BAB IV

PENGUJIAN SISTEM

Pengujian sistem yang dilakukan merupakan pengujian terhadap perangkat keras dan perangkat lunak dari *Micromouse Robot* dan aplikasi pada PC yang telah selesai dibuat.

4.1 Pengujian Sensor Reflektansi IR

4.1.1 Tujuan

Tujuan pengujian sensor reflektansi IR untuk mengetahui perubahan nilai tegangan *output* sensor terhadap perubahan jarak sebagai indikasi sensor berfungsi dengan baik atau tidak.

4.1.2 Alat yang Digunakan

1. Catu data + 5V.
2. Multimeter.
3. Lintasan labirin.

4.1.3 Prosedur Pengujian

1. Hubungkan catu daya pada sensor reflektansi IR.
2. Dengan menggunakan multimeter, catat tegangan *output* sensor reflektansi IR ketika mengubah-ubah jarak antara dinding labirin dengan sensor.

4.1.4 Hasil Pengujian

Hasil dari pengujian ini adalah sensor berfungsi dengan baik, ditandai dengan perubahan tegangan *output* sensor reflektansi IR yang sebanding dengan perubahan pada jarak antara dinding dengan masing-masing sensor. Hasil dari pengukuran tegangan *output* pada masing-masing sensor reflektansi IR dengan menggunakan multimeter dapat ditunjukkan seperti pada tabel 4.1.

Tabel 4.1 Hasil pengukuran tegangan masing-masing sensor reflektansi IR..

Jarak (cm)	Tegangan sensorv(V)				
	0	1	2	3	4
1	3,10	3,55	4,33	3,15	3,12
2	3,06	3,49	4,33	3,11	3,08
3	2,07	2,53	4,33	2,12	2,09
4	1,55	2,01	4,33	1,60	1,57
5	1,09	1,47	4,33	1,14	1,11
6	0,85	1,22	4,03	0,90	0,87
7	0,64	0,98	3,12	0,69	0,66
8	0,53	0,79	2,67	0,58	0,55
9	0,45	0,70	2,31	0,50	0,47
10	0,39	0,62	2,04	0,44	0,41
11	0,34	0,55	1.53	0,39	0,36
12	0,30	0,50	1,27	0,35	0,33
13	0,27	0,44	1,05	0,32	0,29
14	0,25	0,41	0,97	0,30	0,27
15	0,23	0,38	0,89	0,28	0,25
16	0,21	0,36	0,81	0,26	0,23
17	0,20	0,34	0,73	0,25	0,22
18	0,19	0,33	0,65	0,24	0,21
19	0,18	0,32	0,57	0,23	0,20
20	0,17	0,31	0,51	0,22	0,19

Dari hasil pengukuran tegangan *output* pada masing-masing sensor dapat diketahui bahwa semakin dekat jarak sensor dengan dinding maka tegangan *output* sensor akan semakin besar dan sebaliknya.

4.2 Pengujian Aplikasi pada PC sebagai Pencari *Node*.

4.2.1 Tujuan

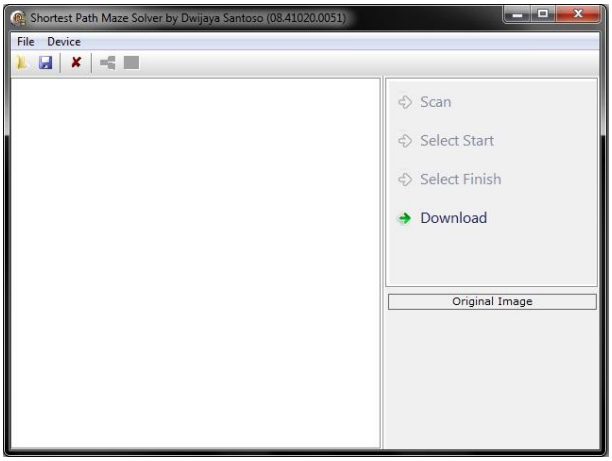
Tujuan pengujian aplikasi pada PC sebagai pencari *node* untuk mengetahui kesesuaian jumlah *node* yang dapat dideteksi oleh aplikasi dengan jumlah *node* yang dihitung secara manual.

4.2.2 Alat yang Digunakan

1. Lintasan labirn dengan 5 bentuk berbeda.
2. *Webcam*.
3. Seperangkat PC yang sudah terdapat aplikasi pencari rute terpendek.

4.2.3 Prosedur Pengujian

1. Dengan menghitung *node* secara manual pada labirin, tandai persimpangan dan jalan buntu yang ditemukan, catat jumlah keseluruhan *node*.
2. Hubungkan kabel *Universal Serial Bus* (USB) *webcam* dengan PC.
3. Jalankan aplikasi pencari rute terpendek pada PC yang tampilannya dapat ditunjukkan seperti pada gambar 4.1.

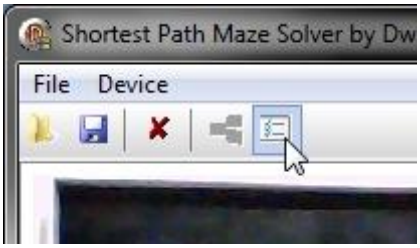


Gambar 4.1 Tampilan program pengolahan citra labirin dan pencarian *node*.

4. Masuk ke modus kamera, pada menu pilih *Device*, kemudian pilih *webcam* yang digunakan untuk pengambilan citra labirin seperti yang ditunjukkan pada gambar 4.2.

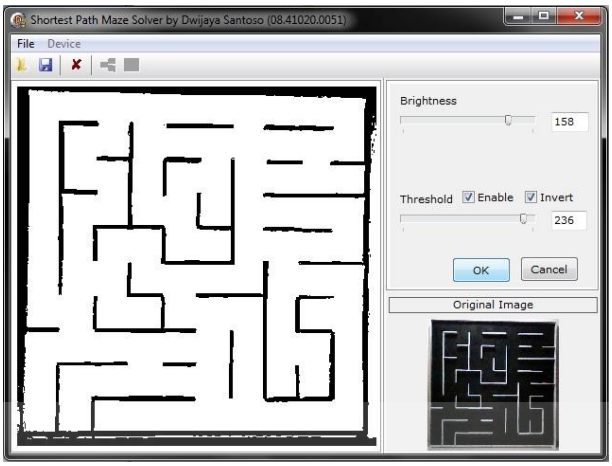
Gambar 4.2 Menu pemilihan *webcam* yang digunakan untuk pengambilan citra labirin.

5. Pilih *setting icon* untuk melakukan pengaturan pengolahan citra yang dapat ditunjukkan seperti pada gambar 4.3.



Gambar 4.3 *Setting icon.*

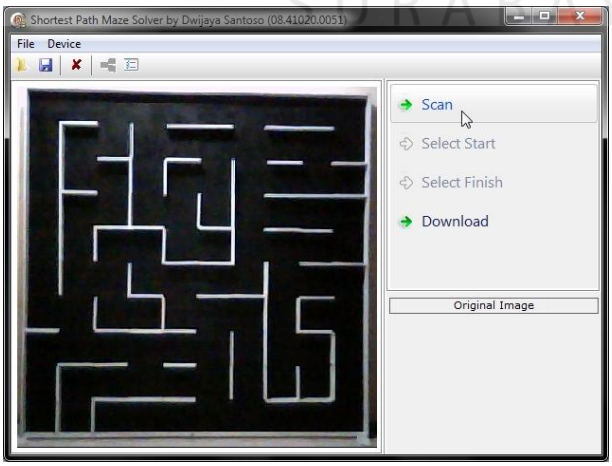
Kemudian lakukan pengaturan untuk mendapatkan citra yang sesuai yaitu lintasan labirin berwarna putih dan dinding labirin berwarna hitam yang tampilannya dapat ditunjukkan seperti pada gambar 4.4.



Gambar 4.4 Tampilan pengaturan pengolahan citra.

Setelah selesai pengaturan, pilih tombol OK untuk menyimpan pengaturan atau *Cancel* untuk membatalkan, dan masuk kembali ke modus kamera.

6. Lakukan pengambilan citra labirin dengan posisi dinding tepian labirin sejajar dengan sisi *window* kamera dan memulai pencarian node dengan menekan tombol *Scan* seperti yang ditunjukkan pada gambar 4.5.



Gambar 4.5 Tombol Scan untuk mengambilan citra dan memulai proses pencarian *node*.

