

Perancangan Intelligent Agent untuk Optimasi Kekeruhan Air Kolam Ikan Menggunakan Metode Pencarian

Zidan Faiq Nasrullah^{*1}, Nonot Wisnu Karyanto^{*2}, Anang Kukuh Adisusilo^{*3}

Email: zfn026@mhs.uwks.ac.id, nonot@uwks.ac.id, anang65@uwks.ac.id

Abstrak

Sistem budidaya ikan memerlukan pengelolaan kualitas air yang optimal untuk menjaga kesehatan dan pertumbuhan ikan. Salah satu permasalahan utama adalah tingkat kekeruhan air yang dapat berubah akibat faktor lingkungan seperti hujan, sisa pakan, dan aktivitas ikan. Penelitian ini bertujuan untuk merancang sebuah *intelligent agent* yang mampu mengoptimalkan kondisi kekeruhan air kolam ikan secara otomatis. Metode yang digunakan meliputi pendekatan *problem solving agent* dengan pemodelan *state space* serta penerapan algoritma pencarian seperti *Breadth First Search (BFS)*, *Depth First Search (DFS)*, dan *heuristic generate and test*. Sistem bekerja dengan membaca data dari sensor kekeruhan, mengklasifikasikan kondisi air ke dalam beberapa *state*, serta menentukan aksi yang sesuai seperti pengaktifan filter atau pemberian notifikasi. Hasil perancangan menunjukkan bahwa sistem mampu menentukan jalur optimal dalam mencapai kondisi air bersih dengan efisien. Selain itu, mekanisme evaluasi berulang memungkinkan sistem beradaptasi terhadap perubahan kondisi lingkungan secara dinamis. Dengan demikian, *intelligent agent* yang dirancang dapat meningkatkan efektivitas pengelolaan kualitas air kolam ikan secara otomatis.

Kata kunci: BFS, DFS, intelligent agent, kekeruhan air, sistem cerdas.

Abstract

This study addresses the need for optimal water quality management in fish farming systems, particularly in controlling water turbidity caused by environmental factors such as rainfall, leftover feed, and fish activity. The objective of this research is to design an intelligent agent capable of automatically optimizing the turbidity level of pond water. The proposed method applies a problem-solving agent approach using state space modeling and search algorithms, including Breadth First Search (BFS), Depth First Search (DFS), and heuristic generate and test. The system operates by reading turbidity sensor data, classifying water conditions into predefined states, and selecting appropriate actions such as activating filtration systems or triggering notifications. The results demonstrate that the system can determine an optimal path to achieve clean water conditions efficiently. Additionally, the iterative evaluation mechanism allows the system to adapt dynamically to environmental changes. Therefore, the designed intelligent agent improves the effectiveness of automated water quality management in fish farming.

Keywords: BFS, DFS, intelligent agent, turbidity, intelligent system.

I. PENDAHULUAN

Sistem budidaya ikan merupakan salah satu sektor penting dalam bidang perikanan yang memerlukan pengelolaan lingkungan secara optimal untuk mendukung pertumbuhan dan kesehatan ikan. Salah satu faktor utama yang memengaruhi kualitas lingkungan tersebut adalah kondisi air kolam, khususnya tingkat kekeruhan. Kekeruhan air yang tidak terkontrol dapat menghambat penetrasi cahaya, mengganggu sistem pernapasan ikan, serta menurunkan kualitas ekosistem kolam secara keseluruhan. Faktor-faktor seperti curah hujan, sisa pakan, dan aktivitas ikan menjadi penyebab utama perubahan tingkat kekeruhan air secara dinamis.[1]

Dalam praktik konvensional, pemantauan kondisi air masih dilakukan secara manual sehingga kurang efisien dan berpotensi menimbulkan keterlambatan dalam pengambilan keputusan. Hal ini mendorong perlunya suatu sistem otomatis yang mampu mendeteksi dan merespons perubahan kondisi air secara cepat dan tepat. Salah satu pendekatan yang dapat digunakan adalah penerapan konsep *intelligent agent* dalam sistem cerdas, yang memiliki kemampuan untuk menerima input dari lingkungan, memproses informasi, serta mengambil keputusan berdasarkan kondisi yang terdeteksi.[2]

