Pencarian Rute Line Follower Mobile Robot Pada Maze Dengan Metode Q Learning

[1]Samsul Arifin, [2]Arya Tandy Hermawan & [2]Yosi Kristian

[1]Sekolah Tinggi Manajemen Informatika dan Komputer Asia Malang

[2]Sekolah Tinggi Teknik Surabaya

sams.soul@ymail.com, arya@stts.edu, yosi@stts.edu

Abstrak

Dalam penelitian ini robot line follower akan digunakan sebagai agen. Robot agen bertugas mencari rute dalam suatu environment berupa maze, tanpa ada bimbingan langsung dari manusia. Robot diberikan algoritma Q Learning yang merupakan salah satu metode dalam domain Reinforcement Learning. Tujuan dari penelitian ini adalah robot harus bisa menemukan rute dari initial state menuju goal state. Algoritma Q Learning berperan untuk menyimpan state-state jalur maze yang telah dilalui dalam matrik Q. Setiap kali robot mencapai salah satu dari state pertigaan, belok kiri, belok kanan atau lurus maka robot akan memilih aksi yang mungkin. Hal ini akan dilakukan berulang sampai nilai pasangan state aksi pada matriks Q mencapai nilai yang optimal.
Dari hasil uji coba didapatkan data sebanyak 34 state yang telah dipelajari oleh robot. Proses training telah mampu meningkatkan pengetahuan robot sehingga bisa menemukan rute dari inisial state menuju ke goal state. Dari 30 kali percobaan tingkat keberhasilan robot untuk menemukan state adalah 23 kali. Ini berarti bahwa tingkat error antara 24%.

Kata Kunci: Robot Line Follower, Reinforcement Learning, Maze, Q Learning

1 Pendahuluan

Robot menjadi salah satu bagian dari teknologi yang mempermudah kegiatan manusia sehari-hari. Sebagai contoh alat untuk membersihkan lantai rumah, kalau dulu menggunakan sapu kemudian berkembang menjadi vacum clearner dan sampai saat ini telah diciptakan robot yang dapat berjalan secara otomatis untuk membersihkan lantai rumah. Masing-masing robot mempunyai karakteristik yang berbeda-beda dilihat bagaimana cara kerjanya dan dilingkungan apa digunakannnya. Line follower Mobile Robot (LFMR) merupakan salah satu jenis robot yang cukup banyak menjadi obyek penelitian. Dalam penelitian ini robot akan digunakan untuk mencari solusi suatu maze. Maze merupakan representasi dari sebuah lingkungan (environment) untuk robot. Maze terdiri dari susunan garis-garis yang membentuk sebuah pola, dimana ada percabangan, belokan, titik start dan finish. Robot tidak hanya akan mengikuti garis saja, tetapi harus mampu memilih jalan keluar menuju finish (goal). Dalam beberapa penelitian yang sebelumnya belum ada penerapan langsung pada Robot. Penelitian sebelumnya hanya membahas tentang simulasi saja, hal ini belum bisa membuktikan secara nyata. Robot harus bisa menemukan solusi dari maze dengan memilih jalur yang terbaik dan error sekecil mungkin. Solusi yang telah didapatkan oleh robot diharapkan bisa mempersingkat waktu perjalanan robot dari start menuju finish. Hal ini dapat dilakukan bila robot mempunyai sebuah kecerdasan. Metode yang akan diterapkan pada penelitian ini, adalah salah satu algoritma dalam Reinforcement Learning (RL) yaitu Q-Learning. Q- Learning menjawab pertanyaan terhadap koordinasi antara behavior. Q-Learning merupakan sebuah simulasi dasar teknik stokastik yang menyediakan cara untuk menghubungkan state, action dan reward berupa nilai Q dalam tabel. Dalam aplikasinya, kontroller sederhana mencari dalam tabel dan memilih decision (keputusan) dengan nilai terbaik, tidak perlu melakukan komputasi online

yang kompleks. Maze yang akan diselesaikan dengan metode Q Learning direpresentasikan menjadi susunan garis-garis. Maze mempunyai informasi state berupa garis lurus, percabangan, initial state dan goal state. Pergerakan robot yang mengakibatkan perubahan informasi dari state satu ke state lainnya akan disimpan nilainya oleh metode Q Learning. Metode Q Learning menyimpan semua informasi setiap state dan action yang dilakukan oleh robot sehingga dapat menemukan rute terbaik dari initial state menuju goal state.