7. Pengujian pencarian *node* dilakukan pada masing-masing bentuk labirin.

4.2.4 Hasil Pengujian

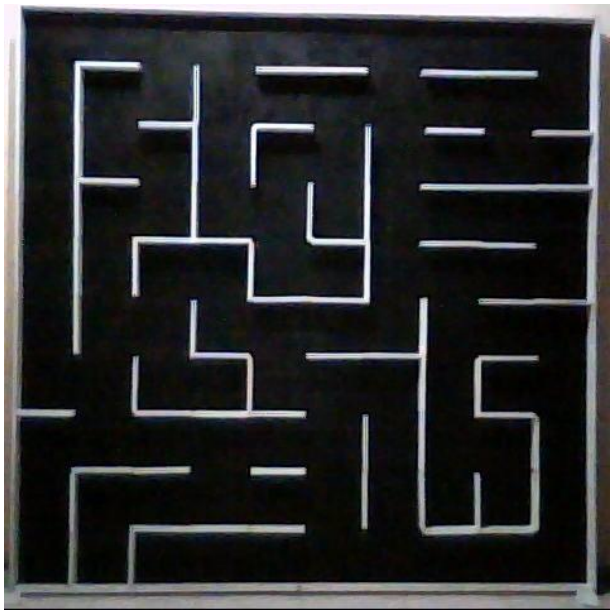
Hasil pengujian ini adalah jumlah *node* yang dapat dideteksi oleh aplikasi pada 6 bentuk labirin, jumlah yang terdeteksi pada 4 bentuk labirin diantaranya sesuai dengan menghitung *node* labirin secara manual dan 1 bentuk diantaranya mengalami *error* saat proses pencarian. Hasil pengujian aplikasi dan penghitungan *node* secara manual pada labirin dapat ditunjukkan seperti pada tabel 4.2.

Tabel 4.2 Hasil pengujian aplikasi dan penghitungan *node* secara manual pada labirin.

Bentuk labirin ke-	Jumlah *node* (manual)	Jumlah *node* (aplikasi)	Keterangan
1	38	38	Sesuai
2	40	40	Sesuai
3	45	45	Sesuai
4	45	0 (*error*)	Tidak sesuai
6	73	73	Sesuai

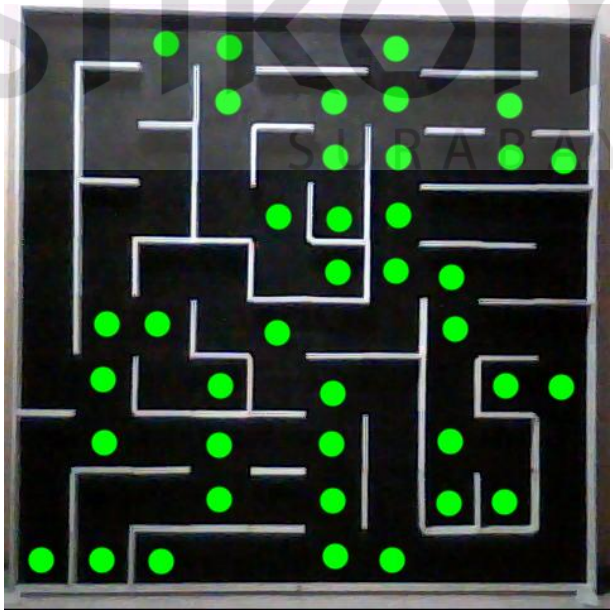
A. Labirin Bentuk ke-1

Citra yang didapat pada labirin bentuk ke-1 dapat ditunjukkan pada gambar 4.6.



Gambar 4.6 Citra labirin bentuk ke-1.

Dengan menghitung secara manual pencarian *node* pada setiap *node* yang ditemukan ditandai dengan lingkaran berwarna hijau yang seluruhnya berjumlah 38 yang dapat ditunjukkan seperti pada gambar 4.7.



Gambar 4.7 Citra labirin bentuk ke-1 yang sudah ditandai secara manual.

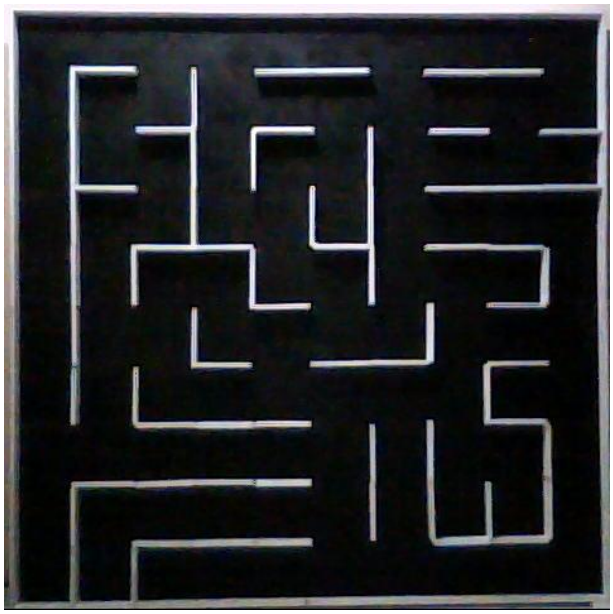
Dengan menggunakan aplikasi pada PC pencarian *node* pada citra labirin bentuk ke-1, *node* yang ditemukan ditandai dengan persegi berwarna hijau yang seluruhnya berjumlah 38 yang dapat ditunjukkan seperti pada gambar 4.8.

Gambar 4.8 *Node* yang dapat dideteksi pada citra labirin bentuk ke-1 menggunakan aplikasi pada PC.

Dari gambar 4.7 dan 4.8, pada labirin bentuk ke-1 dapat diketahui jumlah node yang didapat melalui perhitungan manual sesuai dengan jumlah node yang terdeteksi oleh aplikasi.

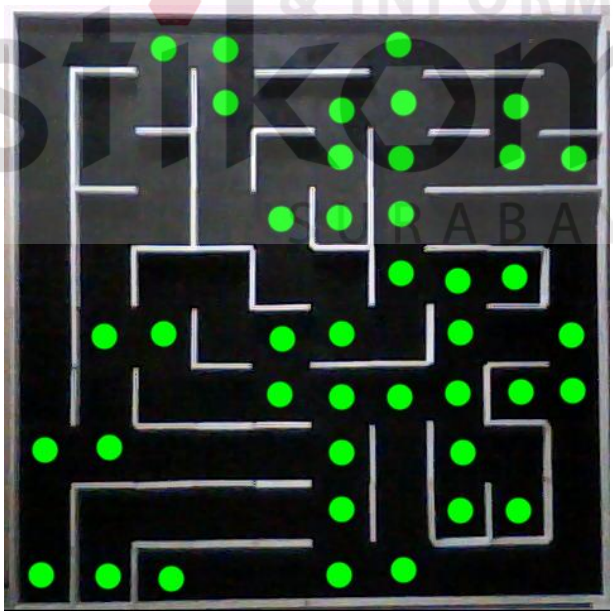
B. Labirin Bentuk ke-2

Citra yang didapat pada labirin bentuk ke-2 dapat ditunjukkan pada gambar 4.9.



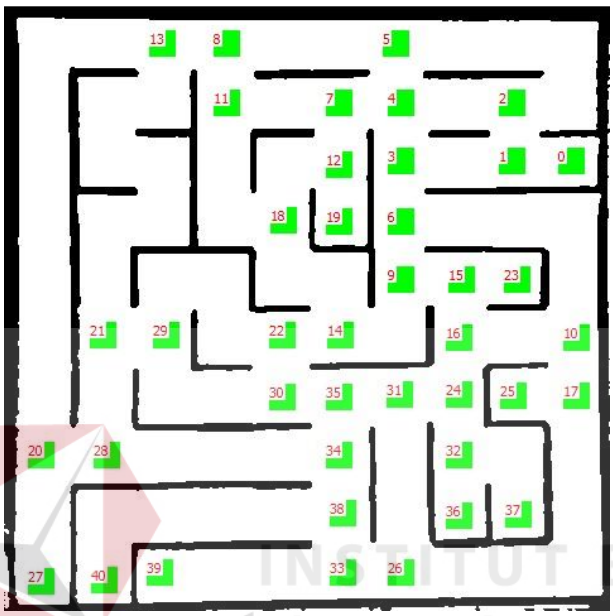
Gambar 4.9 Citra labirin bentuk ke-2.

Dengan menghitung secara manual pencarian *node* pada setiap *node* yang ditemukan ditandai dengan lingkaran berwarna hijau yang seluruhnya berjumlah 40 yang dapat ditunjukkan seperti pada gambar 4.10.



Gambar 4.10 Citra labirin bentuk ke-2 yang sudah ditandai secara manual.

Dengan menggunakan aplikasi pada PC pencarian *node* pada citra labirin bentuk ke-2, *node* yang ditemukan ditandai dengan persegi berwarna hijau yang seluruhnya berjumlah 40 yang dapat ditunjukkan seperti pada gambar 4.11.

