ROBOT OBSTACLE AVOIDANCE DENGAN ALGORITMA Q-LEARNING

Indra Agustian[1], Alex Surapati[2], Aji Arya Dewangga[3], Ruvita Faurina[4]

[1, 2, 3] Teknik Elektro, Fakultas Teknik, Universitas Bengkulu
[4]Teknik Informatika, Fakultas Teknik, Universitas Bengkulu

[1]indraagustian@unib.ac.id, [2]alexsurapati@unib.ac.id, [3]aaji231@gmail.com,[4]ruvita.faurian@unib.ac.id

Abstrak

Perancangan prototipe robot *obsctale avoidance* tipe beroda dengan menerapkan algoritma *Q-Learning* telah diimplementasikan pada penelitian ini. Robot dirancang menggunakan mikrokontroler ATMega2560 pada platform Arduino Mega2560 sebagai pusat kontrol. Robot dilengkapi dengan lima sensor ultrasonik HC-SR04 dan lima sensor IR sharp GP2Y0A21YK0F. Posisi rintangan dibagi menjadi zona dan sektor. Zona menunjukkan posisi kanan atau kiri dan sektor adalah posisi sudut. Berdasarkan kombinasi nilai zona dan sektor, state terdiri atas 144 kondisi, sedangkan action dibagi menjadi lurus, kanan dan kiri. Dari 300 kali percobaan, nilai optimal *learning rate* konvergen di angka 0,5 dan *discount factor* di angka 0,9 setelah mencapai 250 percobaan. Robot mampu beradaptasi dengan cepat pada rintangan statis, dan lebih lama pada rintangan dinamis. Robot akan terus memperbarui nilai *reward* untuk beradaptasi pada setiap eksplorasi baru.

Kata Kunci: reinforcement learning, q-learning, robot beroda, obsctale avoidance, navigasi robot.

Abstract

A prototype of an obstacle avoidance mobile robot was developed by applying the Q-learning algorithm is presented. The robot is consisted of an ATMega2560 microcontroller on the Arduino Mega2560 platform as the main control hardware. The robot is equipped with five HC-SR04 ultrasonic sensors and five GP2Y0A21YK0F sharp IR sensors. The obstacle positions are divided into general zones and sectors. The zone indicates the position right or left and the sector is angle position. Based on the combination of zone and sector values, the state had 144 conditions, while the action was divided into straight, right, and left. From 300 experiments, the optimal value of learning rate converges at 0.5, and the discount factor at 0.9 after reaching 250 trials. The robot could adapt quickly to static obstacles, and longer to dynamic obstacles. The robot then will continue updating the reward value to adapt each new exploration.

***Key Words*: reinforcement learning, q-learning, wheeled robots, observational avoidance, robot navigation.**

1. Pendahuluan

Robot telah menjadi kebutuhan yang sangat penting karena dapat mengemban tugas dan fungsi yang sangat fleksibel dalam membantu pekerjaan manusia [1]. Pembuatan robot dengan keistimewaan dan keahlian khusus sangat dibutuhan dalam dunia industri modern, sehingga sebuah alat dengan kemampuan tinggi yang dapat membantu pekerjaan manusia menjadi sesuatu yang penting[2].

Di dalam teknologi robot, tergabung beberapa tema penelitian yang juga berkembang seperti teknologi sensor, teknologi motor, teknologi suplai daya, teknologi telekomunikasi, teknologi pengendalian dan teknologi kecerdasan buatan.Perkembangan masing-masing teknologi tersebut saling menyempurnakan untuk mendukung kemajuan teknologi robot. Banyak cara yang dapat digunakan untuk menambah kecerdasan robot, salah satunya adalah dengan cara menambahkan berbagai sensor pada robot dan sistem kendali pada robot[3].

Sistem kendali pada robot memegang peranan yang sangat penting untuk meningkatkan efektivitas dan efisiensi dalam proses pengoperasian, terlebih lagi pada sistem yang telah ditanamkan kecerdasan buatan [4][5]. Kecerdasan buatan memungkinkan sebuah robot agar terlatih untuk melakukan tindakan yang dapat dilakukan oleh manusia menjadi lebih baik. *Machine learning* [6][7], merupakan salah satu bagian dari kecerdasan

buatan yang memungkinkan sebuah mesin dapat belajar dari pengalaman masa lalu dan bertindak sendiri, tanpa perlu diprogram secara eksplisit setiap kali mesin tersebut menerima keluaran yang berbeda. Saat ini, machine learning sangat dibutuhkan untuk dunia robotika guna meningkatkan efisiensi pada robot.

Pada penelitian ini, dikembangkan suatu prototipe robot beroda yang dapat menghindari halangan (*obstacle avoidance*) dengan menggunakan metode *Reinforcement Learning* (RL) [8]. Pada metode RL, robot dapat berkomunikasi dengan lingkungannya tanpa dibantu oleh tutor atau guru untuk pembelajarannya.

Tujuan RL adalah untuk menemukan kondisi aksi pada suatu step-time yang mengantarkannya untuk mendapatkan reward positif atau negatif. Aksi yang dilakukan robot tidak menganut benar atau tidaknya aksi sesaat yang dilakukan, melainkan total *reward* yang didapat untuk mencapai tujuan. Robot tersebut selanjutnya dapat menemukan kondisi terbaik dengan sendirinya. Pada umumnya, RL dapat diselesaikan dengan teknik pemrograman dinamis dan informasi tentang keadaan lingkungan biasanya dinyatakan dalam bentuk *Markov Decision Process* (MDP). Dari basis MDP ini,, banyak algoritma RL yang telah dikembangkan oleh peneliti-peneliti sebelumnya, di antaranya yang paling populer adalah *Q-learning* [8] [9] [10], Sarsa [8] dan Monte Carlo [11].

Pada penelitian ini, dirancang *Robot Obstacle Avoidance* (RoA) beroda aktual menggunakan algortima *Q-Learning* pada lingkungan statis dan juga dinamis dan ROA ini diberi nama Qlobot. Pembaharuan nilai-nilai pada tabel Q dilakukan menggunakan persamaan Bellman. Qlobot dirancang dengan menggunakan perangkat keras-perangkat keras yang ekonomis dan bisa diperoleh dengan cukup mudah, secara detail disebutkan pada bagian metode penelitian. Eksperimen aktual dilakukan pada arena dengan rintangan statis dan dinamis. Pada penelitian ini juga dilakukan uji *learning rate* dan *discount factor* untuk menemukan nilai yang paling optimal untuk ROA.