Tujuan dari penelitian ini adalah merancang sistem *intelligent agent* yang mampu mengoptimalkan kondisi air kolam ikan secara otomatis melalui pemilihan aksi yang tepat berdasarkan tingkat kekeruhan. Selain itu, penelitian ini juga bertujuan untuk menganalisis performa metode pencarian dalam menentukan jalur optimal menuju kondisi air yang bersih. Dengan adanya sistem ini, diharapkan pengelolaan kualitas air kolam ikan dapat dilakukan secara lebih efektif, efisien, dan adaptif terhadap perubahan kondisi lingkungan.[3]

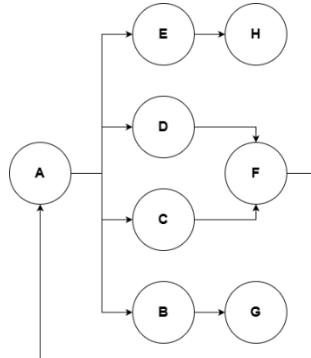
II. METODE PENELITIAN

Berikut adalah metode yang digunakan dalam penelitian ini yaitu :

- 2.1 Desain Sistem
 - A. Model PAGE
 - a. Percept : Data tingkat kekeruhan air yang diperoleh dari sensor, diklasifikasikan menjadi empat kategori: rendah, sedang, tinggi, dan sangat tinggi.
 - b. Action : Tidak melakukan tindakan (kondisi rendah), Mengaktifkan filter selama 20 menit (sedang), Mengaktifkan filter selama 45 menit (tinggi), Mengaktifkan buzzer sebagai notifikasi (sangat tinggi)
 - c. Goal : Menjaga kondisi air tetap jernih dan mencapai state optimal (air bersih).
 - d. Enviroment : Lingkungan sistem terdiri dari kolam ikan, air, sensor kekeruhan, dan perangkat filtrasi.
 - 2.2 Alur Kerja Sistem
 - A. Cara Kerja Agent
 - a. Sensor membaca kondisi : Sensor kekeruhan membaca kondisi air (rendah / sedang / tinggi / sangat tinggi)
 - b. Sistem menganalisis dan Agent menentukan : ini masih bisa ditangani filter?, atau sudah kondisi darurat?
 - c. Agent mengambil keputusan berdasarkan level kekeruhan : pilih durasi filter, atau aktifkan notifikasi
 - d. Sistem menjalankan aksi : Filter nyala sesuai waktu / buzzer aktif
 - e. Setelah aksi selesai : sensor membaca lagi kondisi air
 - f. Loop (berulang) : Kalau masih keruh ulangi proses, kalau sudah bersih berhenti
 - B. Pseudocode


```
cek_kekeruhan()
if keruh_rendah:
tidak_ada_aksi()
elif keruh_sedang:
nyalakan_filter(20 menit)
elif keruh_tinggi:
nyalakan_filter(45 menit)
elif keruh_sangat_tinggi:
kirim_notifikasi()
setelah_filter_selesai:
cek_kondisi_air()
if masih_keruh:
ulangi_proses
else:
selesai
```
 - 2.3 Formulasi Problem
 - A. Initial States : Sistem melakukan proses pembacaan kondisi kekeruhan air menggunakan sensor.
 - B. States :
 - a. A : Pengecekan kondisi air
 - b. B : Kekeruhan rendah
 - c. C : Kekeruhan sedang
 - d. D : Kekeruhan tinggi
 - e. E : Kekeruhan sangat tinggi
 - f. F : Proses filtrasi
 - g. G : Air bersih (*goal state*)
 - h. H : Notifikasi peringatan (buzzer)
 - C. Action :
 - a. Tidak melakukan tindakan (pada kondisi kekeruhan rendah)
 - b. Mengaktifkan filter selama 20 menit (kekeruhan sedang)
 - c. Mengaktifkan filter selama 45 menit (kekeruhan tinggi)
 - d. Mengirim notifikasi buzzer (kekeruhan sangat tinggi)
 - D. Successor Function :