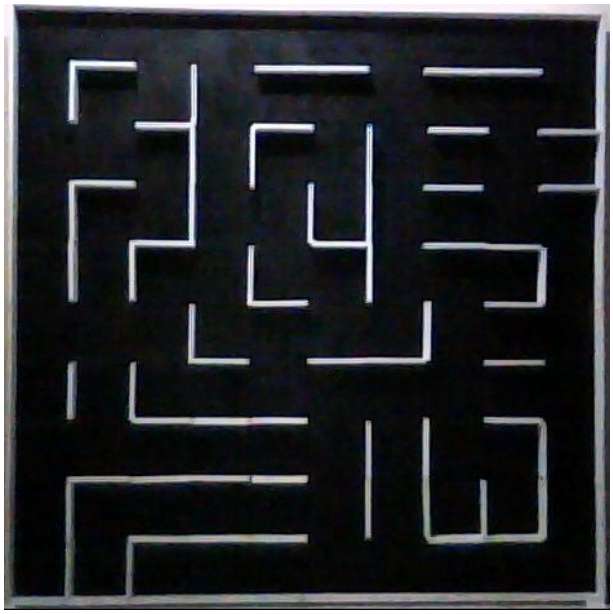


Gambar 4.11 *Node* yang dapat dideteksi pada citra labirin bentuk ke-2 menggunakan aplikasi pada PC.

Dari gambar 4.10 dan 4.11, pada labirin bentuk ke-2 dapat diketahui jumlah node yang didapat melalui perhitungan manual sesuai dengan jumlah node yang terdeteksi oleh aplikasi.

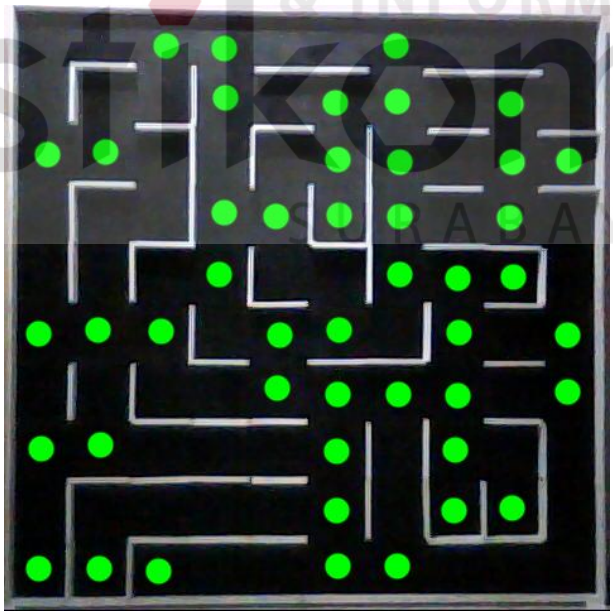
C. Labirin Bentuk ke-3

Citra yang didapat pada labirin bentuk ke-3 dapat ditunjukkan pada gambar 4.12.



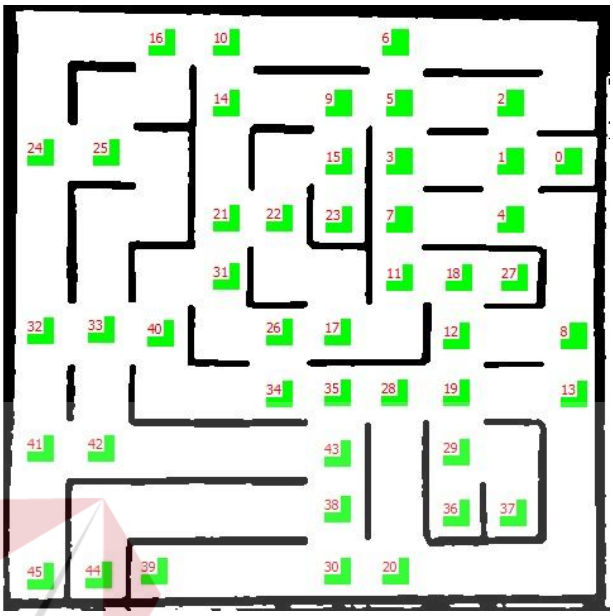
Gambar 4.12 Citra labirin bentuk ke-3.

Dengan menghitung secara manual pencarian *node* pada setiap *node* yang ditemukan ditandai dengan lingkaran berwarna hijau yang seluruhnya berjumlah 45 yang dapat ditunjukkan seperti pada gambar 4.13.



Gambar 4.13 Citra labirin bentuk ke-3 yang sudah ditandai secara manual.

Dengan menggunakan aplikasi pada PC pencarian *node* pada citra labirin bentuk ke-3, *node* yang ditemukan ditandai dengan persegi berwarna hijau yang seluruhnya berjumlah 45 yang dapat ditunjukkan seperti pada gambar 4.14.

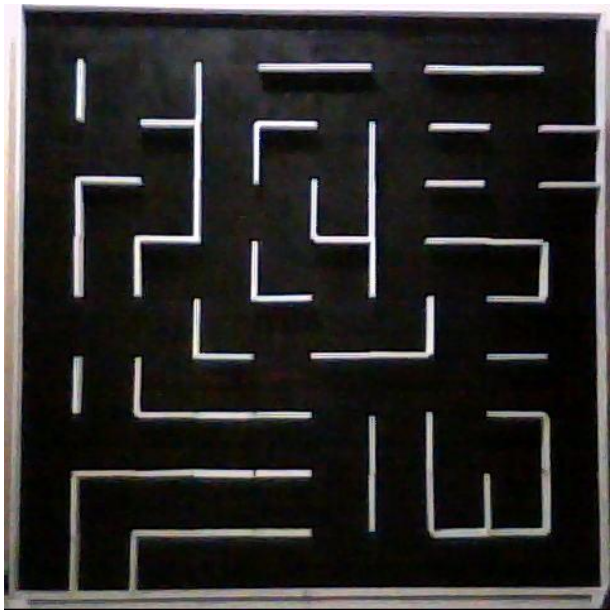


Gambar 4.14 *Node* yang dapat dideteksi pada citra labirin bentuk ke-3 menggunakan aplikasi pada PC.

Dari gambar 4.13 dan 4.14, pada labirin bentuk ke-3 dapat diketahui jumlah *node* yang didapat melalui perhitungan manual sesuai dengan jumlah *node* yang terdeteksi oleh aplikasi.

D. Labirin Bentuk ke-4

Citra yang didapat pada labirin bentuk ke-4 dapat ditunjukkan pada gambar 4.15.

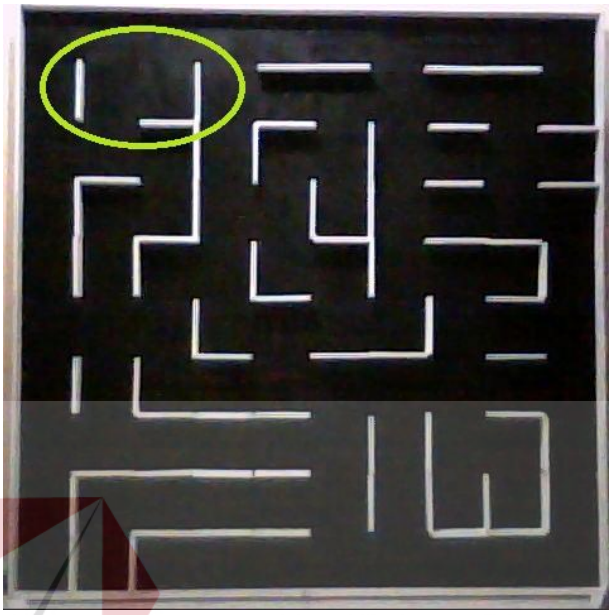


Gambar 4.15 Citra labirin bentuk ke-4.

Dengan menghitung secara manual pencarian *node* pada setiap *node* yang ditemukan ditandai dengan lingkaran berwarna hijau yang seluruhnya berjumlah 45 yang dapat ditunjukkan seperti pada gambar 4.16.

Gambar 4.16 Citra labirin bentuk ke-4 yang sudah ditandai secara manual.

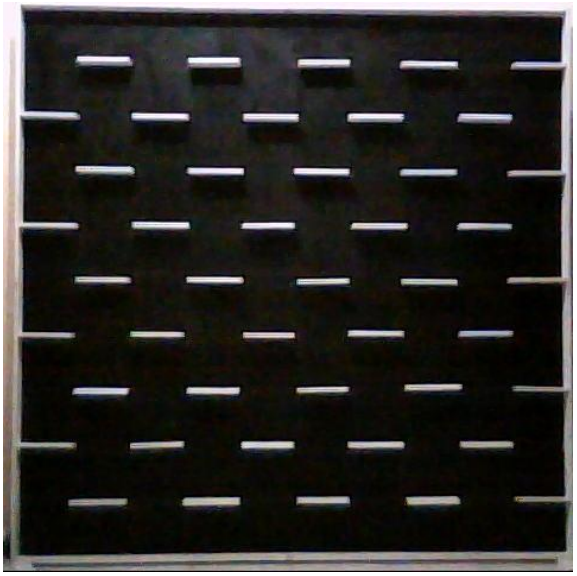
Dengan menggunakan aplikasi pada PC pencarian *node* pada citra labirin bentuk ke-4 mengalami *error* disebabkan adanya lebar lintasan yang melebihi 1 *cell* yang dapat ditunjukkan pada lingkaran hijau seperti pada gambar 4.17.