2. Penelitian Terkait

2.1 *Q-learning*

Q-learning merupakan salah satu teknik RL yang digunakan dalam *machine learning*. *Q-learning* ditemukan pertama kali dalam disertasi Watkins [9] [10] berjudul "*Learning from Delayed Rewards*" dan Watkins mengusulkan algoritma *Q-learning* sebagai model RL untuk melakukan optimasi kontrol MDP. Model *Q-learning* bertujuan untuk mempelajari aturan (policy), dan memberitahukan agent atau robot untuk beraksi (*action*) berdasarkan kondisi (*state*). tidak memerlukan model lingkungan dan dapat menyelesaikan permasalahan transisi stokastik dan *rewards*, tanpa memerlukan adaptasi.

Pada paradigma *Q-learning*, suatu robot berinteraksi dengan lingkungan dan menjalankan sekumpulan action. Lingkungan kemudian dimodifikasi dan robot mempersepsikan state baru melalui sensor. Selanjutnya, pada setiap jangka waktu tertentu 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.

$$s_0 \overset{a_0}{\rightarrow} \{s_1, r_1\} \overset{a_1}{\rightarrow} \{s_2, r_2\} \overset{a_2}{\rightarrow} \{s_3, r_3\} \cdots \overset{a_n}{\rightarrow} \{s_{(n+1)}, r_{(n+1)}\} \quad (1)$$

dengan,

- s= kondisi
- a= keputusan
- r= *reward*

Ilustrasi pembelajaran metode *Q-learning* dapat direpresentasikan dalam Persamaan (1). Pada saat robot berada pada kondisi s_0 akan melakukan aksi $a_0$sehingga robot akan menerima *reward* r_1 kemudian akan berada pada kondisi s_1. Selanjutnya robot akan melakukan aksi a_1, sehingga robot akan menerima *reward* r_2 kemudian akan berada pada kondisi s_2 dan begitu seterusnya hingga pembelajaran dihentikan. Maka urutan dari pembelajaran tersebut dapat direpresentasikan ke dalam Persamaan (2).

$$s \overset{a}{\rightarrow} \{s', r\} \quad (2)$$

dengan,

- s'=Perpindahan agen dari kondisi sebelumnya

Kunci untuk metode *Q-learning* adalah untuk memperbarui nilai tabel-Q yang ditugaskan untuk robot ketika melakukan keputusan yang ada di lingkungannya. Nilai Q hasil pembelajaran dapat diperbaharui dengan menggunakan persamaan Bellman [12] pada Persamaan (3).

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma\, max_a Q(s',a') - Q(s,a)] \quad (3)$$

dengan,

- s= Kondisi saat ini (*state*)
- s'= Perpindahan agen dari kondisi sebelumnya
- a= Keputusan (*action*)
- a'= Keputusan agen sebelumnya
- γ= *Discount factor* (gamma)
- α= *Learning rate* (alfa)

2.2 Navigasi Robot Beroda

Banyak penelitian sebelumnya yang telah mengkaji sistem navigasi robot beroda penghindar rintangan dengan menggunakan algoritma-algoritma kecerdasan buatan, di antaranya sebagai berikut. Pada penelitian [13], dikembangkan suatu perangkat keras robot penghindar halangan dengan kendali fuzzy. Robot

menggunakan beberapa sensor ultrasonik, dan mengadopsi metode *data-fusion* multi-sensor. Robot dapat memperoleh informasi lingkungan secara akurat, sehingga efektivitas deteksi hambatan robot menjadi lebih baik. Hasil simulasi dan pengujian aktual menunjukkan bahwa sistem kendali fuzzy yang diusulkan aman untuk menghindari rintangan robot, dan selanjutnya robot mampu berjalan dengan jalur yang optimal.

Pada penelitian [14] dikembangkan suatu perangkat keras robot beroda dengan sistem kemudi ackerman steering menggunakan sistem navigasi hibrid fuzzy-*Propotional-Derivatif-Integral* dengan masukan dari multi sensor ultrasonik. Kombinasi fuzzy dan PID diklaim bahwa sistem navigasi dan kemampuan menghindari halangan memiliki performa yang lebih baik dibandingkan dengan hanya kendali PID atau hanya fuzzy.

Samsudin dkk. [15] mengembangkan kendali *fuzzy* untuk navigasi robot dan nilai keanggotaan *fuzzy* dioptimasi dengan metode RL dan algoritma genetik. Kombinasi optimasi nilai keanggotaan tersebut diklaim dapat meningkatkan performa navigasi, walaupun menambah beban komputasi.

Terkait ROA dengan algoritma RL *Q-learning*, Prescott [16] membuat simulasi robot tiga roda RL sederhana pada bidang 2D. Robot dirancang untuk menghindari rintangan dengan menggunakan algoritma RL *Q-learning Learning from delayed rewards* yang dikembangkan oleh Watkins [9]. Aturan untuk fungsi aksi diperoleh dari distribusi probabilitas Gaussian yang sebelumnya pernah dilakukan William [17]. Prescott mengklaim bahwa sistem navigasi dapat berjalan dengan baik dan proses pembelajaran berjalan efektif.

Tzeng [18] mengembangkan ROA-nya dengan fokus pada generalisasi dan aspek praktis. Metode sederhana dikembangkan berbasis RL *Q-Learning* dan *Simultaneous Localization and Mapping* dengan hanya menggunakan informasi lokal. Sistem optimasi *Q-Learning* menggunakan algoritma *e-greedy*. Robot diklaim sukses menghindari rintangan statis dan dinamis serta mencapai target. Robot yang dibangun cukup kompleks, menggunakan Raspberry Pi 3, STM32, sensor lidar, kamera, empat roda *mecanum*.

Ribeiro [19] mengembangkan simulasi ROA RL *QLearning* pada V-rep dengan menggunakan ROS (*Robot Operating System)* sebagai perangkat lunak kontrol. Model robot yang digunakan adalah Botn Roll ONE A dengan dua sensor jarak, masing-masing pada bagian kanan-depan dan kiri-depan. Rintangan bersifat statis yaitu dinding-dinding pada jalur labirin sederhana. Hasil simulasi yang dilakukan membuktikan bahwa robot dapat melakukan navigasi dengan baik.