- a. $A \rightarrow \{B, C, D, E\}$
- b. $B \rightarrow \{G\}$
- c. $C \rightarrow \{F\}$
- d. $D \rightarrow \{F\}$
- e. $E \rightarrow \{H\}$
- f. $F \rightarrow \{G\}$ atau kembali ke A (proses evaluasi ulang)
- g. $H \rightarrow \{\text{stop}\}$
- E. Goal Test : Mencapai State G yaitu kondisi kolam optimal (air bersih)
- F. Path Cost : Setiap perpindahan state memiliki biaya = 1
- 2.4 State Space Graph

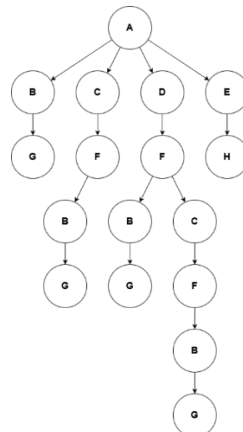


Gambar 1. (gambar menunjukkan State Space Graph dari sistem optimasi kekeruhan air)

Penjelasan :

- A : Proses pengecekan kondisi kekeruhan air menggunakan sensor (kondisi awal sistem)
 B : Air dalam kondisi keruh rendah (masih baik, tidak perlu tindakan)
 C : Air dalam kondisi keruh sedang (memerlukan proses filtrasi ringan)
 D : Air dalam kondisi keruh tinggi (memerlukan proses filtrasi lebih lama)
 E : Air dalam kondisi sangat keruh (kondisi darurat)
 F : Proses filtrasi berlangsung untuk mengurangi tingkat kekeruhan air
 G : Air sudah bersih dan berada dalam kondisi optimal (goal tercapai)
 H : Buzzer aktif sebagai peringatan bahwa kondisi air sangat keruh dan memerlukan penanganan manual(pengurasan kolam).

2.5 Tree Terminology



Gambar 2. (gambar menunjukkan State Space Graph dari sistem optimasi kekeruhan air)

Penjelasan :

- A. Root Node : A, merupakan kondisi awal sistem yaitu proses pengecekan kekeruhan air
 B. Child Node : B, C, D, dan E merupakan child dari A yang menunjukkan hasil klasifikasi kondisi air. Selain itu :
 a. F merupakan child dari C dan D (proses filtrasi)
 b. G merupakan child dari B dan juga dari beberapa node B lainnya (kondisi air bersih)
 c. H merupakan child dari E (buzzer aktif)
 C. Parent Node :

- a. A adalah parent dari B, C, D, dan E
- b. B adalah parent dari G
- c. C dan D adalah parent dari F
- d. F adalah parent dari B dan C (hasil setelah proses filtrasi)
- e. E adalah parent dari H
- D. Leaf Node : G merupakan leaf node karena tidak memiliki turunan dan merupakan kondisi tujuan (goal), H juga merupakan leaf node karena tidak memiliki turunan dan menunjukkan kondisi peringatan (buzzer aktif)
- E. Path (Jalur) :
 $A \rightarrow B \rightarrow G$
 $A \rightarrow C \rightarrow F \rightarrow B \rightarrow G$
 $A \rightarrow D \rightarrow F \rightarrow B \rightarrow G$
 $A \rightarrow D \rightarrow F \rightarrow C \rightarrow F \rightarrow B \rightarrow G$

