

Perancangan Sistem Agen Cerdas Berbasis Sensor untuk Pengangkat Jemuran Otomatis

Fauzan Hilmy Aries^{*1}, Anang Kukuh Adisusilo², Nonot Wisnu Karyanto³

^{1,2,3} Program Studi Informatika, Universitas Wijaya Kusuma Surabaya

Email: ¹ fha037@mhs.uwks.ac.id, ² anang65@uwks.ac.id, ³ nonotwk@uwks.ac.id

^{*}Penulis Korespondensi

Abstrak

Perkembangan kecerdasan buatan (Artificial Intelligence/AI) telah membuka peluang besar dalam otomasi berbagai aktivitas rumah tangga yang sebelumnya bergantung sepenuhnya pada intervensi manusia. Salah satu tantangan sederhana namun berdampak signifikan dalam kehidupan sehari-hari adalah pengelolaan jemuran pakaian di luar ruangan. Perubahan cuaca yang tidak terduga sering kali menyebabkan pakaian yang sedang dijemur menjadi basah karena hujan, terutama ketika penghuni rumah sedang bepergian atau beristirahat. Penelitian sebelumnya telah mengeksplorasi berbagai pendekatan otomasi rumah pintar (smart home), termasuk sistem pengendali berbasis Internet of Things (IoT) untuk pengambilan keputusan cuaca. Namun, pendekatan yang mengintegrasikan formulasi agen kecerdasan buatan secara formal mencakup pemodelan ruang keadaan dan algoritma pencarian masih relatif terbatas untuk kasus pengangkat jemuran otomatis[1].

Penelitian ini bertujuan merancang sistem pengangkat jemuran otomatis berbasis kecerdasan buatan menggunakan pendekatan intelligent agent. Sistem dimodelkan melalui task environment yang mencakup percept, action, goal, dan environment. Metode yang digunakan adalah beberapa algoritma pencarian, yaitu Breadth First Search (BFS), Depth First Search (DFS), Generate and Test, Hill Climbing, Greedy Best-First Search, dan A*. Hasil penelitian menunjukkan bahwa seluruh metode menghasilkan jalur solusi yang sama, yaitu $S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$ dengan total biaya 4. Namun, algoritma A* memberikan performa terbaik karena mampu mempertimbangkan biaya lintasan dan fungsi heuristik secara bersamaan. Sistem yang dirancang mampu bekerja secara otomatis dalam menjaga jemuran tetap kering dan mengurangi ketergantungan pengguna[1].

Kata kunci: BFS, DFS, A*, Heuristic Search, Generate and Test

Abstract

The development of artificial intelligence (AI) has opened up significant opportunities for automating various household activities that previously relied entirely on human intervention. One simple yet impactful challenge in daily life is managing outdoor clothes drying. Unexpected weather changes often cause clothes left to dry to get wet due to rain, especially when household members are traveling or resting. Previous research has explored various smart home automation approaches, including Internet of Things (IoT)-based control systems for weather-based decision-making. However, approaches that integrate artificial intelligence agent formulations, formally including state space modeling and search algorithms, are still relatively limited for the case of automatic clothesline lifting[1].

This study aims to design an automatic clothes drying rack lifting system based on artificial intelligence using an intelligent agent approach. The system is modeled through a task environment that includes percept, action, goal, and environment. The methods used are several search algorithms, namely Breadth First Search (BFS), Depth First Search (DFS), Generate and Test, Hill Climbing, Greedy Best-First Search, and A. The results of the study show that all methods produce the same solution path, namely $S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$ with a total cost of 4. However, the A* algorithm provides the best performance because it is able to consider both the path cost and the heuristic function simultaneously. The designed system is capable of working automatically to keep the laundry dry and reduce user dependency[1].*

Keywords: BFS, DFS, A*, Heuristic Search, Generate and Test

I. PENDAHULUAN

1.1. Latar Belakang

Cuaca di Indonesia bersifat tidak menentu, khususnya pada musim pancaroba di mana pergantian antara musim kemarau dan musim hujan sering terjadi secara tiba-tiba. Kondisi ini menimbulkan masalah praktis yang kerap dialami masyarakat sehari-hari, yaitu pakaian yang sedang dijemur di luar ruangan menjadi basah kembali akibat hujan yang datang tanpa prediksi yang cukup [1]. Permasalahan ini semakin terasa ketika penghuni rumah sedang bepergian, bekerja, atau tidak berada di rumah dalam jangka waktu lama [2].