Jaradat [20], Huang [21], dan Zhang [22] membuat ROA *Q-Learning* pada rintangan dinamis. Namun, Huang hanya melakukan simulasi menggunakan Python dan menggunakan sistem yang sama dengan penelitian Jaradat. Huang menambahkan penentuan *action policy* menggunakan sistem *hibrid selection strategy* dan *e-greedy* klasik secara hybrid, sedangkan Jaradat hanya menggunakan *e-greedy* klasik dan melakukan aktual eksperimen. Zhang melakukan simulasi dengan menggunakan Matlab dan juga melakukan aktual eksperimen dengan menggunakan robot Pioneer 3-DX yang sudah dilengkapi dengan sensor jarak ultrasonik dan laser dan dikoneksikan dengan *Robot Operating System* (ROS). Dari ketiga penelitian ini dapat ditunjukan bahwa sistem ROA pada lingkungan dinamis dapat bekerja dengan baik.

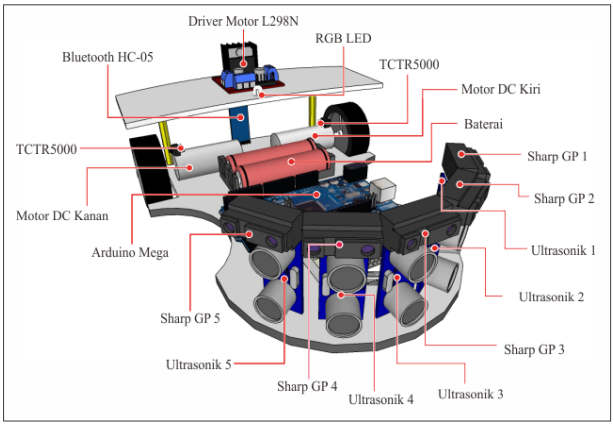
Setianingsih [23] membuat simulasi sistem navigasi autonomous robot beroda dengan teknik penghindar rintangan menggunakan metode *Q-learning, state* dan *action* menggunakan pendekatan fuzzy. Simulasi yang dirancang pada perangkat lunak MobileSim menunjukkan algoritma *Q-learning* berhasil membuat robot menghindari halangan dengan akurasi 85,71%.

Khriji dkk. [24] melakukan simulasi dan eksperimental navigasi robot beroda dengan pendekatan *Q-learning*, *state* dan *action* menggunakan fuzzy, sama seperti [23], dengan *learning rate* dan *discount factor* 0.8. Dari hasil simulasi dan eksperimental Khriji dkk. memastikan bahwa kendali robot beroda dengan algoritma *Q-learning* yang diusulkan mampu belajar dan memilih keputusan terbaik dalam setiap situasi lingkungan.

3. Metode Penelitian

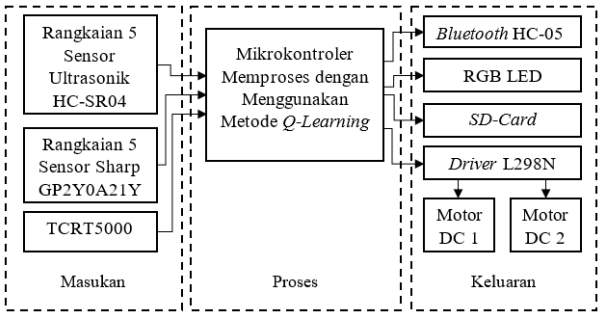
ROA yang dirancang merupakan tipe robot beroda. Robot menggunakan mikrokontroler ATMega2560 pada platform Arduino Mega2560 sebagai pusat kendali. Robot dilengkapi dengan lima sensor ultrasonik HC-SR04 dan lima sensor IR sharp GP2Y0A21YK0F. Desain prototipe robot tersebut ditunjukkan pada Gambar 1. Sensor mendeteksi jarak robot dengan objek penghalang gerak robot yang berada di sekitar robot. Objek yang dideteksi oleh sensor jarak kemudian akan menjadi parameter acuan bahwa di sekitar robot sedang terdapat rintangan. Pembacaan jarak oleh sensor ultrasonik dan IR bersifat hibrid per nomor sensor, ditambahkan metode deteksi untuk memilih bacaan yang akurat.

Selain sensor ultrasonik, robot juga dilengkapi sensor TCRT5000 yang berfungsi sebagai encoder yang akan mengubah masukan sinyal pulsa untuk diolah menjadi parameter kecepatan putar roda. Mikrokontroler selanjutnya akan memproses keluaran sensor menggunakan metode *Q-learning*. Keluaran sistem adalah *driver motor* L298N yang akan menggerakkan motor DC 1 dan motor DC 2 sehingga pergerakan robot akan lebih terkendali. RGB LED digunakan untuk



Gambar 1. Desain prototipe robot beroda *Q-Learning*

indikator yang mewakilkan kondisi robot. Tabel matriks dari hasil pembelajaran metode *Q-learning* akan disimpan di EEPROM dan *sd-card* agar dapat dianalisa dan dapat menyimpan data saat perangkat dihidupkan kembali. Bluetooth HC-05 akan mengirimkan data parameter yang dibutuhkan ke perangkat *smart phone* agar dapat dipantau secara *portable* dan *realtime*. Skema alur sistem pada penelitian ini ditunjukkan pada Gambar 2.

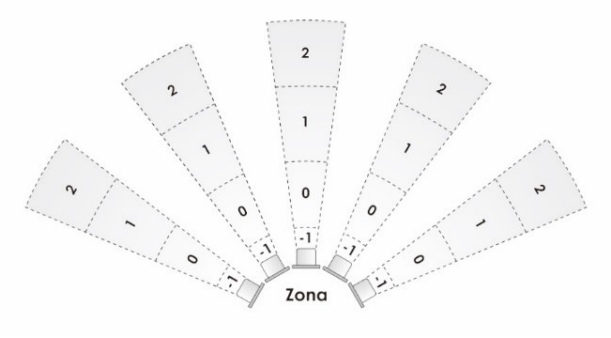


Gambar 2. Diagram Blok Sistem

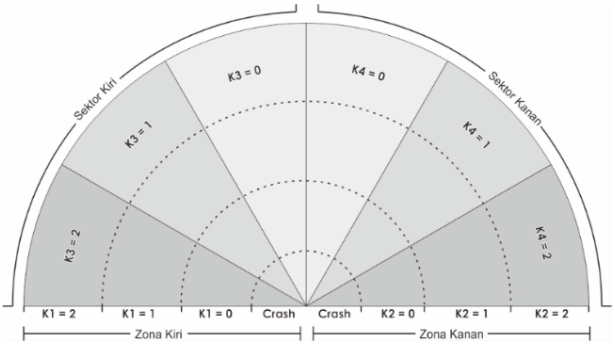
Untuk meringankan beban komputasi, pembacaan sensor hanya dibagi menjadi empat zona berdasarkan jarak dengan rincian sebagaimana ditunjukkan pada Persamaan (4), pada Gambar 3

$$zona = \begin{cases} -1 & (jarak) <= \\ 0 & 10 \leq (jarak) \leq 35 \\ 1 & 35 \leq (jarak) \leq 70 \\ 2 & (jarak) > 70 = \end{cases} \quad (4)$$

