 px	150	150
15	704 x 710 px	55	55
16	745 x 750 px	103	102
17	898 x 857 px	130	130
18	1057 x 705 px	250	250
19	756 x 705 px	289	289
20	863 x 897 px	574	478

D. Hasil Visualisasi *Graph*

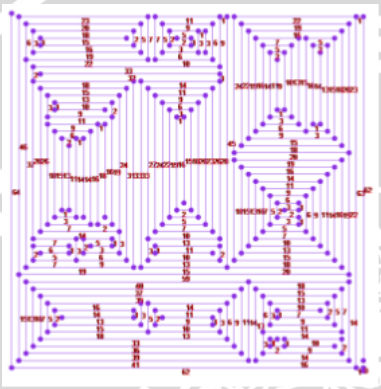
Citra labirin yang telah diproses akan di representasikan kedalam bentuk *graph* yang didalamnya terdapat *vertex* atau *node* serta *edge* masing-masing *vertex* yang terhubung beserta nilai *edge*. Bentuk representasi hasil pengolahan citra kedalam bentuk *graph* seperti ditunjukkan oleh gambar 5.10.



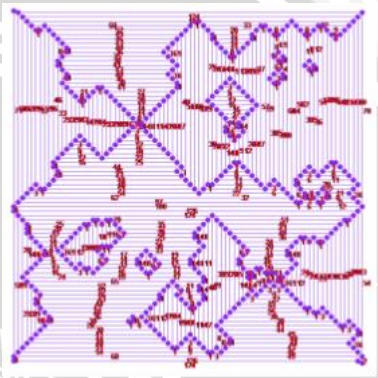
Graph labirin 5 x 5



Graph labirin 12 x 12



Graph labirin 50 x 50



Graph labirin 80 x 80

Gambar 5.10 Representasi *graph* hasil pengolahan citra

E. Pengujian Hasil Output Program

Dalam pengujian hasil *output* program adalah dengan melakukan *load* hasil pengolahan ke dalam aplikasi serta membandingkan dengan citra labirin awal. Pengujian hasil *output* program digunakan untuk mengetahui apakah hasil *output* program sesuai dengan yang diinginkan. Hasil pengujian seperti pada tabel 5.4.

Tabel 5.4. Hasil pengujian *output* program

No	Pengujian	Jumlah Output	Hasil Node
1	Labirin biasa	20	*valid*
2	Labirin dengan *frame*	20	Terdapat 2 *error* dari *output* labirin pada citra *scanner* karena terdapat *noise* dan posisi citra labirin yang tidak simetris serta tegak lurus.
3	Labirin dengan *Gaussian noise*	20	Terdapat 4 *error* dari hasil *output* karena *noise* pada lorong terlalu besar
	Total	60	6 *error*

Dari hasil pengujian *output* program terdapat 6 *error*, yaitu perbedaan jumlah *node* yang dihasilkan. *Error* yang terjadi pada bentuk citra labirin yang mempunyai *noise* yang banyak atau besar pada lorong labirin yang mengakibatkan hasil *output* tidak sesuai, serta bentuk citra dari hasil *scanner* yang tidak simetris dan tegak lurus

sehingga menyebabkan nilai dinding serta lorong yang tidak *valid*, tetapi pada citra labirin biasa, *frame* serta dengan *noise* yang tidak terlalu besar, program yang telah dibuat masih dapat mendeteksi *edge* dan *vertex* dengan baik.