

# SIMULASI SISTEM KENDALI BERBASIS PERILAKU PADA *AUTONOMOUS MOBILE ROBOT* DENGAN METODA *Q-LEARNING*

Casi Setianingsih<sup>1</sup>, Kusprasapta Mutijarsa<sup>2</sup>, Muhammad Ary Murti<sup>3</sup>

<sup>1</sup>Teknik Komputer <sup>3</sup>Teknik Elektro<sup>1,3</sup>Fakultas Teknik Elektro, Universitas Telkom,  
Bandung, Indonesia

<sup>2</sup>Teknik Elektro <sup>2</sup>Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung,  
Bandung, Indonesia

<sup>1</sup>setiacasie@telkomuniversity.ac.id, <sup>2</sup>soni@stei.itb.ac.id, <sup>3</sup>arymurti@telkomuniversity.ac.id

---

## Abstrak

*Autonomous robot* adalah suatu robot yang mampu bekerja secara mandiri tanpa pengendalian langsung dari manusia. Robot bekerja berdasarkan sensor-sensor yang dimilikinya, mengambil keputusan sendiri untuk menyelesaikan misi dalam lingkungan kerjanya. Dalam dunia nyata, lingkungan kerja robot sangat dinamis, selalu berubah, dan tidak terstruktur. Membuat suatu model lingkungan yang tidak terstruktur sangat sulit. Memperoleh model matematik yang tepat dari lingkungan seperti ini hampir tidak mungkin dilakukan. Untuk membuat suatu *autonomous mobile robots* yang mampu bekerja pada lingkungan yang tidak terstruktur dan dinamis, diperlukan suatu metoda tertentu yang adaptif dan mampu belajar.

Berdasarkan permasalahan tersebut maka pada riset ini dirancang suatu *autonomous mobile robot* dengan arsitektur berbasis perilaku yang dapat belajar dan bekerja secara mandiri pada lingkungan yang tidak terstruktur, menggunakan metoda *Reinforcement Learning*. Tujuan metoda ini diterapkan agar robot mampu belajar dan beradaptasi terhadap lingkungan yang tidak terstruktur. Selanjutnya robot dikembangkan agar mampu menyelesaikan misi menemukan target pada posisi tertentu berdasarkan informasi yang diperoleh dari sensor sensor yang ada. Hasil simulasi menunjukkan bahwa algoritma pembelajaran *Reinforcement Learning* berhasil diterapkan pada arsitektur kendali berbasis perilaku di *autonomous mobile robot* dengan akurasi sebesar 85,71%.

**Kata Kunci:** *autonomous mobile robot*, arsitektur kendali berbasis perilaku, *reinforcement learning*, *autonomous robot*.

---

## Abstract

*Autonomous robots* are robots that are able to work independently without direct control from humans. Robots work based on the sensors they have, making their own decisions to complete the mission in their work environment. In the real world, the robot's working environment is very dynamic, always changing, and unstructured. Creating an unstructured environment model is very difficult. Obtaining the right mathematical model of this environment is almost impossible. To create an autonomous mobile robots that are capable of working in an unstructured and dynamic environment, certain methods are needed that are adaptive and capable of learning.

This research focus on design of an autonomous mobile robot with a behavior-based architecture that can learn and work independently in an unstructured environment, using the Reinforcement Learning method. The purpose of this method is to be implemented so that robots are able to learn and adapt to unstructured environments. Furthermore, the robot was developed to be able to complete the mission of finding targets in certain positions based on information obtained from existing sensors. The simulation results showed that the Reinforcement Learning algorithm was successfully applied to a behavior-based control architecture in the autonomous mobile robot with an accuracy of 85.71%.

**Key Words:** *autonomous mobile robot*, behavior-based control architecture, reinforcement learning, *autonomous robot*.

---

## 1. Pendahuluan

*Autonomous robot* atau biasa disebut robot independen adalah robot yang menggabungkan sensor, aksi, catu daya, komunikasi, teori kontrol dan kecerdasan buatan. Robot independen dirancang untuk melakukan tugas di dunia nyata [1]. Robot independen harus dikontrol secara mandiri berdasarkan informasi yang diperoleh dari sensor (*sensor driven*) dan memungkinkan robot untuk bereaksi terhadap peristiwa yang tidak diinginkan ketika sedang bergerak dan berinteraksi dengan lingkungannya. Dukungan kontrol reaktif harus memperhitungkan gangguan yang terjadi pada waktu robot berjalan dan dapat memperbaiki dirinya sendiri terhadap gangguan yang tidak dapat dimodelkan tanpa perlu perencanaan ulang. Perubahan besar dalam lingkungan dan perubahan dalam misi/tugas terus menerus ketika robot berjalan sangat mungkin terjadi, robot independen harus dapat mengubah kebijakan kontrolnya untuk menyesuaikan dengan kondisi lingkungan dan misi/tugas baru. Ini membutuhkan arsitektur sistem kontrol adaptif. Hal ini menjadi alasan pembelajaran mesin diperlukan untuk membantu robot belajar tentang lingkungan mereka agar dapat beradaptasi dengan perubahan. Untuk dapat mengatasi masalah tersebut, dapat digunakan algoritma pembelajaran mesin yaitu *Reinforcement Learning*.