Pembacaan multi-sensor jarak, dibagi menjadi empat state yaitu zona kiri (k1), zona kanan (k2), sektor kiri (k3) dan sektor kanan (k4). Zona multi-sensor menunjukkan posisi rintangan terbaca berada di kiri atau kanan robot, sedangkan sektor menunjukkan posisi sudut



Gambar 3. Zona pembacaan sensor jarak



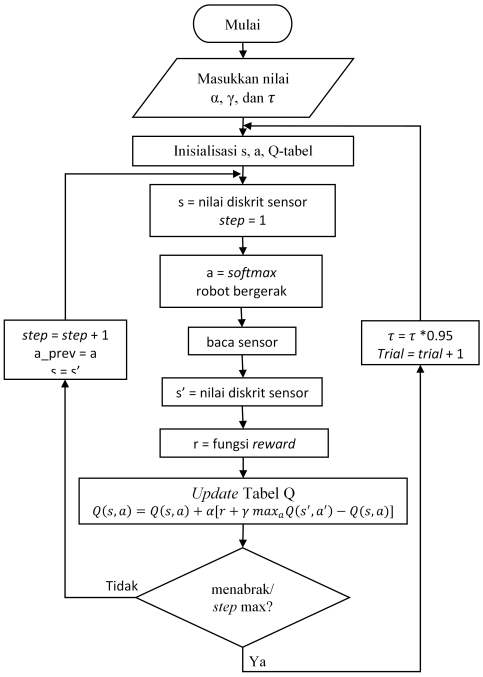
Gambar 4. Pembagian Zona dan Sektor

rintangan terbaca. Nilai posisi k1 = [0;1;2], k2 = [0;1;2], k3 = [0;1;2;3], k4 = [0;1;2;3] seperti ditunjukkan pada Gambar 4. Total kemungkinan *state* adalah 144, merupakan *Cartesian product* dari k1, k2, k3 dan k4. dengan action lurus, kanan dan kiri, seperti ditunjukkan Tabel 1.

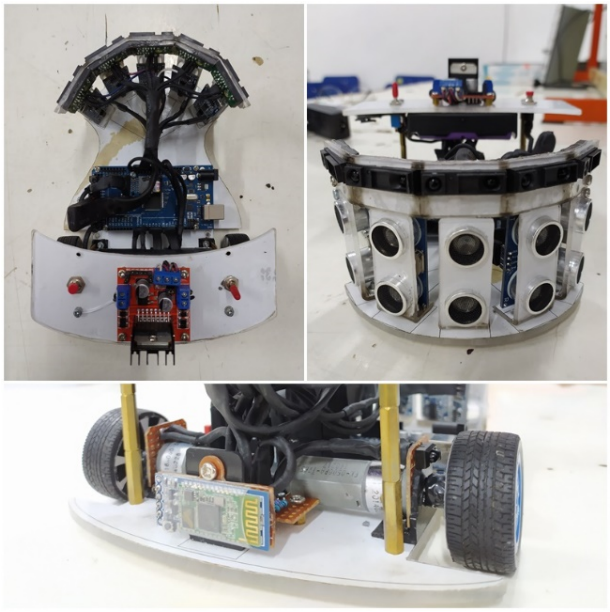
Proses algoritma *Q-learning* yang diusulkan pada penelitian ini ditunjukkan pada Gambar 5. Proses pembelajaran dilakukan untuk mengumpulkan nilai *reward* yang akan disimpan pada Tabel-Q, proses dimulai dengan memberikan parameter *learning rate* (α), *discount factor* (γ) dan temperatur. Kemudian dilakukan inisialisasi nilai s, a pada Tabel-Q. Dilanjutkan dengan pembacaan nilai sensor sehingga nilai state

Tabel 1. Tabel *Q-learning*

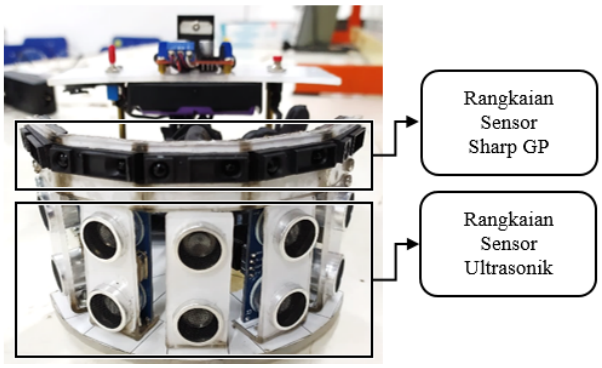
No	*State*				*Action*		
1	k1	k2	k3	k4	Lurus	Kanan	Kiri
2	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0
4	0	0	0	2	0	0	0
5	0	0	1	0	0	0	0
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
142	2	2	2	0	0	0	0
143	2	2	2	1	0	0	0
144	2	2	2	2	0	0	0



Gambar 5. Diagram alir *Robot Q-learning*



Gambar 6. Hasil rancangan perangkat keras prototipe Robot beroda *Q-learning*



Gambar 7. Penempatan Sensor jarak pada Qlobot

didapat dan mengambil nilai action menggunakan fungsi *softmax*. Setelah *action* dijalankan, nilai *state* baru dibaca dan *reward* diberikan. Selanjutnya nilai Tabel-Q diperbaharui dengan menggunakan persamaan Bellman. Jika robot menabrak atau jumlah langkah mencapai maksimal maka nilai τ akan diperbaharui dengan dikalikan 0,95 selanjutnya proses diulang dari inisialisasi Tabel-Q. Jika robot tidak menabrak, maka jumlah step bertambah dan nilai Q(s, a) menjadi Q(s', a'). Selanjutnya, proses diulang dari pembacaan sensor yang akan menjadi nilai parameter *state* baru. Saat percobaan telah mencapai status terminal dilanjutkan dengan variasi nilai α dan γ.