Namun, sebagian besar penelitian tersebut berfokus pada aspek implementasi perangkat keras semata, tanpa memodelkan logika pengambilan keputusan agen secara formal menggunakan kerangka teori kecerdasan buatan. Russell dan Norvig [3] menegaskan bahwa sebuah sistem otomasi yang baik seharusnya dapat dianalisis sebagai *intelligent agent*—entitas yang mempersepsikan lingkungan melalui sensor dan mengambil aksi melalui aktuator secara rasional untuk mencapai tujuan yang telah ditetapkan. Kerangka PAGE (*Percept, Action, Goal, Environment*) menyediakan struktur formal untuk mendefinisikan komponen-komponen agen tersebut secara eksplisit dan terukur [3].

Berdasarkan uraian di atas, penelitian ini merancang sistem agen cerdas untuk pengangkat jemuran otomatis dengan memadukan kerangka PAGE, pemodelan ruang keadaan (*state space*), dan penerapan algoritma pencarian meliputi Breadth-First Search (BFS), Depth-First Search (DFS), Hill Climbing, Greedy, dan A* untuk menentukan jalur aksi yang optimal. Pendekatan ini menghasilkan sistem yang tidak hanya reaktif terhadap perubahan cuaca, tetapi juga dapat dianalisis kualitas keputusannya secara ilmiah dan terstruktur.

1.2. Identifikasi Masalah

Berdasarkan latar belakang yang telah diuraikan, maka permasalahan dalam penelitian ini dapat diidentifikasi sebagai berikut:

- 1) Bagaimana memodelkan sistem pengangkat jemuran otomatis sebagai *intelligent agent* berbasis task environment sehingga mampu bekerja secara mandiri.
- 2) Bagaimana menentukan representasi state space yang tepat untuk menggambarkan kondisi sistem dan lingkungan.
- 3) Bagaimana mengimplementasikan algoritma pencarian seperti BFS, DFS, Generate and Test, Hill Climbing, Greedy, dan A* dalam sistem.
- 4) Bagaimana membandingkan performa masing-masing algoritma dalam menentukan jalur solusi optimal berdasarkan efisiensi dan biaya pencarian.

1.3. Tujuan Penelitian

- 1) Merancang model *intelligent agent* pada sistem pengangkat jemuran otomatis menggunakan pendekatan PAGE (*Percept, Action, Goal, Environment*).
- 2) Mengimplementasikan berbagai algoritma pencarian dalam proses pengambilan keputusan sistem.
- 3) Menganalisis dan membandingkan performa algoritma BFS, DFS, Generate and Test, Hill Climbing, Greedy Best-First Search, dan A* dalam menentukan solusi optimal.
- 4) Menentukan algoritma terbaik yang dapat digunakan dalam sistem pengangkat jemuran otomatis berdasarkan efisiensi dan optimalitas solusi.

1.4. Penelitian Terkait

Penelitian mengenai penerapan algoritma pencarian dalam kecerdasan buatan telah banyak dilakukan dalam berbagai bidang. Studi sebelumnya menunjukkan bahwa algoritma BFS dan DFS sering digunakan sebagai dasar dalam pencarian solusi pada ruang keadaan (*state space*), meskipun memiliki keterbatasan dalam efisiensi dan optimalitas [1].

Penelitian lain menunjukkan bahwa algoritma berbasis heuristik seperti Hill Climbing, Greedy Best-First Search, dan A* mampu meningkatkan efisiensi pencarian dengan memanfaatkan estimasi jarak menuju goal. Algoritma A* secara khusus dikenal memiliki performa terbaik karena mampu mengombinasikan biaya lintasan dan fungsi heuristik secara optimal [4].

Dalam konteks sistem cerdas berbasis lingkungan, penerapan intelligent agent telah banyak digunakan untuk otomasi, seperti pada sistem pengendali suhu (smart thermostat) dan sistem kontrol rumah pintar (smart home). Pendekatan ini memungkinkan sistem untuk beradaptasi terhadap perubahan kondisi lingkungan secara real-time. Perbedaan penelitian ini dengan penelitian sebelumnya terletak pada penerapan berbagai algoritma pencarian dalam satu sistem pengangkat jemuran otomatis, sehingga memungkinkan dilakukan analisis komprehensif terhadap performa masing-masing algoritma dalam menyelesaikan permasalahan yang sama [5].