Pada penelitian ini, untuk dapat menerapkan arsitektur kendali berbasis perilaku pada *autonomous mobile robot* sehingga robot mampu belajar dan beradaptasi terhadap lingkungan yang tidak terstruktur digunakan algoritma pembelajaran RL. Sehingga robot mampu menyelesaikan misi menemukan target pada posisi tertentu berdasarkan sensor-sensor yang dimilikinya pada lingkungan yang dinamis tanpa menabrak halangan.

RL digunakan terkait dengan proses pembelajaran untuk mengendalikan sistem sehingga dapat memaksimalkan kinerja secara numerik yang dapat digunakan untuk menunjukkan tujuan jangka panjang. RL memprediksi dan memiliki efek jangka panjang dengan mempengaruhi keadaan sistem yang sedang dikontrol kedepannya [2]. Tujuan RL adalah agar robot mampu memilih tindakan yang tepat sehingga hadiah yang diterima dapat dimaksimalkan [3]. Salah satu algoritma yang digunakan untuk memecahkan masalah RL adalah *Q-learning* [4]. Pada penelitian ini dibutuhkan *Fuzzy* untuk menyajikan perkiraan pada serangkaian *state* kontinu [5]. Metoda ini biasa disebut dengan *Fuzzy Q Learning* seperti yang pernah dilakukan oleh Glonrennec [6].

Setelah robot dapat bergerak tanpa menabrak rintangan, tujuan penelitian ini selanjutnya adalah untuk dapat menemukan target yang berada di posisi tertentu. Untuk menyelesaikan tugas ini robot membutuhkan bantuan sensor posisi yaitu Odometri. Odometri dapat

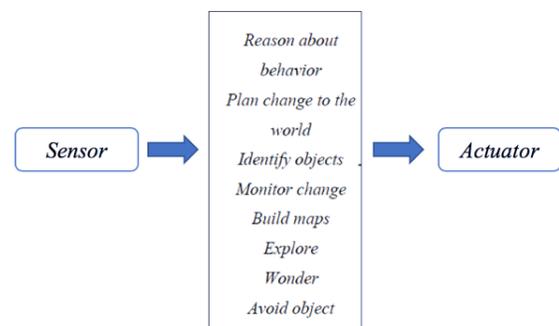
memetakan posisi robot pada sumbu *Cartesian*, sehingga dapat diketahui posisi robot dalam bentuk koordinat dan arah hadap robot [7].

Hasil simulasi menunjukkan bahwa algoritma pembelajaran RL berhasil diterapkan pada arsitektur kendali berbasis perilaku di *autonomous mobile robot* dengan akurasi sebesar 85,71%.

## 2. Tinjauan Pustaka

### 2.1 Behavior-based Robotic

*Behavior-Based Robotic* (BBR) menggunakan pendekatan *task achieving behaviors* (perilaku menunaikan tugas) (Gambar 1), tiap tugas disebut dengan perilaku (*behavior*) [8].



**Gambar 1. Dekomposisi sistem kendali *mobile robot* dengan *task achieving behaviors***

### 2.2 Metoda Koordinasi Antar Perilaku

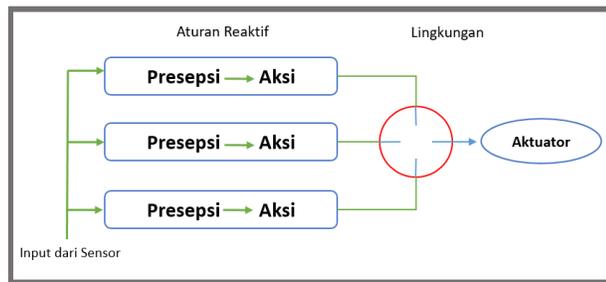
Perilaku-perilaku dengan tujuan yang berbeda pada satu robot dapat menyebabkan masalah baru. Diperlukan sebuah koordinasi yang efektif dari setiap aksi perilaku-perilaku sehingga terbentuk suatu perilaku yang logis, rasional dan tidak saling berlawanan. Permasalahan ini biasa disebut dengan *action selection problem* atau *behavior coordination problem* [9].

Secara umum, ada dua cara untuk menentukan perilaku mana yang akan dipilih untuk dijalankan. Pertama adalah dengan memilih salah satu dari perilaku yang ada dan yang kedua adalah dengan menggabungkan perilaku-perilaku yang ada [10].