Gambar 4.17 Labirin bentuk ke-4 yang terdapat lebar jalur lebih dari 1 *cell*.

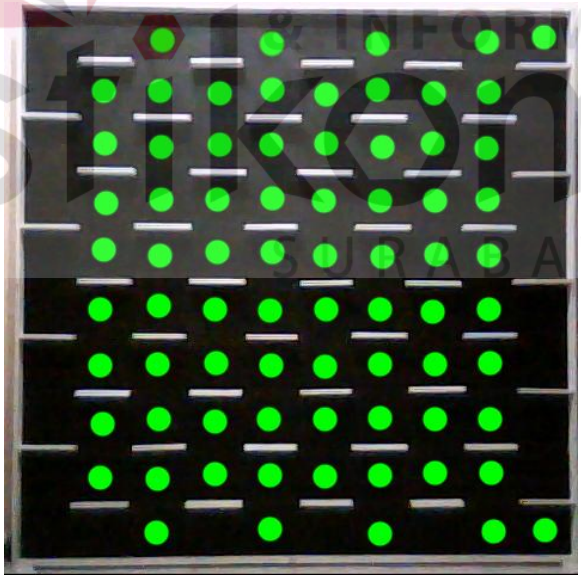
E. Labirin Bentuk ke-5

Citra yang didapat pada labirin bentuk ke-5 dapat ditunjukkan pada gambar 4.18.



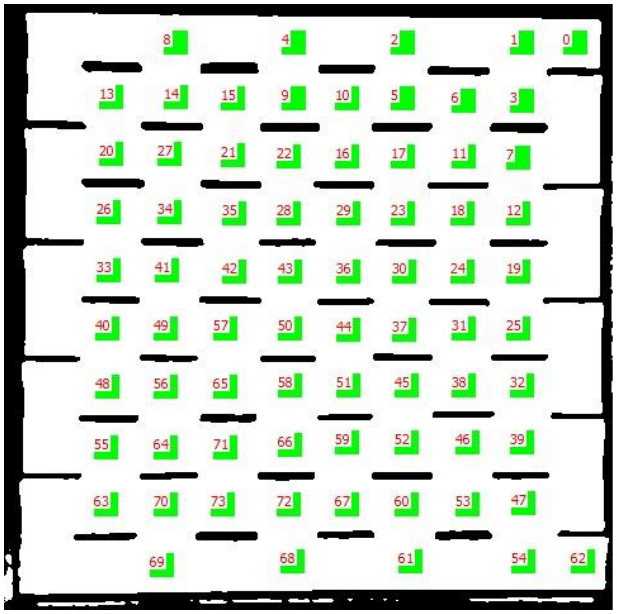
Gambar 4.18 Citra labirin bentuk ke-5.

Dengan menghitung secara manual pencarian *node* pada setiap *node* yang ditemukan ditandai dengan lingkaran berwarna hijau yang seluruhnya berjumlah 45 yang dapat ditunjukkan seperti pada gambar 4.19.



Gambar 4.19 Citra labirin bentuk ke-5 yang sudah ditandai secara manual.

Dengan menggunakan aplikasi pada PC pencarian *node* pada citra labirin bentuk ke-5, *node* yang ditemukan ditandai dengan persegi berwarna hijau yang seluruhnya berjumlah 73 yang dapat ditunjukkan seperti pada gambar 4.20.



Gambar 4.20 *Node* yang dapat dideteksi pada citra labirin bentuk ke-5 menggunakan aplikasi pada PC.

Dari gambar 4.19 dan 4.20, pada labirin bentuk ke-5 dapat diketahui jumlah *node* yang didapat melalui perhitungan manual sesuai dengan jumlah *node* yang terdeteksi oleh aplikasi.

4.3 Pengujian Paket Data Rute Terpendek

4.3.1 Tujuan

Tujuan pengujian paket data untuk mengetahui kesesuaian nilai-nilai arah rute terpendek menggunakan algoritma Dijkstra hasil pengolahan menggunakan aplikasi pada PC dengan hasil pengolahan pada *Micromouse Robot*.

4.3.2 Alat yang Digunakan

1. Seperangkat PC yang sudah terdapat aplikasi pencari rute terpendek.
2. *Webcam*.

3. Lintasan labirin dengan 3 bentuk berbeda dan masing-masing bentuk dengan 3 rute berbeda.
4. *Micromouse Robot*.
5. Kabel penghubung komunikasi UART antara PC dengan *Micromouse Robot*.
6. Catu daya 4,8 V.

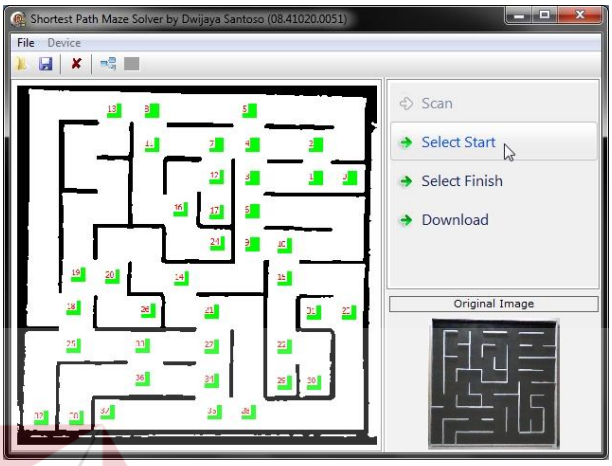
4.3.3 Prosedur Pengujian

1. Download program lengkap ke mikrokontroler ATMega644P dan ATMega328P pada *Micromouse Robot*.
2. Tulis baris program berikut ke *event* "OnRXChar" pada aplikasi pencari rute terpendek (PC).

```
procedure TFormMain.ComPortRxChar(Sender: TObject; Count: Integer);
var
  dt, i : byte;
begin
  comport.Read(dt, 1);
  if (dt = $FF) and (f_uart = false)then
  begin
    f_uart := true;
    exit;
  end;
  if f_receive=1 then
  begin
    Memo1.Lines.Add(inttostr(dt));
    i:=0;
    while integer(i)<integer(uart_cnt-1) do
    begin
      comport.Read(dt, 1);
      Memo1.Lines.Add(inttostr(dt));
      inc(i);
    end;
  end
  else if f_receive=0 then
  begin
    uart_cnt:=dt;
    inc(f_receive);
  end;
end;
```

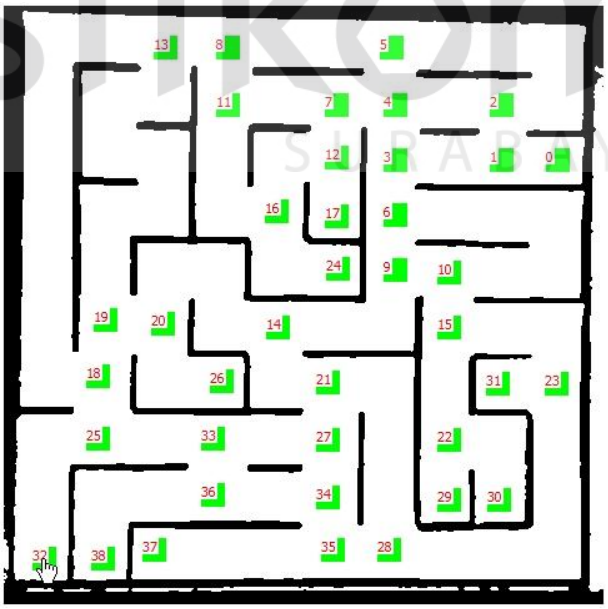
3. Hubungkan *Micromouse Robot* dengan PC menggunakan kabel penghubung.

4. Lakukan pencarian *node* dengan menggunakan aplikasi pencari rute terpendek pada PC.
5. Tentukan titik *start* diawali dengan menekan tombol *start* yang dapat ditunjukkan seperti pada gambar 4.21.



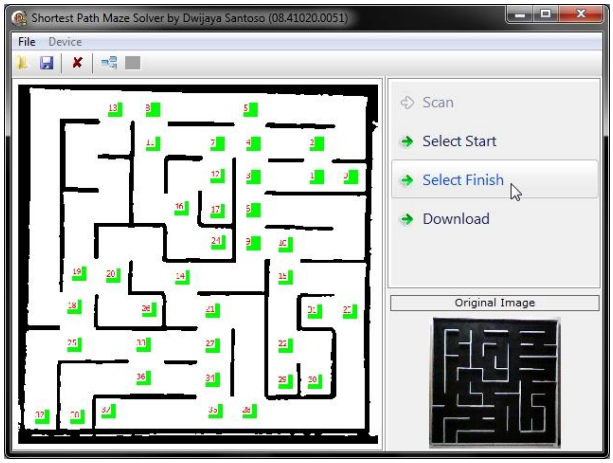
Gambar 4.21 Tombol Start.