II. METODE PENELITIAN

2.1. Desain Task Environment Pengangkat Jemuran Otomatis

Pada penelitian ini, sistem pengangkat jemuran otomatis dimodelkan sebagai *intelligent agent* yang bekerja berdasarkan konsep task environment dalam kecerdasan buatan. Model ini menggunakan pendekatan PAGE (Percept, Action, Goal, Environment) sehingga sistem mampu mengambil keputusan secara otomatis berdasarkan kondisi lingkungan [3].

Percept pada sistem terdiri dari:

- 1) Status hujan (True/False)
- 2) Intensitas cahaya (terang/gelap)

Data diperoleh dari sensor hujan dan sensor cahaya yang membaca kondisi lingkungan secara real-time.

Action pada sistem meliputi:

- 1) Menarik jemuran ke dalam
- 2) Mengeluarkan jemuran ke luar
- 3) Berhenti (standby)

Goal utama sistem adalah menjaga jemuran tetap kering dan memastikan jemuran hanya berada di luar saat kondisi aman (tidak hujan dan terang).

Environment pada penelitian ini adalah:

- 1) Halaman rumah
- 2) Kondisi cuaca (hujan, cerah, mendung, malam)
- 3) Sistem rel jemuran otomatis

Percept sequence sistem dapat direpresentasikan sebagai berikut:

{[cerah, luar], [hujan, luar], [hujan, dalam], [cerah, dalam], ...}

2.2. Agent Function

Agent function dirancang untuk menentukan aksi berdasarkan kondisi lingkungan yang terdeteksi sensor.

Aturan agent function adalah sebagai berikut:

- 1) Jika hujan terdeteksi, maka sistem menjalankan aksi tarik_jemuran
- 2) Jika tidak hujan dan cahaya terang, maka sistem menjalankan aksi keluaran_jemuran
- 3) Jika kondisi tidak memenuhi keduanya, maka sistem menjalankan aksi diam (standby)

Representasi Fungsi Agent :

IF hujan == True THEN

 tarik_jemuran()

ELSE IF hujan == False AND cahaya == terang THEN

 keluaran_jemuran()

ELSE

 diam()

2.3. Agent Program

Program agent merupakan implementasi dari agent function dalam bentuk algoritma sederhana [3].

function JemuranAuto(hujan, cahaya)

 if hujan == True then

```

return tarik_jemuran
else if hujan == False AND cahaya == terang then
    return keluarkan_jemuran
else
    return diam
    
```

2.4. Formulasi Problem Pengangkat Jemuran Otomatis

2.4.1 Initial State

Keadaan awal sistem sebelum agen bekerja:

- 1) Jemuran berada di luar rumah
- 2) Kondisi cuaca belum diketahui (sensor belum membaca)
- 3) Sistem dalam keadaan standby

2.4.2 Actions / Successor Function

Successor function digunakan untuk menghasilkan state baru dari suatu aksi.

Tabel 1. Successor Function

Aksi	Kondisi	State Baru
tarik_jemuran	hujan	dalam
keluarkan_jemuran	cerah	luar
diam	mendung/malam	tetap

Contoh transisi:

- 1) $S_0 \rightarrow S_1$ (hujan)
- 2) $S_1 \rightarrow S_2$ (tarik)
- 3) $S_2 \rightarrow S_3$ (cerah)

2.4.3 Goal Test

Kondisi goal:

- 1) Jemuran tidak terkena hujan
- 2) Jemuran berada di dalam saat hujan
- 3) Jemuran berada di luar saat cerah

Goal tercapai jika:

- 1) Hujan = posisi di dalam
- 2) Cerah = posisi di luar

2.4.4 Path Cost

Path cost digunakan untuk menghitung biaya setiap aksi.