4. Hasil dan Pembahasan

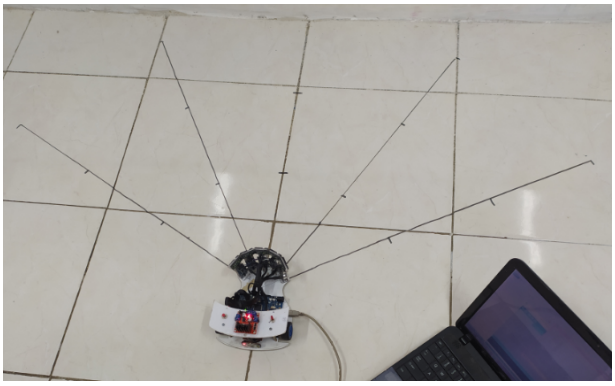
Perangkat keras Robot *Q-learning* yang berhasil dirancang pada penelitian ini ditunjukkan pada Gambar 6. Sebelum digunakan, komponen-komponen perangkat keras telah dilakukan pengujian kelayakan penggunaan, sedangkan penempatan posisi sensor jarak ditunjukkan pada Gambar 7. Gambar 8 menunjukkan skenario eksperimen pembacaan sensor dan pembagian zona dan sektor seperti yang ditunjukkan pada Gambar 3 dan Gambar 4 sebelumnya. Video eksperimen dapat dilihat pada laman https://youtu.be/PUqCMa7AfZ8

4.1 Penentuan Parameter Optimal *Q-Learning*

Penentuan parameter *Q-learning* optimal pada robot *obstacle avoidance* dilakukan untuk mendapatkan hasil pembelajaran terbaik. Parameter penentu tersebut yaitu α dan γ kombinasi dari parameter yang menghasilkan pembelajaran paling efektif dan *error* paling kecil akan digunakan untuk pengujian ke tahap selanjutnya. **Pelatihan pada Perubahan Nilai *Learning Rate***

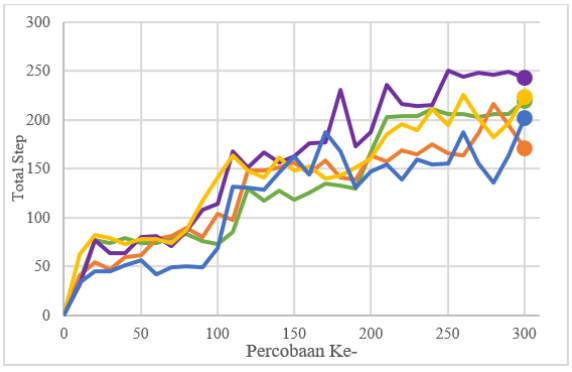
Untuk pelatihan ini dilakukan percobaan sebanyak 300 kali. Setiap percobaan memiliki kondisi terminal, yaitu ketika menabrak atau telah berhasil bergerak sebanyak 250 langkah. Pelatihan ini menggunakan parameter *discount factor* (γ) dinamis untuk mempercepat proses pembelajaran yaitu dengan nilai awal γ 0,1 dan akan terus meningkat hingga γ 0,9 pada percobaan ke-150. Nilai *learning rate* (α) yang divariasikan pada percobaan ini adalah 0,1; 0,3; 0,5; 0,7; 0,9.

Hasil pelatihan variasi *learning rate*



Gambar 8. Skenario zona pembagian zona dan sektor

divisualisasikan dengan grafik sehingga dapat dianalisis perbedaan pada masing-masing variasi. Hasil perolehan step pada pelatihan dengan variasi nilai learning rate dapat dilihat pada Gambar 9. Jumlah nilai step yang didapat pada masing-masing variasi mengalami fluktuasi namun cenderung naik pada tahap awal hingga percobaan akhir. Dari seluruh pengujian, dapat terlihat bahwa belum terdapat nilai yang mencapai konvergen, seluruh variasi *learning rate* tersebut menghasilkan grafik yang belum mencapai titik terminal step yaitu 250 kecuali pada variasi *learning rate* 0,5. Maka dari itu, dari seluruh variasi terlihat grafik perolehan step dengan learning rate 0,5 menunjukkan nilai step yang cenderung lebih cepat stabil. Grafik perbandingan perolehan nilai



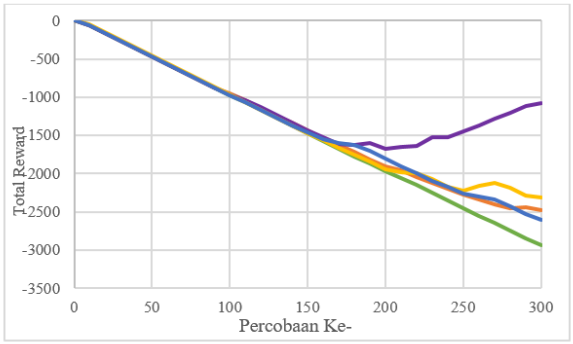
Keterangan:
1. Hijau (*Learning Rate* 0,1)
2. Jingga (*Learning Rate* 0,3)
3. Ungu (*Learning Rate* 0,5)
4. Kuning (*Learning Rate* 0,7)
5. Biru (*Learning Rate* 0,9)

Gambar 9. Grafik Perbandingan Perolehan Jumlah Step dengan Variasi Nilai *Learning Rate*

reward setiap variasi *learning rate* dapat dilihat pada Gambar 10. *Reward* yang didapat dari awal hingga akhir percobaan menunjukkan grafik yang menurun, kecuali pada saat variasi α 0,5. Pada variasi tersebut robot berhasil mencapai titik konvergen sehingga terus menerima nilai *reward* positif. Berbeda dengan percobaan lain yang terus mendapat akumulasi *reward* negatif. Berdasarkan grafik tersebut dapat dilihat bahwa nilai parameter α=0,5 memiliki perolehan nilai *reward* yang paling tinggi. **Pelatihan pada Perubahan Nilai *Discount Factor***

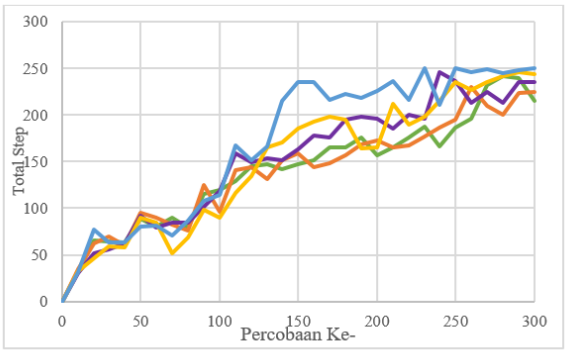
Pelatihan ini dilakukan langkah yang sama dengan percobaan sebelumnya namun dengan parameter nilai α 0,5 dan variasi nilai *discount factor* (γ). Adapun nilai (γ) yang divariasikan pada percobaan ini adalah γ=0,1; 0,3; 0,5; 0,7 dan 0,9. Hasil perolehan step pada pelatihan dengan variasi nilai *discount factor* dapat dilihat pada Gambar 11. Jumlah nilai step yang didapat pada masing-masing variasi mengalami fluktuasi namun cenderung naik pada tahap awal hingga percobaan akhir. Dari seluruh variasi terlihat grafik perolehan step dengan *discount factor* 0,9 menunjukkan nilai step yang cenderung lebih cepat stabil menuju titik konvergen pada percobaan ke 250.