Kemudian dilanjutkan dengan memilih salah satu *node* sebagai titik *start* yang dapat ditunjukkan seperti pada gambar 4.22.



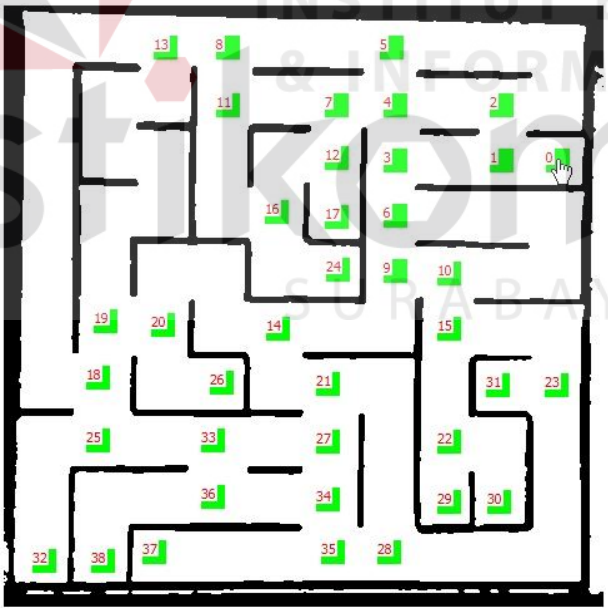
Gambar 4.22 Pemilihan *node* sebagai titik *start*.

6. Tetukan titik *finish* diawali dengan menekan tombol finish yang dapat ditunjukkan seperti pada gambar 4.23.



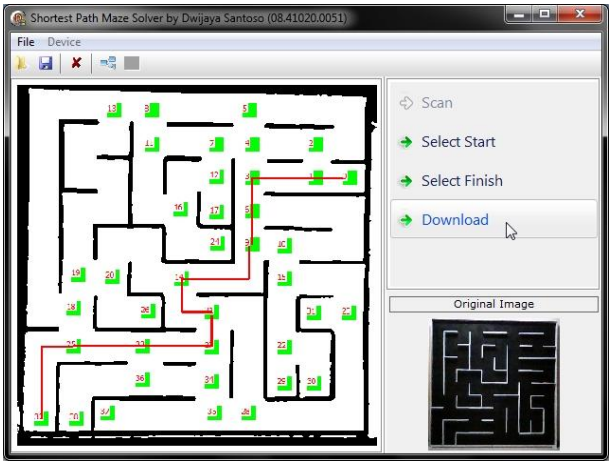
Gambar 4.23 Tombol *finish*.

Kemudian dilanjutkan dengan memilih salah satu *node* sebagai titik *finish* yang dapat ditunjukkan seperti pada gambar 4.24.



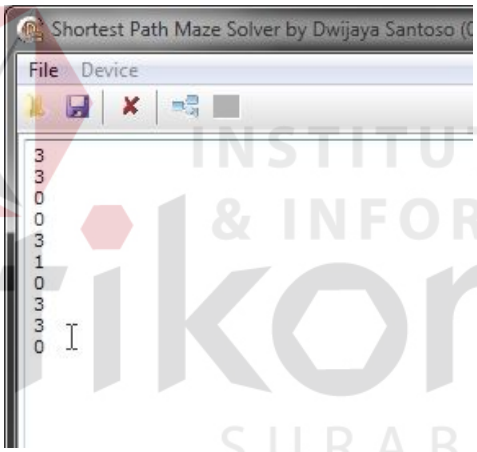
Gambar 4.24 Pemilihan *node* sebagai titik *finish*.

7. Tekan tombol *download* untuk memulai mengirim paket data ke *Micromouse Robot* yang dapat ditunjukkan seperti pada gambar 4.25.



Gambar 4.25 Tombol *download*.

8. Catat nilai-nilai arah rute terpendek yang dikirim oleh *Micromouse Robot* ke PC yang ditampilkan yang dapat ditunjukkan seperti pada gambar 4.26.



Gambar 4.26 Nilai-nilai arah rute terpendek.

9. Bandingkan dengan rute yang dibentuk oleh aplikasi pencari rute terpendek dengan nilai rute yang dikirim oleh *Micromouse Robot*. Nilai 0 untuk utara, 1 untuk barat, 2 untuk selatan, dan 3 untuk timur, sesuai dengan arah mata angin pada peta secara umum.

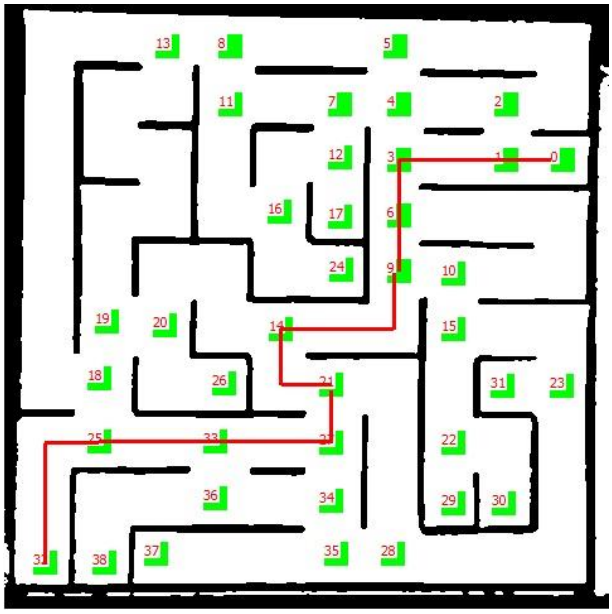
4.3.4 Hasil Pengujian

Hasil pengujian ini adalah rute terpendek menggunakan algoritma Dijkstra hasil pengolahan aplikasi pada PC sesuai dengan hasil pengolahan pada *Micromouse Robot*. *Nodes* yang terdeksi dicari rute terpendeknya menggunakan algoritma Dijkstra dari titik *start* menuju ke *finish* dan rute terpendek yang sudah didapatkan akan ditampilkan oleh aplikasi pada PC. Setelah tombol *download* ditekan, aplikasi (PC) mengirim paket data *nodes* yang terdiri dari *vertex* dan *edge* ke *Micromouse Robot*. Paket data yang diterima dicari rute terpendeknya menggunakan algoritma Dijkstra oleh *Micromouse Robot* dan paket data nilai-nilai arah rute terpendek hasil pengolahan akan dikirim kembali ke aplikasi (PC) untuk ditampilkan.

1. Labirin Bentuk ke-1

A. Rute ke-1

Rute ke-1 pada labirin bentuk ke-1 titik *start* terletak pada *node* ke-32 dan titik *finish* terletak pada *node* ke-0 yang dapat ditunjukkan seperti pada gambar 4.27.



Gambar 4.27 Rute ke-1 pada labirin bentuk ke-1.

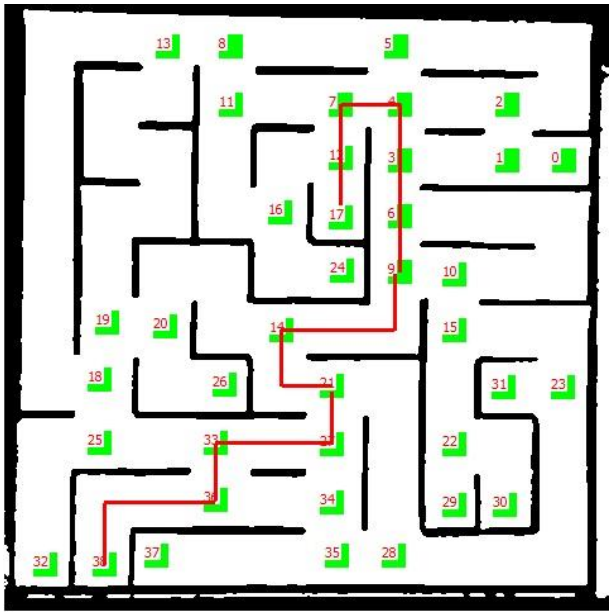
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.3.

Tabel 4.3 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

B. Rute ke-2

Rute ke-2 pada labirin bentuk ke-1 titik *start* terletak pada *node* ke-38 dan titik *finish* terletak pada *node* ke-17 yang dapat ditunjukkan seperti pada gambar 4.28.



Gambar 4.28 Rute ke-2 pada labirin bentuk ke-1.