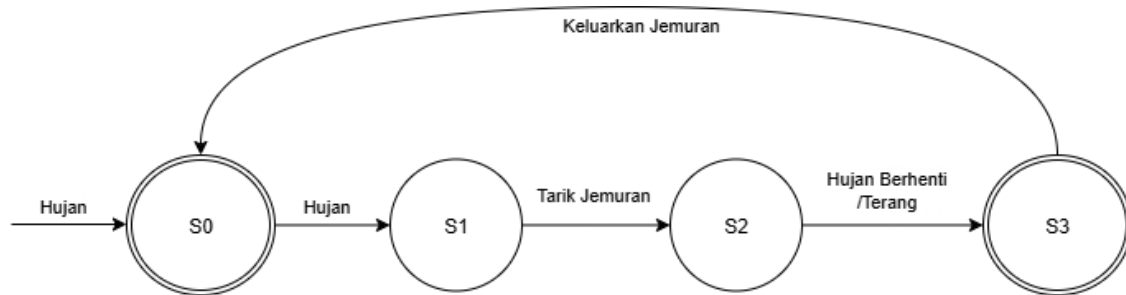
- 1) Tarik jemuran = 2
- 2) Keluarkan jemuran = 2
- 3) Diam = 0

Contoh:

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$

Cost = $2 + 2 = 4$

2.5. State Space Graph



Gambar 1. State Space Graph

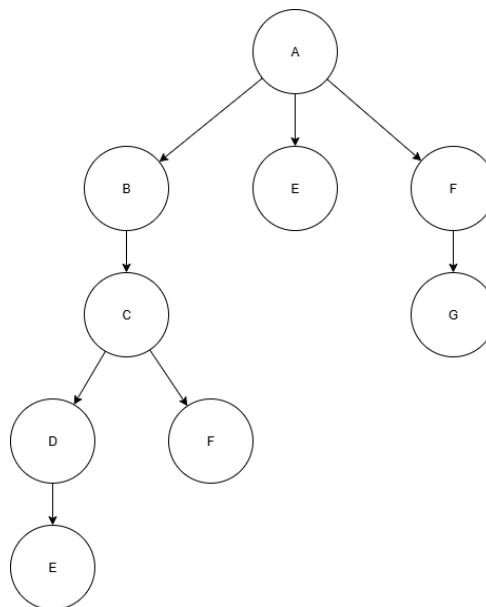
State (keadaan sistem):

- 1) S0: Jemuran di luar, cuaca cerah
- 2) S1: Jemuran di luar, hujan
- 3) S2: Jemuran di dalam, hujan
- 4) S3: Jemuran di dalam, tidak hujan

Transisi:

- 1) S0 ke (hujan) ke S1
- 2) S1 ke (tarik_jemuran) ke S2
- 3) S2 ke (hujan berhenti + terang) ke S3
- 4) S3 ke (Keluarkan_jemuran) ke S0

2.6. Tree Terminology



Gambar 2. Tree Terminology

Level 0 (Root Node)

- 1) Node A (S0): Kondisi Awal. Jemuran berada di luar dan cuaca cerah. Ini adalah titik awal sistem memantau keadaan.

Level 1 (Cabang Pertama - Hasil Pemantauan)

- 1) Node B (S1): Hujan Turun. Sensor mendeteksi hujan saat jemuran masih di luar. Sistem harus mengambil tindakan.
- 2) Node E (Tengah) (S6): Cuaca Tetap Cerah. Tidak ada perubahan, jemuran tetap aman di luar. Ini adalah *Leaf Node* (titik akhir) karena tidak butuh tindakan lanjutan.
- 3) Node F (Kanan) (S7): Mendung/Gelap. Sensor mendeteksi mendung, tapi belum hujan. Kondisi ini butuh pantauan lanjutan.

Level 2 (Tindakan Pertama)

- 1) Node C (Anak dari B) (S2): Jemuran Ditarik. Karena Node B mendeteksi hujan, sistem melakukan aksi menarik jemuran ke dalam. Status saat ini: Jemuran di dalam, cuaca hujan.
- 2) Node G (Anak dari F Kanan) (S8): Standby. Merespons Node F (mendung), sistem memilih untuk diam/standby dan tidak buru-buru menarik jemuran. Ini adalah *Leaf Node*.

Level 3 (Pemantauan Lanjutan di Dalam)

- 1) Node D (Anak dari C) (S3): Hujan Berhenti & Cerah. Saat jemuran ada di dalam (Node C), sensor mendeteksi cuaca kembali terang. Ini memicu perlunya tindakan baru.
- 2) Node F (Anak dari C, Kanan-Bawah) (S5): Masih Hujan. Cuaca tidak berubah. Jemuran tetap dibiarkan di dalam. Ini adalah *Leaf Node*.