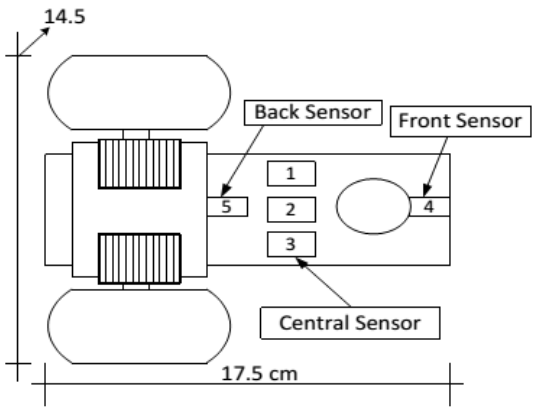
2 Landasan Teori

2.1 Prototipe Mobile Robot

Prototipe robot mobil diimplementasikan dengan sistem transmisi dua roda belakang, ditambah dengan dua motor DC, namun setiap roda berfungsi independen mempunyai sistem gear box dan reduksi. Di depan prototipe ada "crazy wheel" sistem yang memungkinkan untuk mencapai pergerakan bebas hingga 360 derajat. Roda terbuat dari plastik keras dengan plastik polimer kasar yang dibuat untuk menghindari gesekan dengan jalur garis, implementasi ini memungkinkan kinerja membawa beban berat (antara 0,8 dan 1,5 kilogram) dengan konsumsi arus minimal (biasanya 300 milliamps). Gerakan yang diizinkan adalah

1. Arah depan dan belakang
2. Berbelok ke kiri dan ke kanan
3. Berputar 360 derajat

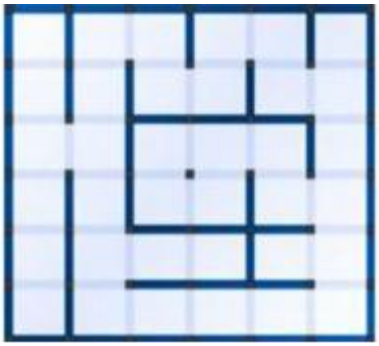
Pembagian sensor dan model penempatan roda depan maupun roda belakang dalam sistem ditunjukkan pada Gambar 1.



Gambar 1 Sistem transmisi letak roda dan sensor

2.2 Desain Maze

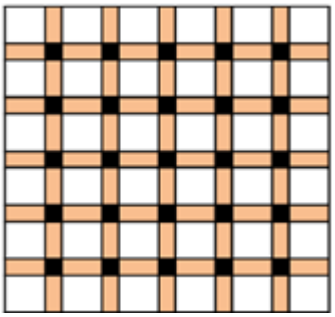
Maze didesain untuk robot agar menemukan solusi dari sel berukuran 6x6 yang ditunjukkan pada Gambar 2. Konstruksi maze yang sebenarnya mempunyai ukuran fisik kira- kira 2,25m^2. Maze dirancang sehingga akan memiliki dua jalur yang harus dipecahkan. Salah satu jalur lebih panjang dari yang lain. Robot harus memutuskan mana salah satu jalur terpendek dan memecahkan maze melalui jalan tersebut.



Gambar 2 Desain maze untuk robot

Agar robot dapat memecahkan maze, robot harus mengetahui bagaimana besar maze, dan membaginya secara virtual menjadi sejumlah sel- sel yang dapat digunakan kemudian menghitung jalur terpendek menuju ke tujuan. Dalam proyek ini, digunakan maze 6x6 sel. Antara dua sel bisa ada dinding. Dengan demikian, dengan deretan 6 sel, ada 5 dinding diantaranya.