2.6 Proses BFS

A. Langkah-langkah BFS

- a) Langkah 1
Fringe: [A]
- b) Langkah 2
Expand $A \rightarrow B, C, D, E$
Fringe: [B, C, D, E]
- c) Langkah 3
Expand $B \rightarrow G$
Fringe: [C, D, E, G]
- d) Langkah 4
Expand $C \rightarrow F$
Fringe: [D, E, G, F]
- e) Langkah 5
Expand $D \rightarrow F$ (sudah ada di fringe, jadi tidak duplikat)
Fringe: [E, G, F]
- f) Langkah 6
Expand $E \rightarrow H$
Fringe: [G, F, H]
- g) Langkah 7
Expand $G \rightarrow GOAL$ (air bersih tercapai)
Fringe: [F, H]

B. Tabel BFS

Langkah	Fringe	Keterangan
Mulai	[[A]]	Inisialisasi (cek kondisi kekeruhan air)
A diproses	[[B, C, D, E]]	A bercabang: B (rendah), C (sedang), D (tinggi), E (sangat tinggi)
B diproses	[[C, D, E, G]]	$B \rightarrow G$ (air bersih, goal tercapai)
C diproses	[[D, E, G, F]]	$C \rightarrow F$ (proses filter ringan)
D diproses	[[E, G, F]]	$D \rightarrow F$ (filter lebih lama)
E diproses	[[G, F, H]]	$E \rightarrow H$ (buzzer aktif / kondisi darurat)
F diproses	[[G, A]]	$F \rightarrow A$ (cek ulang kondisi setelah filter)
A (ulang cek)	[[B, C, D, E]]	Sistem kembali mengecek kondisi air
G ditemukan	-	Goal tercapai (air bersih/optimal)

2.7 Proses DFS

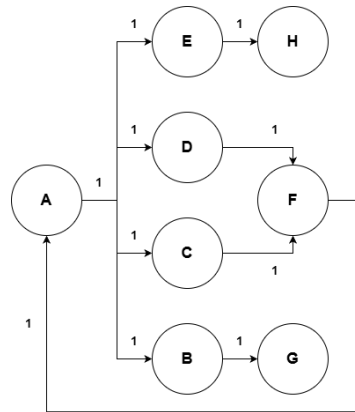
A. Langkah-langkah DFS

- a) Langkah 1
Fringe: [A]
- b) Langkah 2

- Expand A $\rightarrow$ B, C, D, E (DFS ambil jalur pertama dulu)
Fringe: [B, C, D, E]
- c) Langkah 3
Expand B $\rightarrow$ G
Fringe: [G, C, D, E]
- d) Langkah 4
G ditemukan (GOAL)
Fringe: [C, D, E]
- e) Langkah 5(backtrack)
Kembali ke cabang lain dari A
Expand C $\rightarrow$ F
Fringe: [F, D, E]
- f) Langkah 6
Expand F $\rightarrow$ A (cek ulang / loop sistem)
Fringe: [A, D, E]
- g) Langkah 7
A diproses ulang $\rightarrow$ B, C, D, E
Fringe: [B, C, D, E]
- h) Langkah 8
Expand D $\rightarrow$ F
Fringe: [F, E]
- i) Langkah 9
Expand F $\rightarrow$ A (loop kembali)
Fringe: [A, E]
- j) Langkah 10
Expand E $\rightarrow$ H
Fringe: [H]
- k) Langkah 11
H ditemukan (kondisi darurat / akhir cabang)
Fringe: -
- B. Tabel DFS