### 2.3 Arbitrasi (kompetitif)

Arbitrasi atau disebut koordinasi perilaku kompetitif adalah proses pemilihan satu perilaku dari beberapa kemungkinan perilaku yang ada. Pada sistem ini banyak kandidat perilaku yang bersaing, namun hanya ada satu yang dipilih untuk dijalankan. Arbitrasi dapat dilakukan dengan hirarki prioritas tetap atau hirarki dinamis sesuai kebutuhan. Sistem kendali berbasis perilaku sering menggunakan arbitrasi dinamis

untuk menentukan perilaku mana yang akan dipilih berikutnya (Gambar 2) [10].



Gambar 2. Koordinasi perilaku arbitrase

2.4 Reinforcement Learning (RL)

Reinforcement Learning (RL) adalah paradigma pembelajaran yang terkait dengan proses pembelajaran untuk mengendalikan sistem sehingga dapat memaksimalkan kinerja secara numerik yang dapat digunakan untuk menunjukkan tujuan jangka panjang. Hal yang membedakan RL dari Supervised Learning adalah pada proses prediksi, learner hanya membutuhkan sebagian feedback saja yang harus ia terima kembali. Selain itu, prediksi dapat memiliki efek jangka panjang dengan mempengaruhi keadaan sistem yang sedang dikontrol kedepannya [2]. Dalam RL, robot belajar melalui interaksi dengan lingkungannya. Robot menerima keadaan lingkungan mereka dan memilih tindakan yang tepat untuk diberikan kepada lingkungan mereka. Kemudian keadaan lingkungan berubah dan robot menerima reward dari lingkungannya. Tujuan pembelajaran adalah memilih tindakan yang tepat sehingga hadiah yang diterima dapat dimaksimalkan [3].

Reinforcement learning (RL) adalah pendekatan belajar dengan trial and error untuk mencapai tujuan. Dasar belajar yang digunakan yaitu interaksi antara agen dan lingkungan. Algoritma RL tidak menggunakan data latih dan uji seperti teknik belajar dengan pengawasan, tetapi yang diperlukan adalah reward yang diberikan oleh lingkungannya pada setiap aksi yang dijalankan oleh agen. Reward ini bertujuan untuk menguji state lingkungan. Masalah RL adalah bagaimana memaksimalkan jumlah reward yang didapat setiap agen memilih satu aksi.

2.5 Q Learning

Salah satu algoritma yang digunakan untuk memecahkan masalah RL adalah Q-learning [4]. Pada bentuk asli, algoritma ini membutuhkan lingkungan dan agen untuk mengambil nilai dalam set diskrit yang relatif kecil. Fuzzy menyajikan perkiraan, model RL tipe bebas ini telah diusulkan dalam literatur untuk kasus-kasus yang lebih sulit di mana serangkaian tindakan state adalah kontinu [5]. Dalam aplikasi robotik, state dan data hasil sensor bersifat kontinu serta data output

sensor yang diproses tidak terbatas, sehingga Q-learning diskrit tidak dapat digunakan. Oleh karena itu, dalam penelitian ini, logika fuzzy ditambahkan untuk membuat pendekatan pada fungsi Q yang dikenal sebagai fuzzy Q-learning seperti yang dilakukan oleh Glonrennec [6].

Q-learning adalah algoritma temporal difference yang dirancang untuk memecahkan permasalahan Reinforcement Learning dengan tanpa mengetahui probabilitas transisi antar state Finite Markov Decision Problem (FMDP), sehingga dinamika lingkungan robot tidak harus diketahui [1].

2.6 Fuzzy Q Learning (FQL)

Ketika Q Learning bekerja dengan state dan aksi kontinu seperti pada robot, tabel fungsi Q membesar sebagai tempat penyimpanan pasangan state-aksi yang baru. Pada kasus ini, proses pembelajaran akan memerlukan waktu yang sangat lama dan keperluan memori yang besar, sehingga Q Learning sulit diterapkan. Solusi dari permasalahan ini adalah dengan menambahkan logika fuzzy untuk memodifikasi Q-learning agar dapat bekerja pada state dan aksi yang kontinu.

Pada Fuzzy Inference System (FIS) dibutuhkan representasi aksi dan Q fungsi, dengan cara modifikasi rule base. Bentuk rule-rule yang digunakan pada penelitian ini sudah dimodifikasi sesuai dengan persamaan (1) dan (2).

$$state = S_i, aksi = a_i, ai A_i \tag{1}$$

$$state = S_i, aksi = a_i, Q = q (S_i, a_i) \tag{2}$$

Dengan menggunakan FIS Takagi-Sugeno dengan N rule, aksi a(x) untuk vector masukan x(x1,x2, ,xn) dan konsekuen rule (ai)(i = 1)N dapat dihitung dengan persamaan (3).

$$a(x) = \frac{\sum_{i=1}^N \alpha_i(x) \cdot a_i}{\sum_{i=1}^N \alpha_i} \tag{3}$$

Dan nilai Q yang bersesuaian dapat dihitung dengan persamaan (4).