Grafik perbandingan perolehan nilai *reward* setiap variasi discount factor dapat dilihat pada Gambar 12. Reward yang didapat pada seluruh variasi di awal percobaan menunjukkan grafik yang menurun, kemudian nilai tersebut berangsur naik mulai pada percobaan ke 130. Artinya robot tersebut melakukan eksplorasi pada tahap awal percobaan untuk mengumpulkan seluruh kemungkinan yang bisa dilalui. Berdasarkan grafik tersebut juga dapat dilihat bahwa nilai parameter γ= 0,9 dianggap memiliki respons yang paling optimal karena memiliki perolehan *reward* yang paling tinggi dan langkah yang paling banyak. Maka berdasarkan pengujian dengan variasi parameter telah ditentukan nilai parameter optimum adalah *learning rate* 0,5 dan *discount factor* 0,9.



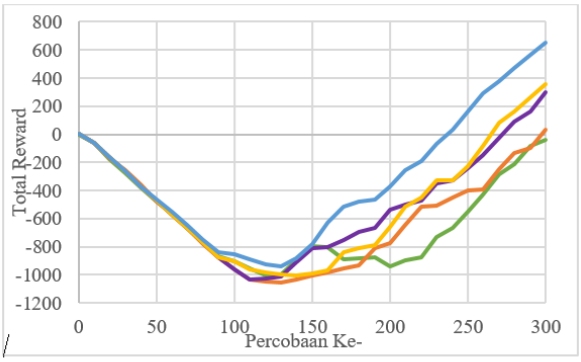
Keterangan:
1. Hijau (*Learning Rate* 0,1)
2. Jingga (*Learning Rate* 0,3)
3. Ungu (*Learning Rate* 0,5)
4. Kuning (*Learning Rate* 0,7)
5. Biru (*Learning Rate* 0,9)

Gambar 10. Grafik perbandingan perolehan *reward* dengan variasi nilai *learning rate*



Keterangan:
1. Hijau (*Disc Factor* 0,1)
2. Jingga (*Disc Factor* 0,3)
3. Ungu (*Disc Factor* 0,5)
4. Kuning (*Disc Factor* 0,7)
5. Biru (*Disc Factor* 0,9)

Gambar 11. Grafik Perbandingan Perolehan Step Dengan Variasi Nilai *Discount Factor*



Keterangan:
1. Hijau (*Disc Factor* 0,1)
2. Jingga (*Disc Factor* 0,3)
3. Ungu (*Disc Factor* 0,5)
4. Kuning (*Disc Factor* 0,7)
5. Biru (*Disc Factor* 0,9)

Gambar 12. Grafik perbandingan perolehan *reward* dengan variasi nilai *discount factor*

Tabel 2. Tabel *Q-learning* Optimal

No	*State*				*Action*		
	k1	k2	k3	k4	Lurus	Kanan	Kiri
1	0	0	0	0	-11.097	0.0088	-9.6387
2	0	0	0	1	-7.8321	-0.4898	-5.116
3	0	0	0	2	0.0075	-0.9874	-4.9952
4	0	0	0	3	-7.5175	0.0068	-6.7582
5	0	0	1	0	-5.2043	-1.1602	-1.2882
6	0	0	1	1	-6.8639	-1.1676	-5.7672
7	0	0	1	2	0.0787	-0.0825	0.0083
8	0	0	1	3	-0.4085	-5	-4.9952
9	0	0	2	0	0	-1.0807	0.0657
:	:	:	:	::	⋮	⋮	⋮
141	2	2	3	0	-0.0037	-0.0007	0.0012
142	2	2	3	1	0.1126	-0.007	-0.1008
143	2	2	3	2	-2.0739	0.0313	-0.0772
144	2	2	3	3	0.0688	-4.891	-4.9805

Tabel *Q-learning* hasil pelatihan optimal terakhir dengan *learning rate* 0,5 dan *discount factor* 0,9 ditunjukkan pada Tabel 2. Nilai-nilai yang ada pada kolom action adalah total *reward* yang diterima sistem selama melakukan eksplorasi. Nilai *reward* tertinggi merupakan aksi yang akan dieksekusi oleh algoritma *Q-learning* pada saat menghadapi *state* tersebut. Nilai hasil pelatihan tersebut selanjutnya disimpan pada memori mikrokontroler agar dapat dieksekusi pada pengujian selanjutnya. Nilai *reward* ini akan terus diperbaiki jika didapatkan suatu kondisi yang meningkatkan nilai reward pada setiap eksplorasi.

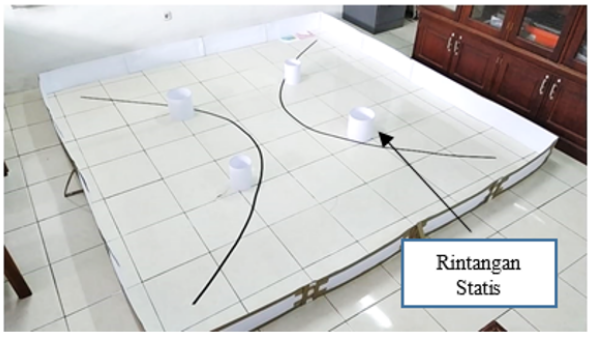
4.2 Pengujian Robot Pada Variasi Arena

Pengujian selanjutnya dilakukan untuk mengetahui kemampuan robot dalam beradaptasi dengan arena yang berbeda. Hal ini dikarenakan pada RL, robot akan terus melakukan pembelajaran pada lingkungan yang terus berubah.