Level 4 (Tindakan Akhir)

- 1) Node E (Anak dari D) (S4): Jemuran Dikeluarkan Lagi. Karena Node D mendeteksi cuaca sudah cerah kembali, sistem mengeluarkan jemuran. Ini adalah *Leaf Node* (tujuan akhir tercapai).

2.7. Implementasi Algoritma Pencarian

- 1) Breadth First Search (BFS), melakukan pencarian berdasarkan level. Menghasilkan solusi optimal tetapi membutuhkan memori besar[3].
- 2) Depth First Search (DFS), menelusuri cabang terdalam terlebih dahulu. Cepat namun tidak menjamin solusi optimal[3].
- 3) Generate and Test, menghasilkan kemungkinan solusi lalu menguji apakah mencapai goal.
- 4) Hill Climbing, menggunakan pendekatan heuristik dengan memilih solusi terbaik lokal.
- 5) Greedy Best First Search, memilih node berdasarkan nilai heuristik terkecil.
- 6) A* Algorithm, menggunakan fungsi evaluasi:

$$f(n) = g(n) + h(n)$$

Dimana :

$g(n)$: biaya dari start

$h(n)$: estimasi ke goal

Algoritma A* digunakan karena mampu menghasilkan solusi optimal secara efisien [3].

III. HASIL DAN PEMBAHASAN

3.1. Hasil Generate and Test

Metode Generate and Test bekerja dengan cara membangkitkan berbagai kemungkinan jalur dari state awal menuju goal, kemudian menguji apakah jalur tersebut memenuhi kondisi tujuan. Pada sistem pengangkat jemuran otomatis, beberapa kemungkinan jalur yang dihasilkan ditunjukkan pada tabel dibawah :

Table 2. Hasil Generate and Test

Jalur	Perhitungan Cost	Hasil
$S_0 \rightarrow \text{diam} \rightarrow S_0$	0	Gagal
$S_0 \rightarrow S_1 \rightarrow \text{diam} \rightarrow S_1$	0	Gagal (jemuran basah)
$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$	$2 + 2$	SUKSES

Berdasarkan hasil pengujian, jalur yang berhasil mencapai kondisi aman (goal) adalah:

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$

Total cost sebesar 4

Metode ini sederhana dalam implementasi, namun kurang efisien karena harus mencoba banyak kemungkinan jalur, terutama jika ruang pencarian semakin besar [3].

3.2. Hasil Hill Climbing

Metode Hill Climbing menggunakan pendekatan heuristik dengan memilih langkah yang memberikan kondisi terbaik (paling dekat dengan goal). Nilai heuristik (h) ditentukan berdasarkan kedekatan kondisi terhadap state aman (jemuran kering).

Table 3. Hasil Hill Climbing

Langkah	State	Tetangga	$h(n)$	Keputusan
1	S_0	S_1	2	Pilih S_1
2	S_1	S_2	1	Pilih S_2
3	S_2	S_3	0	Pilih S_3

Jalur yang dihasilkan:

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$

Total cost sebesar 4

Hill Climbing terbukti lebih cepat karena hanya memilih jalur terbaik pada setiap langkah. Namun, metode ini berpotensi mengalami local optimum jika kondisi lingkungan lebih kompleks [3].

3.3. Hasil Greedy Best-First Search

Greedy Best-First Search memilih node dengan nilai heuristik terkecil tanpa mempertimbangkan biaya yang telah ditempuh.

Nilai heuristik yang digunakan:

- 1) $S_0 = 2$
- 2) $S_1 = 2$
- 3) $S_2 = 1$
- 4) $S_3 = 0$

Table 4. Hasil Greedy BFS

Fringe	Node Dipilih
[$S_0(2)$]	S_0
[$S_1(2)$]	S_1
[$S_2(1)$]	S_2
[$S_3(0)$]	S_3

Jalur yang dihasilkan:

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$

Total cost sebesar 4

Metode ini lebih cepat dibanding BFS karena hanya mempertimbangkan heuristik, namun tidak selalu menjamin solusi optimal pada kasus yang lebih kompleks [3].