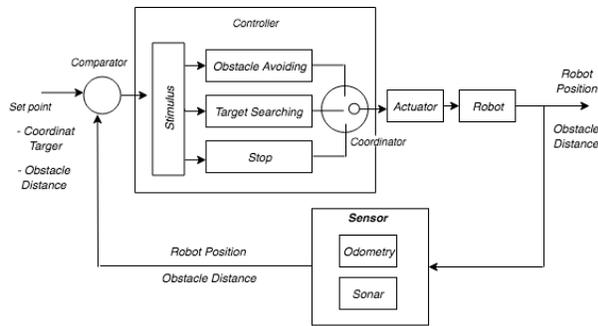
$$Q(x, a) = \frac{\sum_{i=1}^N \alpha_i(x) \cdot q(s_i, a_i)}{\sum_{i=1}^N \alpha_i(x)} \tag{4}$$

3. Perancangan Sistem

Berikut adalah perancangan dari sistem kendali berbasis perilaku pada autonomous mobile robot pioneer 3AT.

3.1 Gambaran Umum Sistem

Diagram blok sistem kendali dari sistem yang dirancang pada penelitian ini dapat dilihat pada Gambar 3.



**Gambar 3. Diagram blok sistem kendali**

Nilai *set point* yang digunakan pada sistem ini yaitu koordinat target (x, y) dan jarak robot terhadap halangan. *Controller* terdiri dari tiga perilaku dan koordinator antar perilaku. Tiga perilaku yang dirancang terdiri dari FQL *obstacle avoiding*, *target searching*, dan *stop*. Metoda koordinasi antar perilaku yang digunakan yaitu metoda kompetitif sehingga hanya akan ada satu perilaku yang akan dijalankan oleh *actuator*. Masukan ke *controller* ini berasal dari pembacaan sensor odometri dan sonar. Luaran dari *controller* akan digunakan sebagai penggerak pada *actuator*. Robot akan berjalan dilingkungan selama belum menemukan target dan tetap akan menghindari halangan pada saat proses pencarian target. Posisi robot dan jarak robot terhadap halangan akan menjadi *feedback* yang dikirim ke *comparator* untuk dibandingkan dengan nilai *set point*.

**3.2 Desain Perilaku**

Pada arsitektur kendali berbasis perilaku sistem terdiri dari beberapa modul dengan tiap modul bertanggung jawab untuk melakukan satu perilaku. Setiap perilaku memiliki tugas dan melakukan proses lengkap mulai dari sensing sampai aksi [11]. Pada penelitian ini dirancang 3 perilaku, yaitu *Obstacle Avoiding*, *Target Searching*, dan *Stop*.

**3.2.1 Obstacle Avoiding**

Perilaku pertama yang dirancang yaitu *obstacle avoiding*. Perilaku ini dirancang dengan FQL dan bertugas untuk menghindari setiap halangan yang robot temui dan deteksi ketika sedang bergerak. Robot dapat mendeteksi halangan dengan bantuan *sensor range finder*. Pada penelitian ini jenis *sensor range finder* yang digunakan yaitu sonar.

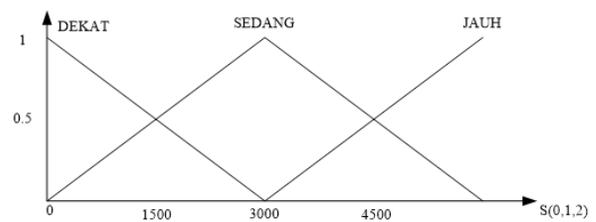
Nilai nol dapat diartikan robot telah menabrak halangan dan nilai 1000 artinya jarak robot dari halangan adalah 1 m. Pada penelitian ini, jarak robot dengan halangan 1 m sudah dianggap terjadi tabrakan dan robot akan mendapat *punishment* sebesar -1. Sebaliknya robot akan mendapatkan *reward* sebesar 1 ketika robot jauh dari halangan atau dengan nilai sensor  $i$  2000 mm. Jika

berada diantara jarak itu maka nilai yang diperoleh robot adalah 0. Pembagian sudut sonar pada robot dapat dilihat pada Tabel 1.

**Tabel 1. Pengelompokan sensor sonar**

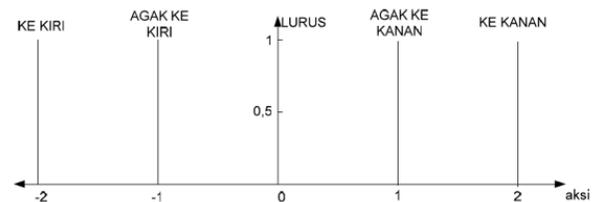
Masukan	Sudut Cakupan Sonar	Keterangan
S0	30° sampai 90°	Sonar kiri
S1	-30° sampai 30°	Sonar depan
S2	-90° sampai -30°	Sonar kanan

*Obstacle avoiding FQL* memiliki tiga fungsi keanggotaan segitiga seperti ditunjukkan oleh Gambar 4.