Secara total, berturut-turut ada sebelas unit sel atau dinding. Informasi ini disimpan dalam 11x11 array, seperti ditunjukkan pada Gambar 3. Unit putih adalah sel dimana robot dapat ditempatkan didalamnya. Unit jingga adalah lokasi dari dinding. Unit hitam menunjukkan persimpangan dinding yang diabaikan oleh algoritma. Perbatasan eksternal maze juga diabaikan karena mereka adalah batas tetap maze. Keduanya antara sel putih dan dinding jingga diset ke nol sebagai kondisi awal mereka.



Gambar 3 Array mapping maze

Sebelum robot memutuskan kemana akan berpindah, robot harus memeriksa apakah dikelilingi oleh dinding di salah satunya dari tiga arah, kanan, kiri dan depan. Robot membaca jarak beberapa hambatan pada tiap arah dan memeriksa apakah jarak masing-masing lebih dari 20 cm. Salah satu yang tidak lebih dari 20 cm diperbaharui sebagai dinding pada masing- masing sisinya.

2.3 Metode Q Learning

Q-Learning merupakan sebuah teknik kecerdasan buatan yang telah berhasil dimanfaatkan untuk memecahkan MDP kompleks seperti sistem model realistik. Pada paradigma Q-Learning, suatu robot berinteraksi dengan lingkungan dan menjalankan sekumpulan action. Lingkungan kemudian dimodifikasi dan robot mengpersepsikan state baru melalui

sensor. Selanjutnya, pada setiap epoch robot menerima sinyal reward dari lingkungan. Dalam strategi pembelajaran ini, tujuan didefinisikan dan proses pembelajaran terjadi melalui interaksi trial dan error dalam sebuah lingkungan dinamis. Robot diberi reward berdasarkan action yang dilakukannya. Berikutnya s dinyatakan sebagai suatu state dan d adalah decision, tujuan dari strategi pembelajaran ini adalah mengajari robot mengoptimalkan pengontrolan policy, s→d, untuk memaksimalkan jumlah reward yang diterima dalam jangka panjang. Selama proses pembelajaran nilai Q setiap state-decision dipasangkan, Q(s,d) disimpan dan diupdate. Nilai Q merepresentasikan fungsi dari proses eksekusi decision (d) saat lingkungan dalam state (s). Q- Learning secara langsung memilih fungsi optimal nilai decision, tidak bergantung pada policy saat ini yang dijalankan

$$Q(s_t, d_t) = (1-\alpha).Q(s_t, d_t) + \alpha.(r + \gamma.\max_{d_{t+1}} Q(s_{t+1}, d_{t+1})) \quad (1)$$

Dimana r menyatakan reward yang diterima pada epoch t, 0<γ <1 menyatakan faktor diskon dan α menyatakan learning rate. Q*(s,d) menyatakan nilai Q optimal yang diharapkan dari state-decision yang dipasangkan (s,d). Jika setiap decision dieksekusi dalam setiap state yang tidak terbatas, Q(s,d) akan konvergen dengan Q*(s,d).

2.3.1 Algoritma Q-Learning

Dapatkan *state* s_t saat ini
Pilih sebuah *decision* d_t kemudian
eksekusi
Dapatkan *state* s_{t+1} baru dan *immediate reward r.*
Update matrik $Q(s_t, d_t)$ dengan persamaan:

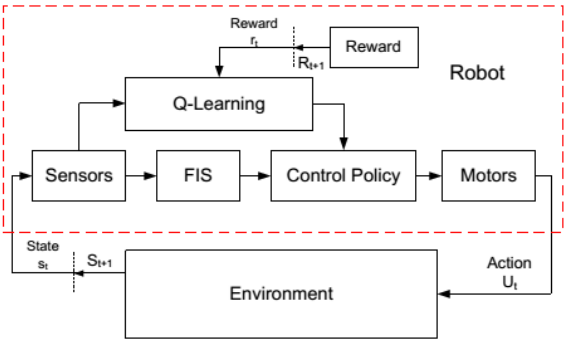
$$Q(s_t, d_t) = (1-\alpha).Q(s_t, d_t) + \alpha.(r + \gamma.\max_{d_{t+1}} Q(s_{t+1}, d_{t+1}))$$

Menentukan $s_t = s_{t+1}$
Bila $s_t \neq s_{optimal}$ kembali ke langkah 2

dimana α adalah training rate dan adalah faktor diskon. Kedua parameter didapatkan dari interval [0,1].