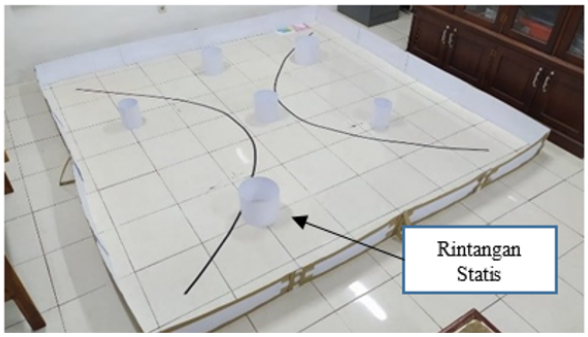
Pada pengujian sebelumnya, telah dilakukan pengujian dengan menggunakan arena A yang ditunjukkan Gambar 13(a), sehingga didapat parameter optimal dan tabel hasil pelatihan di antaranya adalah *learning rate* 0,5; *discount factor* 0,9. Beberapa parameter hasil pelatihan tersebut akan digunakan kembali pada pengujian ini sehingga kondisi awal Tabel *Q-learning* telah terisi *reward* hasil eksplorasi sebelumnya.

Pengujian pertama dilakukan dengan menjalankan robot pada arena A yang berisikan 4 rintangan statis. Percobaan kedua dilakukan dengan mengubah susunan rintangan menjadi arena B yang berisikan 6 rintangan statis. Pada percobaan ke 3 dilakukan dengan mengubah susunan rintangan menjadi arena C yang berisikan 4 rintangan statis dan 2 rintangan dinamis, sehingga posisi rintangan akan terus berubah selama robot dijalankan.

Seluruh percobaan dilakukan sebanyak 100 kali dengan limit 250 *step*. Adapun hasil pengujian robot pada variasi arena dapat dilihat pada Gambar 14. Seluruh variasi arena mengalami fluktuasi pada awal percobaan dan menuju titik konvergen dengan nilai yang berbeda-beda. Pada arena A, robot lebih cepat mencapai titik konvergen karena arena tersebut merupakan tempat robot dilatih sebelumnya. Pada arena B yang merupakan arena statis, robot membutuhkan beberapa kali percobaan untuk menuju kondisi konvergen sehingga mampu mencapai kondisi terminal pada percobaan ke 20 dan seterusnya. Sedangkan pada arena C yang merupakan arena dinamis, robot membutuhkan waktu yang cukup lama untuk beradaptasi yaitu dapat mencapai



(a)



(b)

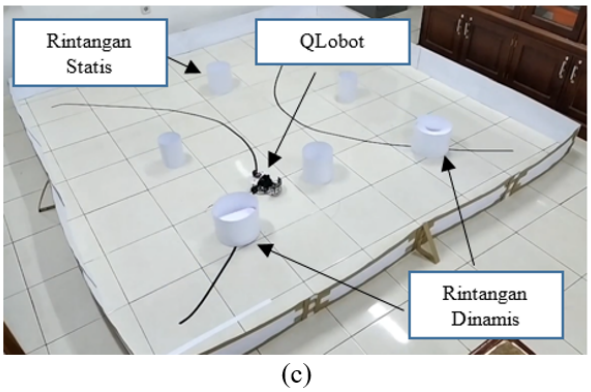
konvergen pada percobaan ke 60 dan seterusnya. Arena B dan C ditunjukkan pada Gambar 13(b) dan Gambar 13(c).

Berdasarkan data pada Tabel 3 dapat diketahui perolehan nilai akumulasi pada arena A memiliki nilai *step* dan *reward* paling banyak, kemudian diikuti dengan perolehan step dan *reward* di arena B dan C. Namun, dari keseluruhan pengujian robot masih mengalami *error* dengan mengalami tabrakan, yaitu dengan jumlah tabrakan terbanyak pada arena C sebanyak 32 kali. Hal tersebut dikarenakan robot sudah memiliki pengalaman bereksplorasi di kondisi arena A, sedangkan pada variasi arena B dan C, robot memerlukan lebih banyak waktu untuk mengenali lingkungan baru.

Data pada Tabel 4 menunjukkan bahwa robot mengalami *error* pada awal pengujian. *Error* yang dialami pada arena A sebesar 10% pada 10 percobaan pertama. Pada arena B robot mengalami error hingga

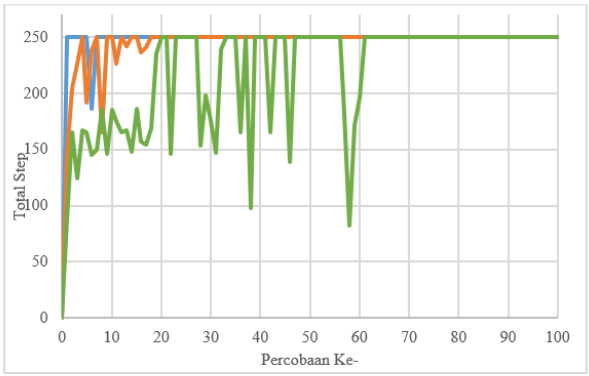
Tabel 3. hasil pengujian dengan variasi arena

No	Arena	Akumulasi *Step*	Akumulasi *Reward*	Akumulasi *Crash*
1	A	24758	923.9	1
2	B	24400	743.77	10
3	C	22084	570.94	32



(c)

Gambar 13. Variasi Eksperimen (a) arena A, (b) arena B, (c) arena C

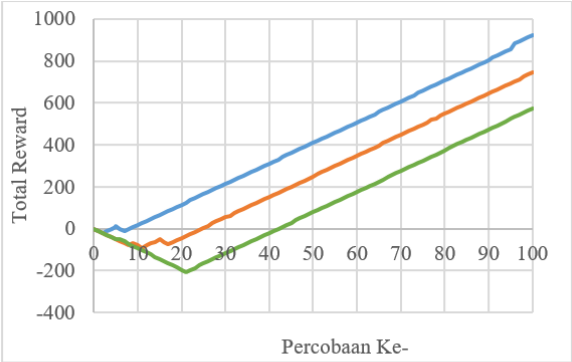


(a)

percobaan ke 20, sedangkan pada arena C, robot mengalami error hingga percobaan ke-60. Penyebab *error* tersebut tentunya penyesuaian robot ketika menghadapi lingkungan yang berbeda, pada saat robot mendeteksi rintangan baru, aksi robot akan bertindak sesuai dengan aksi terbaik pada tabel *Q-learning* hasil pembelajaran sebelumnya.