**Gambar 4. Fungsi keanggotaan masukan *obstacle avoiding FQL***

Jarak antara robot dan halangan menjadi input *fuzzy* yang digambarkan sesuai pada fungsi keanggotaannya. Nilai keluaran FQL berbeda dengan sistem logika *fuzzy* biasa, luarannya adalah nilai estimasi dari fungsi Q yang besarnya tidak pasti. Setiap nilai mewakili aksi yang akan diberikan ke robot (Gambar 5).



**Gambar 5. Fungsi keanggotaan luaran *obstacle avoiding FQL***

Perbedaan sistem FQL dengan sistem logika *fuzzy* biasa, yaitu terletak pada luaran FQL yang berbentuk nilai estimasi dari fungsi q yang besarnya tidak pasti. Akan tetapi masing-masing nilai estimasi akan dipastikan bersesuaian dengan aksi kendali robot. Fungsi keanggotaan luaran FQL yang digunakan adalah *singleton* sebagaimana ditunjukkan pada Gambar 5. Hal ini dikarenakan algoritma *Q learning* ini mengharuskan *state-state* lingkungan dan agen melakukan aksi dengan mengambil nilai dalam sebuah set diskrit yang relatif kecil [6].

### 3.2.2 Target Searching

Setelah robot dapat bergerak tanpa menabrak halangan, misi selanjutnya adalah *target searching*. Robot harus dapat menemukan target yang berada pada posisi tertentu. Pada penelitian sebelumnya yang sudah dilakukan Casi dkk, pada behavior target searching menggunakan metoda *fuzzy Q learning* [12]. *Behavior* ini bertugas untuk mencari sebuah lampu sebagai target. Robot akan mengikuti sumber cahaya dengan bantuan *light sensor*. Dengan 2 masukan dari nilai light sensor (sisi kiri dan kanan robot), 3 fungsi keanggotan pada *input fuzzy* (jauh, sedang, dekat) dan 5 fungsi keanggotan pada *output fuzzy* (kiri, agak ke kiri, lurus, agak ke kanan, kanan) maka target searching memiliki 9 rule dengan 5 aksi disetiap rule. Untuk memilih *rule* terbaik maka digunakan metode pemilihan aksi yaitu *e-greedy selection*. Fungsi *target searching* FQL yaitu bergerak mencari dan menuju target. Jika robot bergerak mendekati sumber cahaya maka robot akan mendapatkan *reward* dan sebaliknya jika robot berada jauh dari sumber cahaya maka robot akan mendapatkan *punishment*.

Pada penelitian ini, untuk menyelesaikan tugas *target searching robot* tidak menggunakan metoda FQL. Untuk misi ini robot menggunakan perhitungan posisi dengan bantuan sensor posisi. Sensor yang umum digunakan untuk mendapatkan data posisi dari suatu robot adalah *rotary encoder*, untuk selanjutnya data yang dihasilkan sensor akan dimasukkan ke dalam perhitungan odometri oleh sensor odometri untuk mendapatkan posisi relatif dari robot terhadap posisi awal robot. Posisi relatif didasarkan pada perhitungan rotasi roda untuk mendapatkan perkiraan posisi robot dan bukan posisi absolut robot [7].

Odometri adalah penggunaan data dari sensor pergerakan untuk memperkirakan perubahan posisi dari waktu ke waktu. Odometri ini akan memetakan posisi robot dalam bidang koordinat kartesius sehingga akan didapatkan data posisi berupa titik koordinat (*path*) dan arah hadap (*heading*) dari robot tersebut [13].

### 3.2.3 Stop

Perilaku selanjutnya yang dirancang yaitu *stop*. Perilaku ini dirancang untuk membuat robot berhenti ketika telah sampai pada koordinat target yang dituju. Diperlukan nilai batas toleransi pada titik koordinat target. Hal ini dimaksudkan untuk mempermudah robot menemukan posisi koordinat target, karena jika robot sampai pada titik koordinat lebih dari posisi target dengan batas maksimal 300 dan kurang dari koordinat target dengan batas maksimal 300, robot dianggap sudah menemukan target.

Untuk posisi target yang memiliki koordinat (7000,2000), dengan adanya batas toleransi target, apabila robot sudah mencapai koordinat  $x = 7000 \pm 300$

dan koordinat  $y = 2000 \pm 300$  sudah dianggap berhasil menemukan target dan kecepatan robot akan diubah menjadi nol.