Algoritma diatas merupakan algoritma Q Learning yang sudah dilakukan modifikasi dalam penelitiannya Khriji. Modifikasi dilakukan pada bagian persamaan yang digunakan untuk proses update matriks Q.

3 Desain Sistem



Gambar 4 Arsitektur sistem robot

Sistem yang dimodelkan dalam penelitian ini lebih terarah pada proses pembelajaran dari robot (yaitu software). Arsitektur sistem dari robot secara keseluruhan ditunjukkan pada Gambar 4.

Gambar 4 menjelaskan interaksi robot dengan lingkungan dalam reinforcement learning

Pada tiap kali langkah t, robot mengamati state saat ini melalui sensor dan membangkitkan sebuah aksi dari serangkaian aksi yang mungkin berhubungan dengan state . Satu langkah berikutnya, sebagai konsekuen dari aksi sebelumnya robot menerima reward atau punishment berupa angka ketika itu juga robot mencapai state baru . Pada setiap kali melangkah, robot mengimplementasikan sebuah pemetaan state untuk penyeleksian probabilitas tiap kemungkinan aksi. Pemetaan ini disebut sebagai learner's control policy yang dinyatakan dengan π.

Robot mengeksplorasi environment untuk meningkatkan pengetahuan otaknya dalam mengenali pola dari maze. Tiap-tiap bagian pada gambar blok arsitektur sistem robot memiliki fungsi masing-masing. Environment merupakan inti permasalahan yang harus dipecahkan. Robot harus mampu mendapatkan segala informasi data dalam bentuk apapun yang terdapat dalam environment sehingga robot bisa mengenali lingkungannya dengan baik.

Sensor berperan untuk melihat kondisi environment, kondisi inilah yang dinamakan state. Setelah state diketahui maka sensor akan mengirimkan datanya ke memori Q learning dan Fuzzy Inference System. Q learning berfungsi untuk menyimpan nilai pasangan state dan action. Fuzzy Inference System berfungsi sebagai kontroler untuk navigasi robot agar bisa mengikuti garis dengan baik. Control Policy berfungsi untuk mengevaluasi pasangan state dan action yang memiliki nilai maksimum.

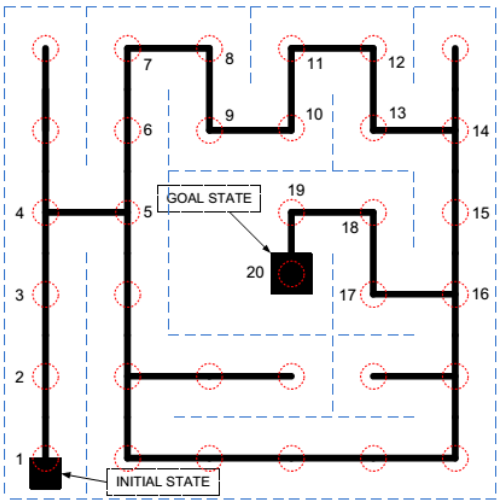
3.1 Penerapan Algoritma Q Learning

Dalam subbab ini akan diuraikan secara detail mengenai tracing dari algoritma Q Learning. Tracing algoritma Q Learning ini bertujuan untuk memudahkan pemahaman dalam menganalisa perhitungan dalam sistem robot. Sehingga dapat diketahui bila terjadi kesalahan dalam implementasi.

Kelebihan Q-Learning adalah sifatnya yang off policy (dapat mengikuti aturan apapun untuk menghasilkan aturan optimal), kemudahan algoritmanya, dan kemampuannya untuk

konvergen pada aturan optimal. Jadi dengan menggunakan Q-Learning akan mampu menyelesaikan permasalahan maze dengan konfigurasi yang bisa berubah-ubah menjadi 3 bentuk yang berbeda (fleksibel).