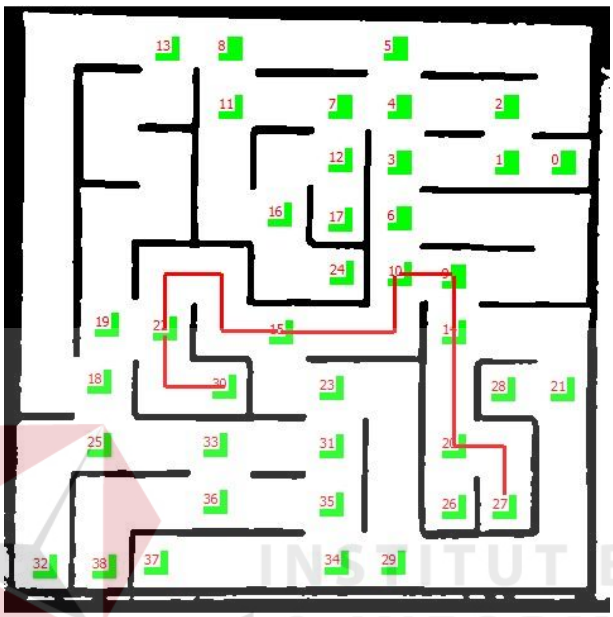
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.4.

Tabel 4.4 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
38	36	0	0	Sesuai
36	33	0	0	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	4	0	0	Sesuai
4	7	1	1	Sesuai
7	12	2	2	Sesuai
12	17	2	2	Sesuai

C. Rute ke-3

Rute ke-3 pada labirin bentuk ke-1 titik *start* terletak pada *node* ke-30 dan titik *finish* terletak pada *node* ke-27 yang dapat ditunjukkan seperti pada gambar 4.29.



Gambar 4.29 Rute ke-3 pada labirin bentuk ke-1.

Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.5.

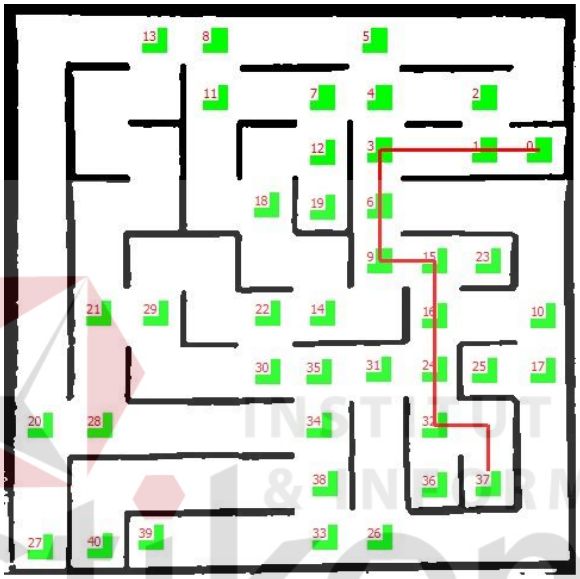
Tabel 4.5 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
30	21	1	1	Sesuai
21	15	0	0	Sesuai
15	10	3	3	Sesuai
10	9	3	3	Sesuai
9	14	2	2	Sesuai
14	20	2	2	Sesuai
20	27	3	3	Sesuai

2. Labirin Bentuk ke-2

A. Rute ke-1

Rute ke-1 pada labirin bentuk ke-2 titik *start* terletak pada *node* ke-0 dan titik *finish* terletak pada *node* ke-37 yang dapat ditunjukkan seperti pada gambar 4.30.



Gambar 4.30 Rute ke-1 pada labirin bentuk ke-2.

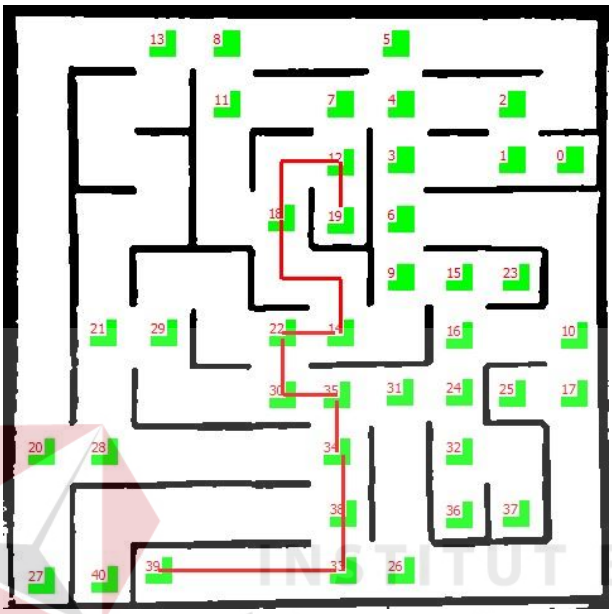
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.6.

Tabel 4.6 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
0	1	1	1	Sesuai
1	3	1	1	Sesuai
3	6	2	2	Sesuai
6	9	2	2	Sesuai
9	15	3	3	Sesuai
15	16	2	2	Sesuai
16	24	2	2	Sesuai
24	32	2	2	Sesuai
32	37	3	3	Sesuai

B. Rute ke-2

Rute ke-2 pada labirin bentuk ke-2 titik *start* terletak pada *node* ke-19 dan titik *finish* terletak pada *node* ke-39 yang dapat ditunjukkan seperti pada gambar 4.31.



Gambar 4.31 Rute ke-2 pada labirin bentuk ke-2.

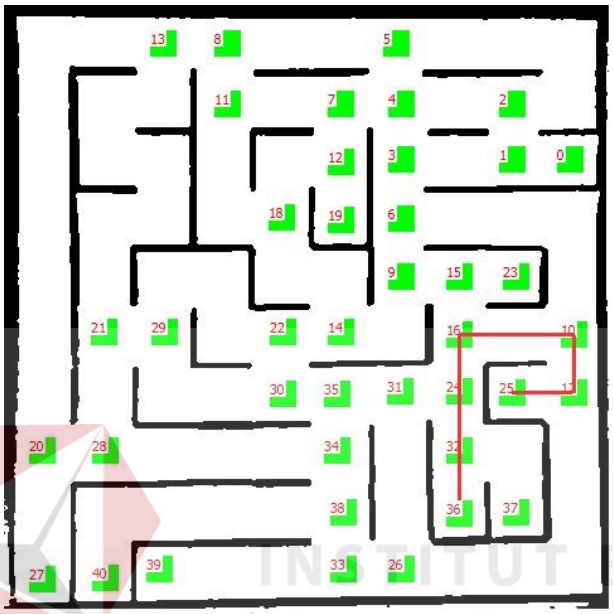
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.7.

Tabel 4.7 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
19	12	0	0	Sesuai
12	18	1	1	Sesuai
18	14	2	2	Sesuai
14	22	1	1	Sesuai
22	30	2	2	Sesuai
30	35	3	3	Sesuai
35	34	2	2	Sesuai
34	38	2	2	Sesuai
38	33	2	2	Sesuai
33	39	1	1	Sesuai

C. Rute ke-3

Rute ke-3 pada labirin bentuk ke-2 titik *start* terletak pada *node* ke-25 dan titik *finish* terletak pada *node* ke-36 yang dapat ditunjukkan seperti pada gambar 4.32.



Gambar 4.32 Rute ke-3 pada labirin bentuk ke-2.

Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.8.

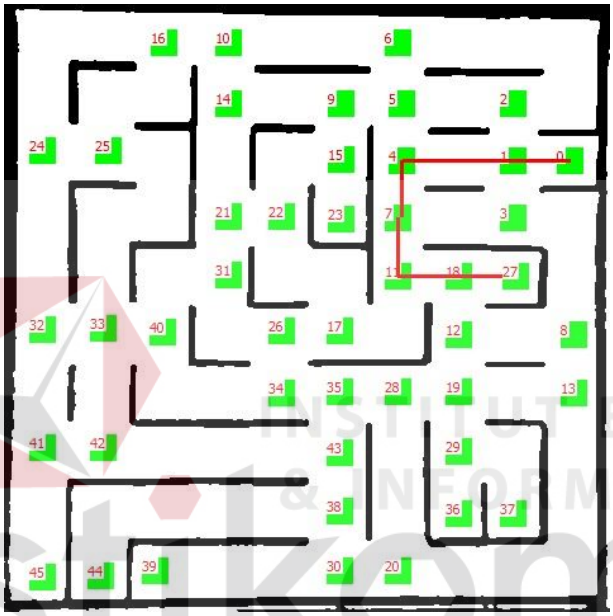
Tabel 4.8 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
25	17	3	3	Sesuai
17	10	0	0	Sesuai
10	16	1	1	Sesuai
16	24	2	2	Sesuai
24	32	2	2	Sesuai
32	36	2	2	Sesuai

3. Labirin Bentuk ke-3

A. Rute ke-1

Rute ke-1 pada labirin bentuk ke-3 titik *start* terletak pada *node* ke-27 dan titik *finish* terletak pada *node* ke-0 yang dapat ditunjukkan seperti pada gambar 4.33.



Gambar 4.33 Rute ke-1 pada labirin bentuk ke-3.