Langkah	Fringe	Keterangan
Mulai	[[A]]	Inisialisasi (cek kondisi kekeruhan air)
A diproses	[[B, C, D, E]]	A diekspansi (urut prioritas: B dulu)
B diproses	[[C, D, E, G]]	B $\rightarrow$ G (jalur ke kondisi bersih)
G diproses	[[C, D, E]]	Goal ditemukan di cabang B
(Backtrack)	[[C, D, E]]	Kembali ke node lain karena DFS tidak semua cabang sekaligus
C diproses	[[D, E, F]]	C $\rightarrow$ F (proses filtrasi)
F diproses	[[D, E, A]]	F $\rightarrow$ A (cek ulang kondisi setelah filter)
A (ulang)	[[B, C, D, E]]	Sistem kembali ke root untuk eksplorasi
D diproses	[[E, F]]	D $\rightarrow$ F (filter lebih lama)
F diproses	[[E, A]]	F $\rightarrow$ A (loop pengecekan ulang)
E diproses	[[H]]	E $\rightarrow$ H (buzzer aktif / kondisi darurat)
H diproses	-	Node akhir (tidak menuju goal)

- 2.8 Heuristic Search
A. Generate and Test



Gambar 3. (gambar menunjukan State Space Generate and Test dengan cost 1)

Penjelasan Tabel :

State	Makna	h(n)
A	cek kondisi sensor	1
B	air bersih / rendah keruh	1
C	sedang	1
D	tinggi	1
E	sangat tinggi (buruk)	1
F	hasil filter (stabil)	1
G	optimal (goal)	1
H	darurat (buzzer)	1

B. Tabel Kemungkinan

No	Jalur	Valid	Total Cost
1	A → B → G	Ya	2
2	A → C → F → B → G	Ya	4
3	A → D → F → B → G	Ya	4
4	A → D → F → C → F → B → G	Ya	6
5	A → E → H	Tidak	2 (buntu / darurat, tidak ke goal)
6	A → C → F → A → B → G	Ya	5
7	A → E → F → B → G	Tidak	4 (tidak valid, E tidak ke F secara normal sistem)

C. Perhitungan Manual

- $A \rightarrow B \rightarrow G$
 $= 1 (A-B) + 1 (B-G)$
 Valid – Goal tercapai (G)
 Total Cost = 2
- $A \rightarrow C \rightarrow F \rightarrow B \rightarrow G$
 $= 1 (A-C) + 1 (C-F) + 1 (F-B) + 1 (B-G)$
 Valid – Goal tercapai (G)
 Total Cost = 4
- $A \rightarrow D \rightarrow F \rightarrow B \rightarrow G$
 $= 1 (A-D) + 1 (D-F) + 1 (F-B) + 1 (B-G)$
 Valid – Goal tercapai (G)
 Total Cost = 4
- $A \rightarrow D \rightarrow F \rightarrow C \rightarrow F \rightarrow B \rightarrow G$
 $= 1 (A-D) + 1 (D-F) + 1 (F-C) + 1 (C-F) + 1 (F-B) + 1 (B-G)$
 Valid – Goal tercapai (G)
 Total Cost = 6
- $A \rightarrow E \rightarrow H$
 $= 1 (A-E) + 1 (E-H)$

Tidak valid – tidak sampai ke goal (G), kondisi darurat (buzzer aktif)

Total Cost = 2

f) $A \rightarrow C \rightarrow F \rightarrow A \rightarrow B \rightarrow G$
 $= 1 (A-C) + 1 (C-F) + 1 (F-A) + 1 (A-B) + 1 (B-G)$

Valid – Goal tercapai (G) (loop terjadi tetapi tetap mencapai goal)

Total Cost = 5

Kesimpulan:

Jalur paling optimal: $A \rightarrow B \rightarrow G$ (cost = 2)

Jalur lain lebih panjang karena proses filtrasi (F)

$E \rightarrow H$ adalah kondisi darurat dan tidak menuju goal

III. HASIL DAN PEMBAHASAN

3.1 Representasi State Space

Sistem pengendalian kekeruhan air kolam ikan direpresentasikan dalam bentuk state space graph yang menggambarkan hubungan antar kondisi berdasarkan aksi yang dilakukan oleh agent. Setiap state menunjukkan kondisi kekeruhan air, sedangkan transisi antar state merepresentasikan aksi seperti proses filtrasi atau pemberian notifikasi.