3.4. Hasil A Algorithm*

Algoritma A* menggunakan fungsi evaluasi:

$$f(n)=g(n)+h(n) \quad f(n) = g(n) + h(n)$$

dengan:

- 1) $g(n)$ = biaya dari start
- 2) $h(n)$ = estimasi ke goal

Table 5. Hasil A*

Node	$g(n)$	$h(n)$	$f(n)$	Keputusan
S1	2	2	4	Dipilih
S2	4	1	5	Dipilih
S3	4	0	4	GOAL

Jalur optimal yang dihasilkan:

$S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$

Total cost sebesar 4

Hasil ini menunjukkan bahwa algoritma A* mampu menghasilkan solusi optimal dengan mempertimbangkan biaya dan heuristik secara bersamaan [3].

3.5. Perbandingan Keseluruhan Algoritma

Table 6. Perbandingan Keseluruhan Algoritma

Algoritma	Jalur	Cost	Optimal	Keterangan
Generate & Test	$S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$	4	Ya	Tidak efisien
Hill Climbing	$S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$	4	Tidak selalu	Cepat
Greedy	$S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$	4	Tidak selalu	Heuristik
A*	$S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$	4	Ya	Paling optimal

Berdasarkan hasil pengujian:

- 1) BFS dan A* menghasilkan solusi optimal
- 2) DFS lebih cepat tetapi tidak selalu optimal
- 3) Hill Climbing dan Greedy lebih efisien tetapi berisiko salah
- 4) A* memberikan keseimbangan terbaik antara efisiensi dan optimalitas

IV. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa sistem pengangkat jemuran otomatis berbasis kecerdasan buatan berhasil dirancang menggunakan pendekatan *intelligent agent* dengan model *task environment* yang meliputi *percept*, *action*, *goal*, dan *environment*. Sistem mampu mendeteksi kondisi lingkungan melalui sensor hujan dan cahaya, kemudian menentukan aksi secara otomatis untuk menjaga jemuran tetap kering.

Hasil implementasi berbagai algoritma pencarian, yaitu Breadth First Search (BFS), Depth First Search (DFS), Generate and Test, Hill Climbing, Greedy Best-First Search, dan A* menunjukkan bahwa seluruh metode mampu menemukan jalur solusi yang sama, yaitu $S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$. Jalur tersebut merupakan kondisi optimal yang memastikan jemuran tidak terkena hujan.

Dari hasil perbandingan, algoritma A* memberikan performa terbaik karena mampu menghasilkan solusi optimal dengan mempertimbangkan biaya lintasan dan fungsi heuristik secara bersamaan. Sementara itu, algoritma BFS juga menghasilkan solusi optimal namun dengan kebutuhan memori yang lebih besar, sedangkan DFS, Hill Climbing, dan Greedy memiliki keunggulan dalam kecepatan tetapi tidak selalu menjamin optimalitas pada kondisi yang lebih kompleks.

Dengan demikian, sistem yang dirancang tidak hanya mampu bekerja secara otomatis, tetapi juga dapat dianalisis secara ilmiah menggunakan pendekatan kecerdasan buatan. Untuk pengembangan selanjutnya, sistem dapat ditingkatkan dengan menambahkan parameter lingkungan lain, integrasi

perangkat keras secara langsung, serta penerapan perhitungan biaya berbasis energi agar sistem menjadi lebih realistis dan efisien.

REFERENSI

- [1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, dan M. Zorzi, “Internet of Things for Smart Cities,” *IEEE Internet of Things Journal*, vol. 1, no. 1, hlm. 22–32, Feb 2014, doi: 10.1109/JIOT.2014.2306328.
- [2] B. Alvando, “PROTOTYPE JEMURAN PAKAIAN OTOMATIS MENGGUNAKAN ARDUINO UNO,” *JURNAL PERENCANAAN, SAINS DAN TEKNOLOGI (JUPERSATEK)*, vol. 4, no. 1, hlm. 771–775, Jul 2021.
- [3] S. J. Russell, S. Russell, dan P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson, 2020.
- [4] K. A. A. Pahlevi, A. I. Pradana, dan D. Hartanti, “Rancang Bangun Sistem Jemuran Otomatis Berbasis Internet Of Things (IoT),” *Adopsi Teknologi dan Sistem Informasi (ATASI)*, vol. 4, no. 2, hlm. 148–155, Sep 2025, doi: 10.30872/atasi.v4i2.3174.
- [5] R. R. Mastiyanto, A. H. Al-Azhari, dan D. Djuniadi, “ATAP JEMURAN CERDAS BERBASIS ARDUINO DENGAN SENSOR HUJAN DAN CAHAYA,” vol. 13, no. 1, 2024.