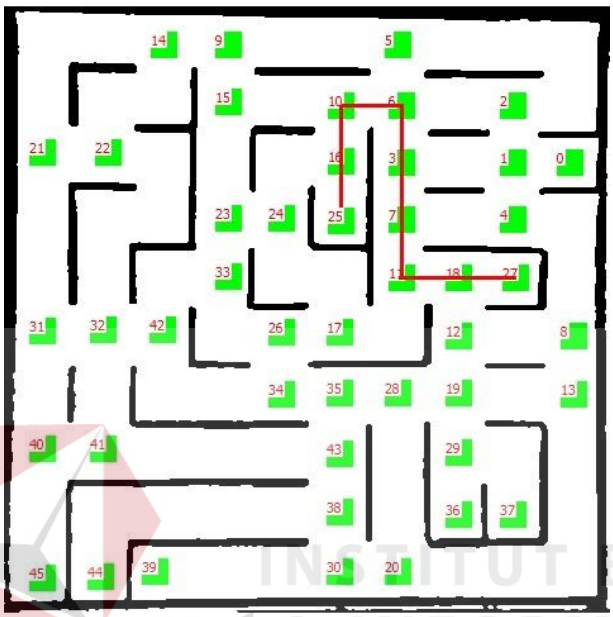
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.9.

Tabel 4.9 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
27	18	1	1	Sesuai
18	11	1	1	Sesuai
11	7	0	0	Sesuai
7	4	0	0	Sesuai
4	1	3	3	Sesuai
1	0	3	3	Sesuai

B. Rute ke-2

Rute ke-2 pada labirin bentuk ke-3 titik *start* terletak pada *node* ke-25 dan titik *finish* terletak pada *node* ke-27 yang dapat ditunjukkan seperti pada gambar 4.34.



Gambar 4.34 Rute ke-2 pada labirin bentuk ke-3.

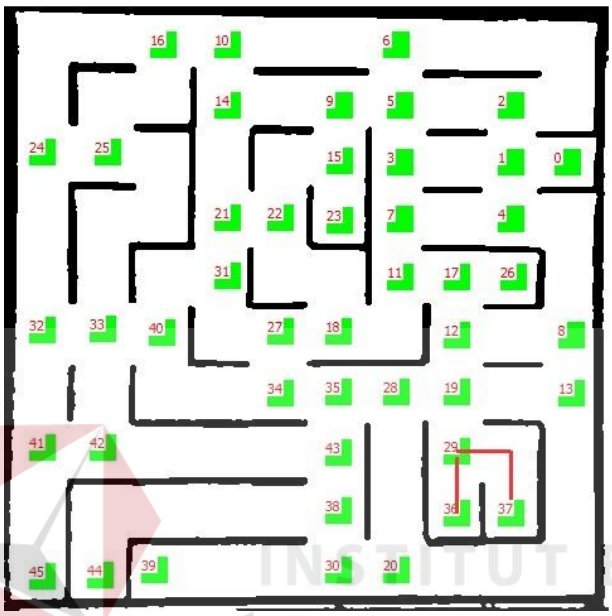
Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.10.

Tabel 4.10 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
25	16	0	0	Sesuai
16	10	0	0	Sesuai
10	6	3	3	Sesuai
6	3	2	2	Sesuai
3	7	2	2	Sesuai
7	11	2	2	Sesuai
11	18	3	3	Sesuai
18	27	3	3	Sesuai

C. Rute ke-3

Rute ke-3 pada labirin bentuk ke-3 titik *start* terletak pada *node* ke-37 dan titik *finish* terletak pada *node* ke-36 yang dapat ditunjukkan seperti pada gambar 4.35.



Gambar 4.35 Rute ke-3 pada labirin bentuk ke-3.

Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada mikrokontroler dapat ditunjukkan seperti pada tabel 4.11.

Tabel 4.11 Kesesuaian rute hasil pengolahan aplikasi pada PC dengan pengolahan pada *Micromouse Robot*.

Previous node	*Next node*	Arah (PC)	Arah (Mikrokontroler)	Keterangan
37	29	0	0	Sesuai
29	36	2	2	Sesuai

4.4 Pengujian Pergerakan *Micromouse Robot*

4.4.1 Tujuan

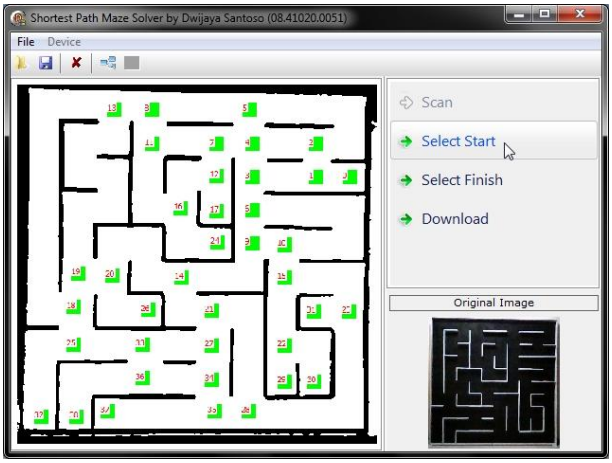
Tujuan pengujian pergerakan *Micromouse Robot* untuk mengetahui kesesuaian rute pergerakan robot dengan rute terpendek hasil pengolahan menggunakan algoritma Dijkstra.

4.4.2 Alat yang Digunakan

1. Seperangkat PC yang sudah terdapat aplikasi pencari rute terpendek.
2. *Webcam*
3. Lintasan labirin.
4. *Micromouse Robot*.
5. Kabel penghubung komunikasi UART antara PC dengan *Micromouse Robot*.
6. Catu daya 4,8 V.

4.4.3 Prosedur Pengujian

1. Download program lengkap ke mikrokontroler ATMega644P dan ATMega328P pada *Micromouse Robot*.
2. Hubungkan *Micromouse Robot* dengan PC menggunakan kabel penghubung.
3. Lakukan pencarian *node* dengan menggunakan aplikasi pencari rute terpendek pada PC.
4. Tentukan titik *start* diawali dengan menekan tombol *start* yang dapat ditunjukkan seperti pada gambar 4.36.

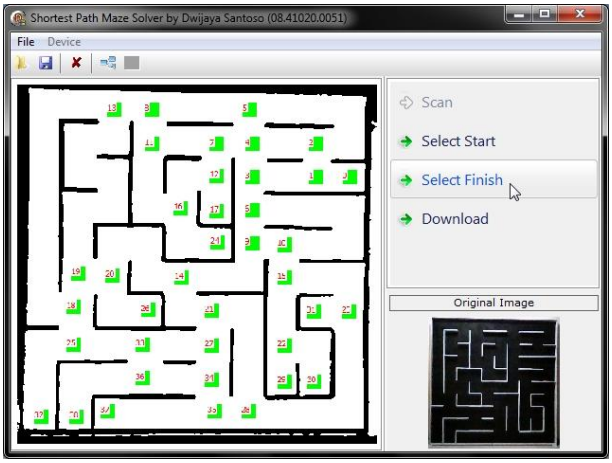


Gambar 4.36 Tombol *Start*.

5. Kemudian dilanjutkan dengan memilih salah satu *node* sebagai titik *start* yang dapat ditunjukkan seperti pada gambar 4.37.

Gambar 4.37 Pemilihan *node* sebagai titik *start*.

6. Tetukan titik *finish* diawali dengan menekan tombol *finish* yang dapat ditunjukkan seperti pada gambar 4.38.

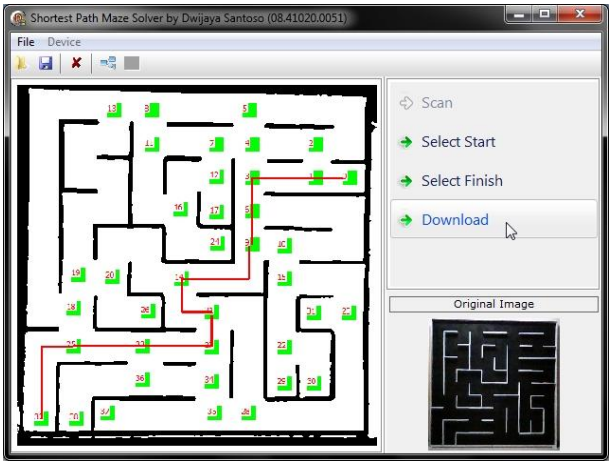


Gambar 4.38 Tombol *finish*.

7. Kemudian dilanjutkan dengan memilih salah satu *node* sebagai titik *finish* yang dapat ditunjukkan seperti pada gambar 4.39.

Gambar 4.39 Pemilihan *node* sebagai titik *finish*.

8. Tekan tombol *download* untuk memulai mengirim paket data ke *Micromouse Robot* yang dapat ditunjukkan seperti pada gambar 4.40.