## 4. Simulasi dan Hasil Pengujian

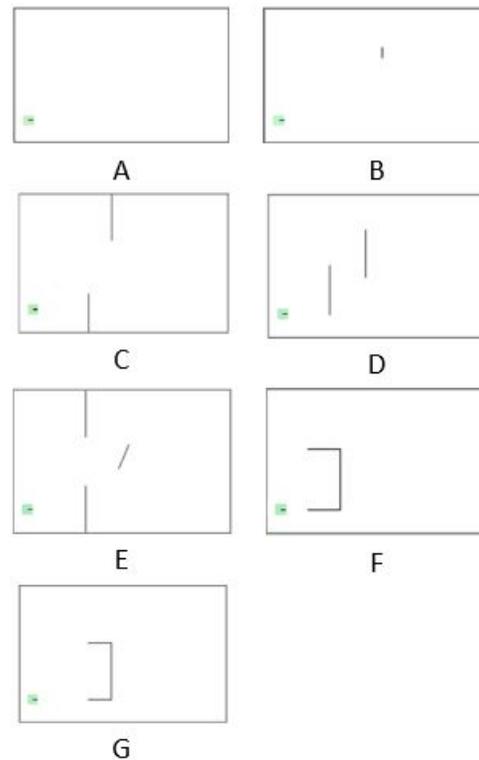
### 4.1 Desain Ruang Simulasi

Simulasi program dilakukan menggunakan software simulator MobileSim. Pada penelitian ini digunakan lingkungan dengan ukuran  $9000 \times 6000$  mm.

Untuk pembuatan lingkungan uji simulasi digunakan software Mapper3. Dengan software ini kita dapat membuat lingkungan dan *obstacle* yang dibutuhkan sesuai skenario uji lapangan yang akan dilakukan.

### 4.2 Hasil dan Analisis

Pengujian algoritma dilakukan dengan membuat simulasi menggunakan simulator MobileSim dan menggunakan ruang simulasi yang sudah dirancang sebelumnya sebagai lingkungan tempat simulasi dilakukan, dengan Luas Lingkungan ( $x = 9000$ ,  $x = 6000$ ), Titik Awal Robot ( $x = 600$ ,  $y = 1000$ ), koordinat target ( $x = 6000$ ,  $y = 3500$ ), dan nilai toleransi sebesar 300.

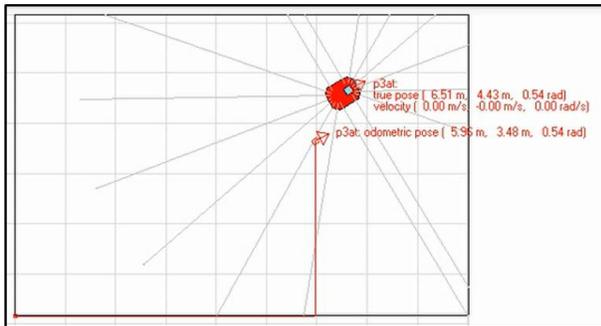


**Gambar 6. Map lingkungan uji simulasi**

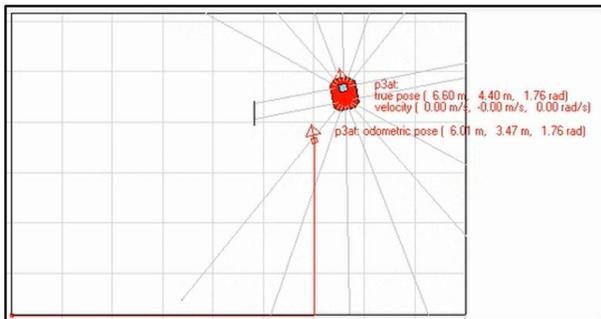
Pada simulasi map A tidak diberikan halangan pada lingkungan. Pada simulasi map B, C, D, E, F, dan G robot sudah mampu beradaptasi terhadap lingkungan yang baru, dibuktikan dengan robot mampu menghindari

halangan yang berbeda-beda. Selanjutnya robot dapat menemukan posisi target dan berhenti di dalam area toleransi. Gambar 6 menunjukkan map dari setiap simulasi percobaan dengan berbagai skenario. Dengan kotak hijau menunjukkan titik awal robot dan garis hitam di tengah area menunjukkan halangan.

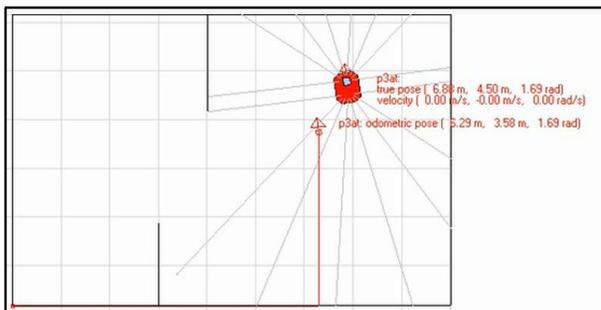
Hasil uji simulasi pada 7 Map berbeda dengan robot digambarkan oleh kotak warna merah di dalam area, seperti ditunjukkan Gambar 6, adalah sebagai berikut:



Gambar 7. Hasil uji simulasi map A



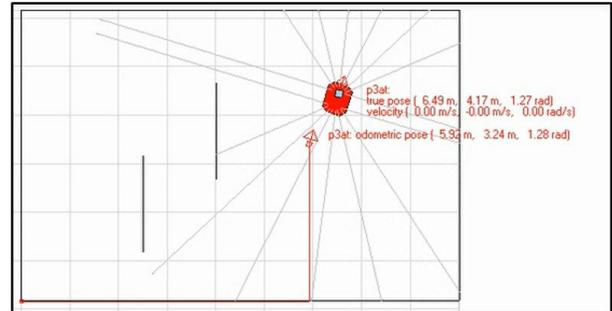
Gambar 8. Hasil uji simulasi map B



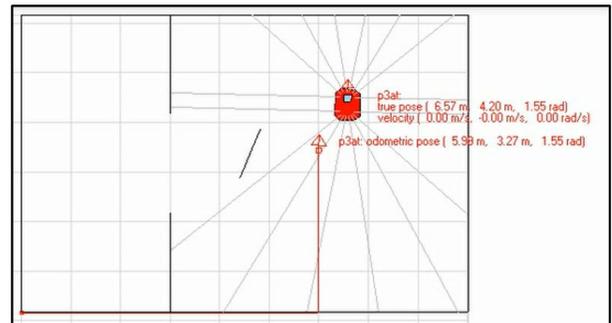
Gambar 9. Hasil uji simulasi map C

Hasil uji simulasi 7 map dengan 5x uji untuk masing-masing map, dapat dilihat pada Gambar 7, 8, 9, 10, 11, 12, dan 13. Untuk detail pencapaian robot pada setiap map adalah sebagai berikut:

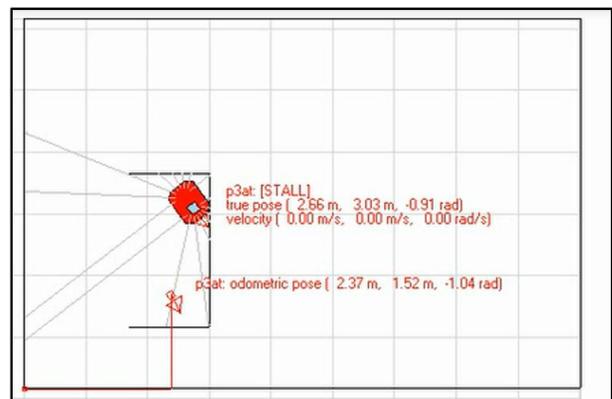
Tabel 2 menunjukkan hasil pengujian dari 7 map, dengan ok artinya berhasil dan nok artinya tidak berhasil.



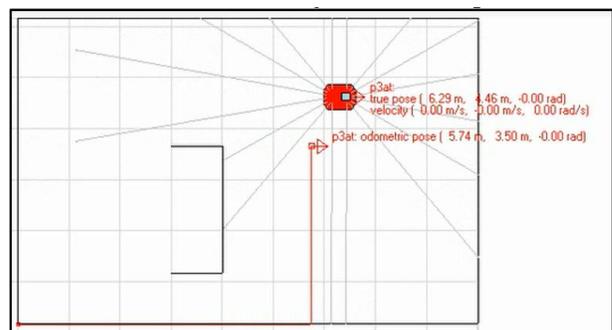
Gambar 10. Hasil uji simulasi map D



Gambar 11. Hasil uji simulasi map E



Gambar 12. Hasil uji simulasi map F



Gambar 13. Hasil uji simulasi map G

**Tabel 2. Ini Contoh Judul Tabel**

Map	Rata-rata X	Rata-rata Y	Target Searching	Obstacle Avoiding	Stop
A	5929	3461	ok	ok	ok
B	6014	3432	ok	ok	ok
C	6298	3536	ok	ok	ok
D	5903	3200	ok	ok	ok
E	5988	3230	ok	ok	ok
F	-	-	nok	nok	nok
G	5701	3495	ok	ok	ok

Berdasarkan hasil pengujian, besar akurasi pada sistem dapat dihitung dengan persamaan (5).

$$Akurasi = \frac{Total\ Sukses}{Total\ Pengujian} \cdot 100\% \quad (5)$$

$$= \frac{30}{35} \cdot 100\% = 85,71\% \quad (6)$$

Pada simulasi map A tidak diberikan halangan pada lingkungan sehingga tidak ada penghalang antara jalur robot dan target. Robot dapat dengan mudah menemukan posisi target dan berhenti di dalam area toleransi seperti ditunjukkan pada Gambar 7.

Pada simulasi map B, C, D, E, dan G robot sudah mampu beradaptasi terhadap lingkungan yang baru, dibuktikan dengan robot mampu menghindari halangan yang diberikan. Selanjutnya robot dapat menemukan posisi target dan berhenti di dalam area toleransi seperti ditunjukkan pada Gambar 8, 9,10, 11, dan 13.

Pada simulasi F robot menabrak halangan dan tidak dapat melanjutkan pergerakan untuk menemukan target. Ini dikarenakan jarak halangan dengan robot terlalu dekat dan robot tidak memiliki ruang gerak yang cukup. Hal ini menyebabkan robot terjebak pada bentuk halangan sudut dan halangan yang mengelilinginya sehingga robot tidak dapat melanjutkan perjalanan mencari target seperti ditunjukkan pada Gambar 12.