Environment pada Gambar 5 adalah salah satu bentuk konfigurasi dari maze. Tanda lingkaran merah pada environment merupakan state yang nanti akan disimpan dalam matrik. Nomor-nomor disamping lingkaran adalah contoh penomoran untuk setiap state. Konfigurasi maze ini akan dijadikan untuk perancangan metode Q Learning. Pada kondisi awal, robot diasumsikan belum mengenali environment yang berbentuk maze ini



Gambar 5 Penentuan state state pada maze

Robot akan diletakkan dalam maze kemudian robot akan berjalan menelusuri garis-garis. Kegiatan robot dalam menelusuri ruangan ini yang disebut sebagai proses eksplorasi. Eksplorasi ini kemudian dibagi ke dalam episode- episode. Satu episode menyatakan satu kali robot menyelesaikan eksplorasi dari initial state sampai menemukan goal state. Percobaan ini terus dilakukan sampai beberapa episode sehingga akan membentuk nilai Q yang maksimum. Tanda lingkaran merah yang ada pada maze merupakan posisi dimana robot akan menyimpan state lingkaran tersebut. Selain untuk pembacaan state, pada tanda lingkaran tersebut digunakan untuk pemberian reward atau punishment pada robot.

Cara kerja algoritma Q Learning untuk memecahkan permasalahan maze:

3.1.1 Langkah 1

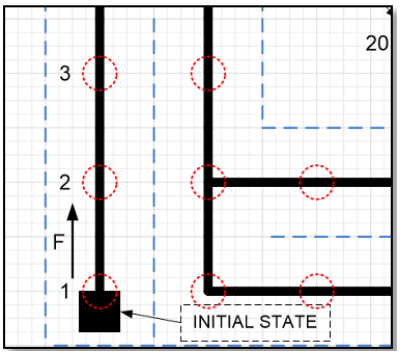
- Inisialisasi Matrik Q dengan 0

		Action			
	State	TL	TR	F	R
Q =	1	0	0	0	0
	2	0	0	0	0
	3	0	0	0	0
	4	0	0	0	0
	5	0	0	0	0
	...	...	...	...	...
	19	0	0	0	0

- Inisialisasi: γ+0.5, r=100 Jika bergerak ke S20, r=-1 jika bergerak ke rute buntu dan r=0 jika bergerak ke lainnya

3.1.2 Langkah 2

- Posisi robot saat ini adalah s1, tersedia aksi F
- Jalankan aksi F



3.1.3 Langkah 3

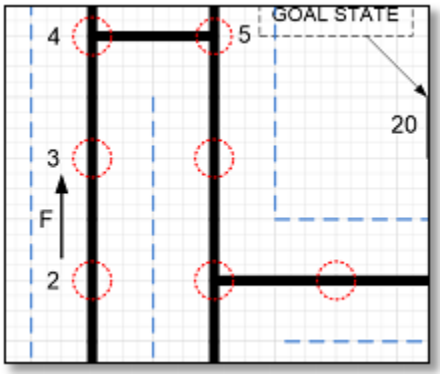
- Update Q(s1,F), posisi saat ini s2
- Update Q(s1,F): Q(s1,F)=r+0.5 max(Q(s2,F))=0

  Sehingga matriks (poongan matriks) menjadi

		Action			
	State	TL	TR	F	R
Q =	1	0	0	0	0
	2	0	0	0	0
	3	0	0	0	0
	4	0	0	0	0
	5	0	0	0	0
	...	...	...	...	...
	19	0	0	0	0

3.1.4 Langkah 4

- Gerakan berikutnya, posisi robot saat ini adalah s2, aksi yang tersedia F
- Jalankan aksi F



3.1.5 Langkah 5

- Update Q(s2,F), posisi saat ini s3
- Update Q(s2,F): Q(s2,F)=r+0.5*max(Q(s3,F))=0

  Sehingga matriks(potongan matriks) menjadi:

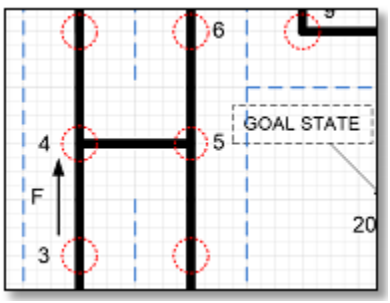
Action

State	TL	TR	F	R
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0
5	0	0	0	0
...	..	..	..	..