State awal (A) merupakan proses pengecekan kondisi air menggunakan sensor. Berdasarkan hasil pembacaan, sistem dapat berpindah ke state B (kekeruhan rendah), C (sedang), D (tinggi), atau E (sangat tinggi). Untuk kondisi sedang dan tinggi, sistem akan menuju state F sebagai proses filtrasi, sedangkan kondisi sangat tinggi akan menuju state H sebagai kondisi darurat. Tujuan akhir sistem adalah mencapai state G, yaitu kondisi air bersih. Representasi ini memungkinkan sistem untuk dianalisis menggunakan berbagai metode pencarian guna menentukan jalur terbaik menuju kondisi optimal.

3.2 Analisis Breadth First Search (BFS)

Metode Breadth First Search (BFS) digunakan untuk menelusuri state space secara melebar berdasarkan tingkat kedalaman. Proses pencarian dimulai dari state awal A dan mengeksplorasi seluruh node pada level yang sama sebelum berpindah ke level berikutnya.

Hasil analisis menunjukkan bahwa BFS menemukan jalur solusi tercepat yaitu:

$A \rightarrow B \rightarrow G$

Jalur ini menunjukkan bahwa ketika kondisi awal langsung berada pada tingkat kekeruhan rendah, sistem tidak memerlukan proses filtrasi dan dapat langsung mencapai kondisi optimal. BFS menjamin solusi optimal karena mempertimbangkan jumlah langkah minimum menuju goal state. Namun demikian, metode BFS memiliki kelemahan pada penggunaan memori yang relatif besar karena harus menyimpan seluruh node pada setiap level pencarian.

3.3 Analisis Depth First Search (DFS)

Metode Depth First Search (DFS) bekerja dengan menelusuri satu jalur secara mendalam hingga mencapai kondisi akhir sebelum kembali (backtracking) ke jalur lainnya. Pada sistem ini, DFS pertama kali mengeksplorasi jalur:

$A \rightarrow B \rightarrow G$

Hasil tersebut menunjukkan bahwa DFS dapat menemukan solusi dengan cepat jika jalur awal langsung mengarah ke kondisi optimal. Namun, pada kondisi lain, DFS juga dapat menelusuri jalur yang lebih panjang, seperti melalui proses filtrasi berulang:

$A \rightarrow C \rightarrow F \rightarrow A \rightarrow D \rightarrow F \rightarrow A$

Hal ini menunjukkan bahwa DFS tidak selalu menghasilkan solusi optimal karena dapat terjebak pada jalur yang lebih dalam tanpa mempertimbangkan efisiensi langkah. Meskipun demikian, DFS memiliki keunggulan dalam penggunaan memori yang lebih kecil dibandingkan BFS.

3.4 Analisis Heuristic Search

Pendekatan heuristic digunakan untuk mengevaluasi beberapa kemungkinan jalur berdasarkan nilai estimasi terhadap kedekatan dengan kondisi optimal. Setiap state diberikan nilai heuristik, di mana state dengan nilai lebih kecil dianggap lebih dekat dengan goal state.

Berdasarkan perhitungan, beberapa jalur yang mungkin adalah:

$A \rightarrow B \rightarrow G$ (cost = 2)

$A \rightarrow C \rightarrow F \rightarrow B \rightarrow G$ (cost = 4)

$A \rightarrow D \rightarrow F \rightarrow B \rightarrow G$ (cost = 4)

$A \rightarrow D \rightarrow F \rightarrow C \rightarrow F \rightarrow B \rightarrow G$ (cost = 6)

Hasil analisis menunjukkan bahwa jalur $A \rightarrow B \rightarrow G$ merupakan jalur paling optimal karena memiliki nilai cost paling kecil. Sementara itu, jalur yang melibatkan proses filtrasi memiliki cost lebih tinggi karena memerlukan langkah tambahan.

Pendekatan heuristic terbukti membantu dalam mempercepat proses pencarian dengan memprioritaskan state yang lebih dekat ke tujuan.