Pada penelitian sebelumnya yang sudah dilakukan oleh Casi dkk, yaitu melakukan simulasi penerapan *behavior based architecture* dengan algoritma *Reinforcement Learning* pada autonomous mobile robot mendapatkan akurasi sebesar 100% dalam menemukan target [12]. Sedangkan pada penelitian ini akurasi robot dalam menemukan target yaitu sebesar 85,71%. Perbedaan penelitian sebelumnya dengan penelitian ini terletak pada 3 hal, yaitu:

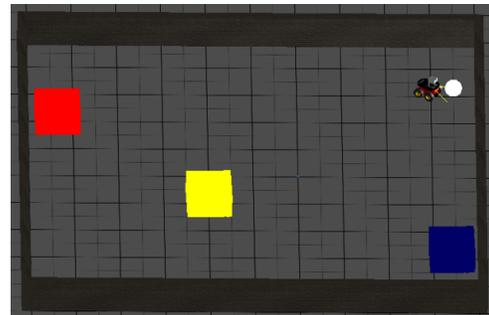
1. Sensor dan target

Penelitian sebelumnya menggunakan bantuan sensor cahaya untuk menemukan target yang berbentuk lampu, sedangkan penelitian ini menggunakan bantuan sensor odometri yang akan

menghitung posisi relatif robot dalam mencari target dengan bentuk apapun.

2. Map

Pada penelitian sebelumnya simulasi pengujian menggunakan halangan yang berbentuk solid seperti kubus dan bola seperti pada Gambar 14 [12]. Sedangkan pada penelitian ini menggunakan halangan berbentuk ruang, sehingga robot dapat terjebak di dalam halangan tersebut seperti ditampilkan pada Gambar 12.



**Gambar 14. Lingkungan uji penelitian Casi dkk**

3. *Software* simulasi

*Software* Webots adalah sebuah *software* simulasi yang dipakai pada penelitian sebelumnya dan penelitian ini menggunakan *software simulator* MobileSim.

**5. Kesimpulan dan Saran**

**5.1 Kesimpulan**

Penerapan arsitektur kendali berbasis perilaku pada autonomous mobile robot berhasil dilakukan dengan semua perilaku bekerja sesuai fungsinya. Dengan diterapkannya Reinforcement Learning, robot mampu belajar menghindari halangan yang diberikan. Robot juga mampu beradaptasi terhadap lingkungan yang tidak terstruktur yaitu mampu menghindari halangan yang ditempatkan pada posisi, lingkungan, dan tingkat kesulitan yang berbeda serta mampu menyelesaikan misi menemukan target yang sudah ditentukan, dengan akurasi sebesar 85,71%.

**5.2 Saran**

Pada penelitian selanjutnya agar robot dapat melanjutkan perjalanan ketika robot menabrak, dapat ditambahkan perilaku mundur dilanjutkan dengan berbelok yang harus dijalankan ketika robot menabrak dan terjebak di dalam halangan.

**Daftar Pustaka**

[1] M. Carreras Pérez *et al.*, *A proposal of a behavior-based control architecture with reinforcement learning for an autonomous underwater robot.* Universitat de Girona, 2003.

- [2] C. Szepesvári, "Algorithms for reinforcement learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 4, no. 1, pp. 2–5, 2010.
- [3] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [4] P. Dayan and C. Watkins, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.
- [5] L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška, "Continuous-state reinforcement learning with fuzzy approximation," in *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*. Springer, 2005, pp. 27–43.
- [6] P. Y. Glorennec and L. Jouffe, "Fuzzy q-learning," in *Proceedings of 6th international fuzzy systems conference*, vol. 2. IEEE, 1997, pp. 659–662.
- [7] D. Anderson, "Imu odometry," 2006. [Online]. Available: [http://www.geology.smu.edu/dpa-www/robo/Encoderimu\\_odo](http://www.geology.smu.edu/dpa-www/robo/Encoderimu_odo)
- [8] R. Brooks, "A robust layered control system for a mobile robot," *IEEE journal on robotics and automation*, vol. 2, no. 1, pp. 14–23, 1986.
- [9] P. Pirjanian, "Behavior coordination mechanisms-state-of-the-art," Citeseer, Tech. Rep., 1999.
- [10] M. J. Matarić, J. Maja, R. C. Arkin *et al.*, *The robotics primer*. Mit Press, 2007.
- [11] I.-S. Kweon, M. Watanabe, K. Onoguchi, and Y. Kuno, "Behavior-based mobile robot using active sensor fusion," in *IEEE International Conference on Robotics and Automation, 1992*. IEEE, 1992.
- [12] C. Setianingsih and K. Mutijarsa, "Penerapan behavior based architecture dan reinforcement learning pada autonomous mobile robot," *The 3rd Indonesian Symposium on Robot Soccer Competition*, Jun 2015.
- [13] E. Olson, "A primer on odometry and motor control," *Electronic Group Discuss*, 2004.