Q=

3.1.6 Langkah 6

- Gerakan berikutnya, posisi robot saat ini adalah s3, aksi yang tersedia F
- Jalankan aksi F



3.1.7 Langkah 7

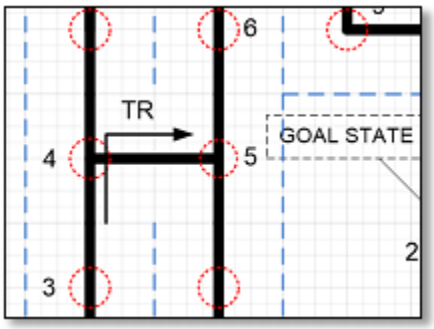
- Update Q(s3,F), posisi saat ini s4
- Update Q(s3,F): Q(s3,F)=e+0.5*max(Q(s4,F)=0

  Sehingga matriks (potongan matriks) menjadi:

	State	TL	TR	F	R
	1	0	0	0	0
	2	0	0	0	0
Q=	3	0	0	0	0
	4	0	0	0	0
	5	0	0	0	0
	...	..	..	..	..

3.1.8 Langkah 8

- Gerakan berikutnya, posisi robot saat ini adalah s4, aksi yang tersedia F dan TR
- Pilih dan jalankan aksi TR



3.1.9 Langkah 9

- Update Q(s4,F), posisi saat ini s5
- Update Q(s4,F): Q(s4,F)=r+0.5*max(Q(s5,F))=0

  Sehingga matriks (potongan matriks) menjadi:

	State	TL	TR	F	R
	1	0	0	0	0
	2	0	0	0	0
Q=	3	0	0	0	0
	4	0	0	0	0
	5	0	0	0	0
	...	..	..	..	..

- Langkah berikutnya sesuai dengan urutan langkah diatas sampai menemukan goal state atau s20.

3.1.10 Langkah terakhir pasa episode ini

Langkah terakhir dari episode ini adalah saat robot mencapai goal state

- Update Q(s19,TL), posisi saat ini s20, Goal State
- Update Q(s19,TL): Q(s19,TL)=r+0.5*max(Q(s20,TL))=100

  Sehingga matriks (potongan matriks) menjadi:

		Action			
	State	TL	TR	F	R
	...	...	...	...	...
	15	0	0	0	0
Q =	16	0	0	0	0
	17	0	0	0	0
	18	0	0	0	0
	19	100	0	0	0

3.1.11 Langkah berikutnya

Mengulangi langkah diatas dari initial state sampai tercapai goal state sampai beberapa episode sehingga matrik Q terisi dengan nilai yang maksimum

4 Hasil dan Kesimpulan

Algoritma pada control policy menuntun robot agar sedapat mungkin tidak melewati rute-rute yang dianggap tidak memiliki nilai yang optimal. Hal ini dapat bermanfaat untuk menyingkat waktu training.

4.1 Proses training

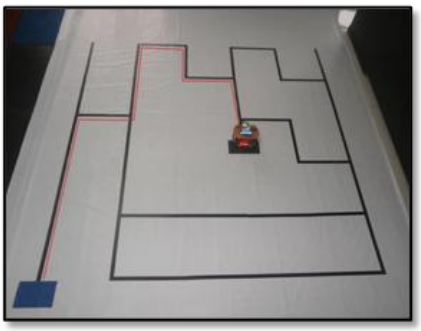
State	Action			
	TL	TR	F	R
1	0	0	216	0
2	0	0	188	0
3	0	0	144	0
4	1	90	-1	0
5	0	0	-2	0
6	0	0	0	-2
7	0	75	0	0
8	0	0	55	0
9	46	37	0	0
10	0	0	28	0
11	15	0	11	0
12	11	0	0	0
13	0	0	11	0
14	0	0	15	0
15	214	0	0	0
16	0	121	0	0
17	0	44	0	0
18	54	486	0	0
19	1039	0	0	0
20	0	0	0	1039
21	0	0	17	0
22	0	0	12	0
23	0	0	13	0
24	0	0	23	0
25	0	221	0	0
26	471	0	0	0
27	30	0	0	0
28	0	28	0	0
29	0	0	0	-1
30	-1	38	1	0
31	0	0	59	0
32	126	93	0	0
33	43	0	33	0
34	17	0	0	0