5. Kesimpulan

Pada penelitian ini dirancang sebuah prototipe robot beroda dengan kendali navigasi penghindar rintangan menggunakan algoritma *Q-learning*. Nilai parameter optimal *learning rate* 0,5 dan *discount factor* 0,9. Dengan 300 kali percobaan didapat total langkah 49999, total *reward* 650,064, dan total crash 141. Robot mampu mencapai kondisi konvergen pada percobaan ke 250. pada saat robot mendeteksi rintangan baru, robot akan bertindak sesuai dengan aksi terbaik pada tabel Q-learning hasil pembelajaran sebelumnya dan akan memperbaiki nilai *reward* untuk beradaptasi secara realtime. Dari uji performa, pada rintangan statis arena A, robot masih mengalami tabrakan dalam 10 kali



(b)

Keterangan:

1. Biru (Arena A), 2. Jingga (Arena B), 3. Hijau (Arena C)

Gambar 14. (Grafik Perbandingan Perolehan (a) Jumlah Step (b) Jumlah *Reward* Pengujian Robot dengan Variasi Arena.)

Tabel 4. Data *error* hasil pengujian robot *obstacle avoidance* pada variasi arena

No	Percobaan Ke-	Eror(%)		
		Arena A	Arena B	Arena C
1	1-10	10	60	100
2	11-20	0	40	90
3	21-30	0	0	40
4	31-40	0	0	40
5	41-50	0	0	20
6	51-60	0	0	40
7	61-70	0	0	0
8	71-80	0	0	0
9	81-90	0	0	0
10	91-100	0	0	0

percobaan dengan eror 10%, dan setelah itu robot sama sekali tidak mengalami tabrakan. Pada arena B, robot baru berhasil menghindari rintangan setelah 20 kali percobaan, 10 percobaan sebelumnya masih terdapat error 40%. Hal ini disebabkan karena rintangan lebih sulit dari arena A. Pada arena C, robot baru bisa menghindari rintangan setelah 60 kali percobaan, dan 10 percobaan sebelumnya robot masih mengalami tabrakan dengan *error* 40%. Secara keseluruhan ROA yang dirancang menunjukkan performa yang sangat baik, namun untuk rintangan yang lebih kompleks, robot perlu kembali belajar dan beradaptasi.

Daftar Pustaka

[1] Y. Away, R. Munadi, M. Ikhsan, I. Muddin *et al.*, "Perancangan lengan robot 5 derajat kebebasan dengan pendekatan kinematika," *Jurnal Rekayasa Elektrika*, vol. 11, no. 2, pp. 69–72, 2014.

[2] J. G. Keramas, *Robot technology fundamentals*. Delmar Publishers, 1999.

[3] M. Mizukawa, "Robot technology (rt) trend and standardization," in *IEEE Workshop on Advanced Robotics and its Social Impacts, 2005.* IEEE, 2005, pp. 249–253.

[4] P. Lin, K. Abney, and R. Jenkins, *Robot ethics 2.0: From autonomous cars to artificial intelligence*. Oxford University Press, 2017.

[5] N. J. Nilsson, *Principles of artificial intelligence*. Morgan Kaufmann, 2014.

[6] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, "Machine learning, neural and statistical classification," 1994.

[7] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[8] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[9] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.

[10] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[11] N. Metropolis and S. Ulam, "The monte carlo method," *Journal of the American statistical association*, vol. 44, no. 247, pp. 335–341, 1949.

[12] R. E. Bellman and S. E. Dreyfus, *Applied dynamic programming*. Princeton university press, 2015, vol. 2050.

[13] J. Yi, X. Zhang, Z. Ning, and Q. Huang, "Intelligent robot obstacle avoidance system based on fuzzy control," in *2009 First International Conference on Information Science and Engineering*. IEEE, 2009, pp. 3812–3815.

[14] A. T. Kusuma, A. Indra, H. Faisal, and S. Agus, "Sistem kendali fuzzy-pid pada robot wall follower ackerman steering," *AMPLIFIER Jurnal Ilmiah BidangTeknik Elektro dan Komputer*, vol. 7, no. 1, pp. 1–5, 2017.

[15] K. Samsudin, F. A. Ahmad, and S. Mashohor, "A highly interpretable fuzzy rule base using ordinal structure for obstacle avoidance of mobile robot," *Applied Soft Computing*, vol. 11, no. 2, pp. 1631–1637, 2011.

[16] T. J. Prescott and J. E. Mayhew, "Obstacle avoidance through reinforcement learning," in *Advances in neural information processing systems*. Citeseer, 1992, pp. 523–530.

[17] R. J. Williams, *Reinforcement-learning connectionist systems*. College of Computer Science, Northeastern University, 1987.

[18] E. J. Tzeng, E. Yang, S. Chen, and J. Chen, "A practical obstacle avoidance method using q-learning with local information," in *IFToMM World Congress on Mechanism and Machine Science*. Springer, 2019, pp. 2149–2158.

[19] T. Ribeiro, F. Gonçalves, I. Garcia, G. Lopes, and A. F. Ribeiro, "Q-learning for autonomous mobile robot obstacle avoidance," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2019, pp. 1–7.

[20] M. A. K. Jaradat, M. Al-Rousan, and L. Quadan, "Reinforcement based mobile robot navigation in dynamic environment," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 135–149, 2011.

[21] L. Huang, H. Qu, M. Fu, and W. Deng, "Reinforcement learning for mobile robot obstacle avoidance under dynamic environments," in *Pacific Rim International Conference on Artificial Intelligence*. Springer, 2018, pp. 441–453.

[22] Y. Zhang, X. Wei, and X. Zhou, "Dynamic obstacle avoidance based on multi-sensor fusion and q-learning algorithm," in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. IEEE, 2019, pp. 1569–1573.

[23] C. Setianingsih, K. Mutijarsa, and M. A. Murti, "Simulasi sistem kendali berbasis perilaku pada autonomous mobile robot dengan metoda q-learning," *TEKTRIKA-Jurnal Penelitian dan Pengembangan Telekomunikasi, Kendali, Komputer, Elektrik, dan Elektronika*, vol. 4, no. 2, pp. 54–61, 2019.

[24] L. Khriji, F. Touati, K. Benhmed, and A. Al-Yahmedi, "Mobile robot navigation based on q-learning technique," *International Journal of Advanced Robotic Systems*, vol. 8, no. 1, p. 4, 2011.