3.5 Pembahasan

Berdasarkan hasil pengujian menggunakan tiga metode pencarian, dapat disimpulkan bahwa setiap metode memiliki karakteristik yang berbeda dalam menentukan solusi. BFS menghasilkan solusi optimal dengan jumlah langkah minimum, namun membutuhkan memori yang lebih besar. DFS lebih hemat memori tetapi tidak menjamin solusi terbaik karena bergantung pada urutan eksplorasi.

Sementara itu, pendekatan heuristic memberikan efisiensi tambahan dengan memanfaatkan informasi estimasi dalam memilih jalur terbaik. Dalam konteks sistem pengendalian kekeruhan air, kombinasi metode ini memberikan fleksibilitas dalam menentukan strategi pencarian yang sesuai dengan kondisi lingkungan.

Implementasi intelligent agent pada sistem ini menunjukkan kemampuan dalam melakukan pengambilan keputusan secara otomatis berdasarkan kondisi yang terdeteksi. Selain itu, adanya mekanisme evaluasi berulang memungkinkan sistem untuk beradaptasi terhadap perubahan kondisi air secara dinamis hingga mencapai keadaan optimal.

IV. KESIMPULAN

Berdasarkan hasil perancangan dan analisis yang telah dilakukan, penelitian ini berhasil mengimplementasikan intelligent agent untuk mengoptimalkan kondisi kekeruhan air kolam ikan secara otomatis menggunakan pendekatan state space search. Sistem yang dirancang mampu merepresentasikan kondisi lingkungan ke dalam beberapa state serta menentukan aksi yang sesuai berdasarkan tingkat kekeruhan air yang terdeteksi oleh sensor.

Penerapan algoritma pencarian seperti Breadth First Search (BFS), Depth First Search (DFS), dan heuristic search menunjukkan bahwa metode BFS mampu menghasilkan solusi optimal dengan jumlah langkah minimum, yaitu melalui jalur $A \rightarrow B \rightarrow G$. Sementara itu, DFS memiliki keunggulan dalam efisiensi penggunaan memori, namun tidak selalu menghasilkan solusi terbaik. Pendekatan heuristic memberikan efisiensi tambahan dengan membantu sistem dalam memilih jalur yang lebih dekat menuju kondisi optimal.

Selain itu, sistem yang dirancang memiliki kemampuan evaluasi berulang (iterative evaluation) yang memungkinkan agent untuk menyesuaikan tindakan terhadap perubahan kondisi lingkungan secara dinamis. Dengan demikian, sistem ini tidak hanya mampu melakukan pengambilan keputusan secara otomatis, tetapi juga adaptif terhadap kondisi nyata pada kolam ikan.

Ke depan, penelitian ini dapat dikembangkan lebih lanjut dengan mengintegrasikan metode pembelajaran mesin (machine learning) untuk meningkatkan akurasi prediksi kondisi air serta memperluas parameter yang diamati, seperti pH, suhu, dan kadar oksigen terlarut, sehingga sistem menjadi lebih komprehensif dalam mendukung budidaya ikan berbasis teknologi cerdas.

REFERENSI

- [1] P. Y. Yang, Y. C. Liao, and F. I. Chou, "Artificial Intelligence in Internet of Things System for Predicting Water Quality in Aquaculture Fishponds," *Computer Systems Science and Engineering*, vol. 46, no. 3, pp. 2862–2880, 2023, doi: 10.32604/csse.2023.036810.
- [2] F. Ardhia Maharani, M. Fathurahman, and A. Rachman, "Rancang Bangun Sistem Monitoring Kualitas Air Berbasis Internet Of Things (IoT) pada Tambak Ikan Bandeng," 2024.
- [3] I. Rizqullah and Y. Irawan, "Digital Transformation of Catfish Ponds with AI-based Monitoring System," *bit-Tech*, vol. 8, no. 1, pp. 789–798, Aug. 2025, doi: 10.32877/bt.v8i1.2716.