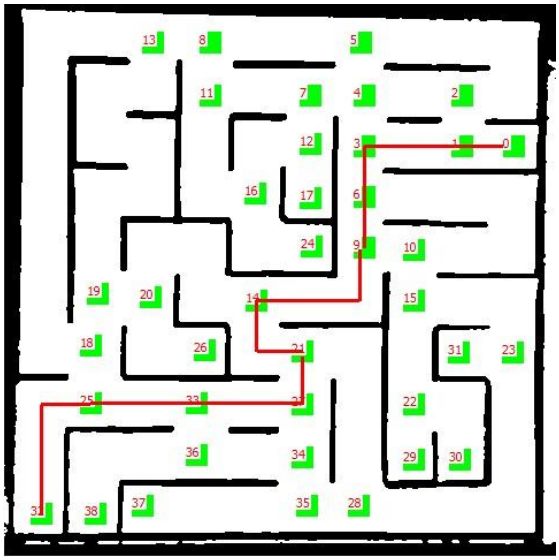
Gambar 4.40 Tombol *download*.

9. Letakkan *Micromouse Robot* pada titik *start* labirin, hadapkan pada arah awal.
10. Tekan tombol *start* untuk memulai pergerakan *Micromouse Robot*.

4.4.4 Hasil Pengujian

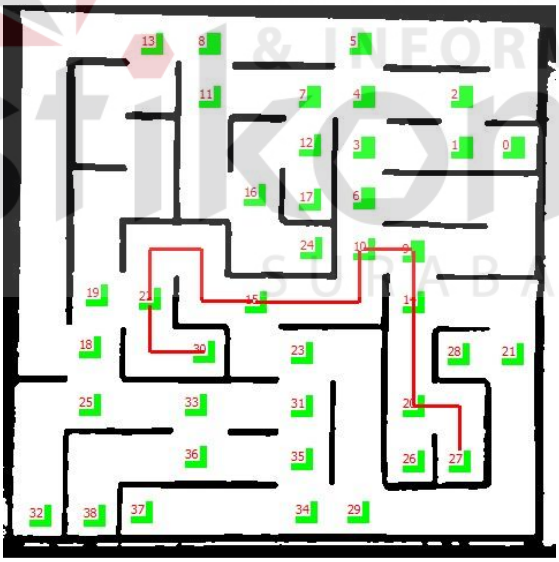
Hasil pengujian ini adalah sebagian besar pergerakan *Micromouse Robot* tidak dapat sampai ke titik *finish*, hanya beberapa *node* yang mampu dilalui sesuai dengan rute terpendek hasil pengolahan menggunakan algoritma Dijkstra.

Rute ke-1 untuk pengujian pergerakan *Micromouse Robot* titik *start* berada pada *node* ke-32 dan titik *finish* berada pada *node* ke-0 seperti yang ditunjukkan pada gambar 4.41.



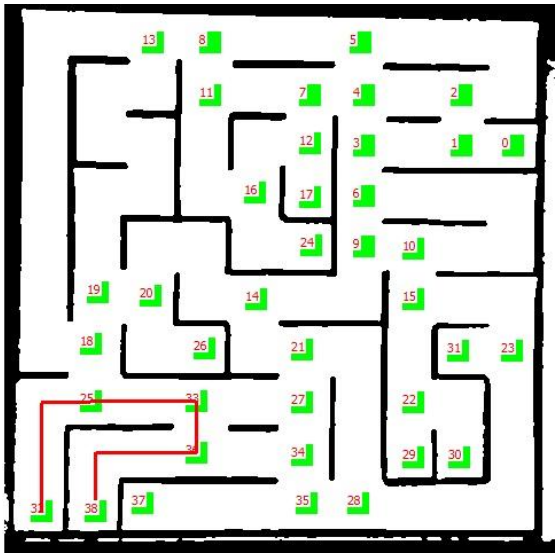
Gambar 4.41 Rute ke-1.

Rute ke-2 untuk pengujian pergerakan *Micromouse Robot* titik *start* berada pada *node* ke-30 dan titik *finish* berada pada *node* ke-27 seperti yang ditunjukkan pada gambar 4.42.



Gambar 4.42 Rute ke-2.

Rute ke-3 untuk pengujian pergerakan *Micromouse Robot* titik *start* berada pada *node* ke-32 dan titik *finish* berada pada *node* ke-38 seperti yang ditunjukkan pada gambar 4.43.

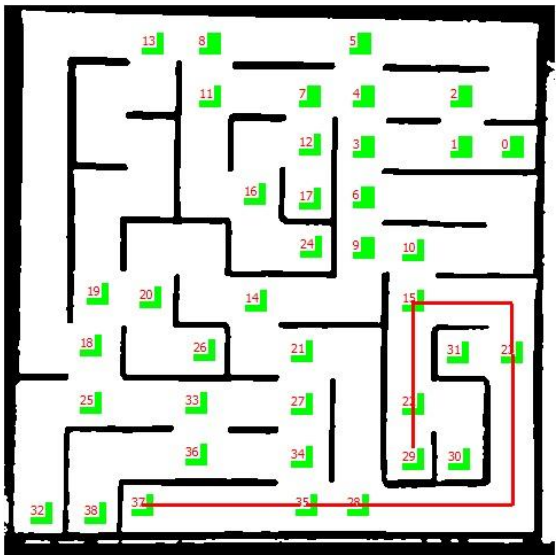


Gambar 4.43 Rute ke-3.

Rute ke-4 untuk pengujian pergerakan *Micromouse Robot* titik *start* berada pada *node* ke-38 dan titik *finish* berada pada *node* ke-17 seperti yang ditunjukkan pada gambar 4.44.

Gambar 4.44 Rute ke-4.

Rute ke-5 untuk pengujian pergerakan *Micromouse Robot* titik *start* berada pada *node* ke-37 dan titik *finish* berada pada *node* ke-29 seperti yang ditunjukkan pada gambar 4.45.



Gambar 4.45 Rute ke-5.

Hasil dari pengujian pergerakan *Micromouse Robot* terhadap 5 rute yang telah ditentukan dapat ditunjukkan seperti pada tabel 4.12.

Tabel 4.12 Pengujian pergerakan *Micromouse Robot*.

Rute	*Source*	*Destination*	PC	Robot	Tingkat keberhasilan
1	32	0	32-25-33-21-14-9-6-3-1-0	32-25	20%
	32	0	32-25-33-21-14-9-6-3-1-0	32-25-33-21-14	50%
	32	0	32-25-33-21-14-9-6-3-1-0	32-25-33	30%
	32	0	32-25-33-21-14-9-6-3-1-0	32-25	20%
	32	0	32-25-33-21-14-9-6-3-1-0	32-25	20%
	32	0	32-25-33-21-14-9-6-3-1-0	32-25	20%
2	30	27	30-22-15-10-9-14-20-27	30-22	25%
	30	27	30-22-15-10-9-14-20-27	30-22	25%
	30	27	30-22-15-10-9-14-20-27	30-22	25%
	30	27	30-22-15-10-9-14-20-27	30	12.5%
	30	27	30-22-15-10-9-14-20-27	30-22	25%

	30	27	30-22-15-10-9-14-20-27	30-22	25%
3	31	38	31-25-33-36-38	31-25	40%
	31	38	31-25-33-36-38	31	20%
	31	38	31-25-33-36-38	31-25	40%
	31	38	31-25-33-36-38	31-25-33	60%
	31	38	31-25-33-36-38	31-25	40%
	31	38	31-25-33-36-38	31	20%
4	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36	15.3%
	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36	15.3%
	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36	15.3%
	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36-33	23%
	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36	15.3%
	38	17	38-36-33-27-21-14-9-6-3-4-7-12-17	38-36-33	23%
5	37	29	37-35-28-23-15-22-29	37-35	28.5%
	37	29	37-35-28-23-15-22-29	37-35	28.5%
	37	29	37-35-28-23-15-22-29	37	14.2%
	37	29	37-35-28-23-15-22-29	37-35-28	42.9%
	37	29	37-35-28-23-15-22-29	37-35	28.5%
	37	29	37-35-28-23-15-22-29	37	14.2%

Micromouse Robot pada pergerakannya tidak sepenuhnya mampu melalui *nodes* rute terpendek hasil pengolahan menggunakan algoritma Dijkstra dari titik *start* menuju *finish*. Pada tingkat keberhasilan terendah, *Micromouse Robot* hanya mampu berangkat dari titik *start* kemudian terjadi kegagalan pada pendeteksian tikungan sehingga sering kali menabrak dinding dan berhenti. Tingkat keberhasilan tertinggi dari pergerakan *Micromouse Robot* yaitu mampu melewati 5 *node* dari 10 *node* yang telah ditentukan. Kegagalan pendeteksian pada tikungan

dan simpangan lebih disebabkan ketidakstabilan sensor ketika robot melakukan gerakan sehingga data yang didapat dari sensor tidak sesuai dengan yang diharapkan.