Gambar 6 Hasil training

Gambar 6 menunjukkan perolehan nilai hasil training pada episode terakhir. Pada state 4 bila robot dari state 3 terlihat nilai TR lebih besar dari pada nilai F maka robot akan lebih memilih TR sebagai rute yang akan dilalui. Pada state 9 nilai TL lebih besar dari pada TR ini berarti robot akan memilih TL. Pada state 18 nilai TR lebih besar dari pada nilai TL berarti robot akan memilih TR. Pada state 30 nilai TR lebih besar dari pada nilai TL sehingga robot akan memilih TR. Pada state 11 nilai TL lebih besar dari pada F sehingga robot akan memilih TL. Pada state 32 dan 33 ada dua kemungkinan nilai yang berbeda saat menuju rute yang sama, pemilihan nilai berdasarkan dari arah datangnya robot. Pada state 32 bila robot datang dari state 33 maka robot akan memilih jalur TL sedangkan bila datang dari state 31 robot akan memilih jalur TR. Kondisi ini juga terjadi pada state 33 dimana bila robot datang dari state 24 akan dipilih jalur TL sedangkan bila robot dari arah state 34 robot akan memilih jalur F. Pasangan nilai state dan aksi yang telah terbentuk akan membantu robot dalam behavior.

4.2 Proses testing

Pertama kali robot dijalankan, dari initial state robot berjalan lurus sampai mencapai state 4, kemudian dari state 4 robot berbelok kekanan sampai pada state 9. Dari state 9 robot berbelok kekiri menuju ke state 8. Setelah sampai pada state 8 robot bergerak lurus menuju state 7, kemudian belok ke kanan sampai pada state 16. Dari state 16 robot belok kekanan lagi menuju state 15, kemudian belok kiri menuju state 18. Setelah sampai pada state 18 robot memilih jalur kanan sampai menuju ke state 19. Dari state 19 robot lurus dan mencapai state 20 yaitu goal state. Pergerakan robot dari initial state menuju goal state di tunjukkan pada Gambar 7.



Gambar 7 Rute 1 yang dilewati

4.3 Kesimpulan

Dalam metode Q Learning ada algoritma untuk menentukan aksi yaitu control policy. Control policy dalam penelitian sudah berfungsi dengan baik, hal ini terbukti dengan didapatkan tabel Q yang berisi nilai yang sudah optimal. Keberhasilan proses pembelajaran robot untuk menemukan rute juga sangat dipengaruhi oleh pemberian nilai reward pada tiap state. Berdasarkan hasil uji coba didapatkan nilai reward yang paling baik adalah 5 untuk semua state sedangkan -10 untuk state jalur buntu.

5 Daftar Pustaka

[1] I. Elshamarka, A. B. S. Saman, "Design and Implementation of a Robot For Maze-Solving using Flood-Fill Algoritm" International Journal of Computer Applications (0975 – 8887) Volume 56– No.5, October 2012.

[2] Román Osorio C., José A. Romero, Mario Peña C., Ismael López-Juárez, "Inteligent Line Follower Mini-Robot System", International Journal of Computers, Communications & Control, Vol. I (2006), No. 2, pp. 73-83.

[3] Lazhar Khriji, Farid Touati, Kamel Benhmed and Amur Al-Yahmedi, "Mobile Navigation Based on Q-Learning Technique" International Journal of Advanced Robotic Systems, Vol. 8, No. 1 (2011) ISSN 1729- 8806, pp 45-51.

[4] Russell Stuart dan Norvig Peter, "Artificial Intelligence A Modern Approach", Prentice Hall, 2003.

[5] Sutton Richard S. dan Barto Andrew G, "Reinforcement Learning: An Introduction", The MIT Press, 2005.