Penerapan Algoritma A* dalam Mencari Rute Terpendek dari Pintu Utara I ke Pintu Keluar Timur Kebun Binatang Ragunan melalui Kadang Hewan Primata

Nur Anissah Mardiyanti [1, a)], Gabriel Olivia Yvonne Manurung [2, b)], Gilbert Danielson Jonathan Otto Mamesah [3, c)]

Program Studi Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta

Email: [a)]*nuranissahmardiyanti_1305621036@mhs.unj.ac.id*,
[b)]*gabrieloliviayvonnemanurung_1305621023@mhs.unj.ac.id*,
[c)]*gilbertdanielsonjonathanottomamesah_1305621015@mhs.unj.ac.id*

Abstract

Ragunan Zoo is one of the popular destinations in Jakarta, Indonesia, offering a variety of attractions and sightseeing spots. However, navigation within it can be complex, such as finding the shortest route from Gate North I to East Exit through the Primate Animal Enclosure. To address this navigation challenge, the A* Algorithm (Astar) is considered as a potential solution. The A* Algorithm is an efficient search approach capable of finding the shortest path. This research aims to evaluate the performance of the A* algorithm in finding the shortest routes in complex environments such as zoos. By utilizing appropriate heuristic functions, simulations are conducted to provide a better understanding of the A* algorithm's ability to navigate confined areas such as zoos. The research method includes the implementation steps of the A* algorithm and the analysis of experimental results comparing the generated routes with manually searched routes using Microsoft Excel and Python programming. It is hoped that the results of this research can contribute to the development of efficient navigation applications and enrich the visitor experience at Ragunan Zoo.

Keywords: A* Algorithm, Shortest Path, Ragunan

Abstrak

Kebun Binatang Ragunan adalah salah satu destinasi populer di Jakarta, Indonesia, yang menawarkan beragam atraksi dan tempat wisata. Namun, navigasi di dalamnya dapat menjadi rumit, seperti ketika mencari rute terpendek dari Pintu Utara I ke Pintu Keluar Timur melalui Kadang Hewan Primata. Untuk mengatasi tantangan navigasi ini, Algoritma A* (Astar) dipertimbangkan sebagai solusi yang potensial. Algoritma A* adalah pendekatan pencarian yang efisien dan dapat membaca jalur terpendek. Penelitian ini bertujuan untuk mengevaluasi kinerja algoritma A* dalam mencari rute terpendek di lingkungan kompleks seperti kebun binatang. Dengan memanfaatkan fungsi heuristik yang sesuai, simulasi dilakukan untuk memberikan pemahaman yang lebih baik tentang kemampuan algoritma A* dalam menavigasi area-area terbatas seperti kebun binatang. Metode penelitian mencakup langkah-langkah implementasi algoritma A* dan analisis hasil eksperimen yang membandingkan rute yang dihasilkan dengan metode pencarian rute secara manual menggunakan Microsoft Excel dan pemrograman Python. Harapannya, hasil penelitian ini dapat memberikan kontribusi pada pengembangan aplikasi navigasi yang efisien dan memperkaya pengalaman pengunjung di Kebun Binatang Ragunan.

Kata-kata kunci: Algoritma A*, Jarak Terpendek, Ragunan

PENDAHULUAN

Kebun Binatang Ragunan yang terletak di Jakarta Selatan, Indonesia, merupakan salah satu objek wisata favorit bagi masyarakat lokal dan wisatawan. Berdasarkan situs resmi Taman Margasatwa Ragunan, kebun binatang ini didirikan pada tanggal 19 September tahun 1864 dengan luas mencapai 147 hektar. Kebun Binatang Ragunan menampung lebih dari 2.000 hewan dari 220 spesies, termasuk hewan primata. Primata adalah hewan vertebrata yang termasuk dalam subkelas *eutherian*, yang dikenal sebagai mamalia sejati atau mamalia berplasenta yang tidak memiliki kantong (Pawestri dkk., 2021). Karakteristik yang sering ditemui pada primata meliputi memiliki lima jari, gigi dengan bentuk yang serupa, dan rancangan tubuh yang tergolong primitif. Selain itu, salah satu fitur khas dari primata adalah adanya kuku di jari-jarinya, serta arah ibu jari yang berbeda menjadi salah satu ciri khas mereka (Azwir dkk., 2021).

Menjelajahi Kebun Binatang Ragunan dan mengunjungi kandang hewan primata dapat menjadi pengalaman yang menyenangkan dan edukatif. Namun, dengan luasnya kebun binatang ini dapat membuat pengunjung merasa bingung dan kehilangan arah menyebabkan pengunjung berjalan kaki cukup jauh sehingga memakan waktu dan melelahkan. Meskipun Kebun Binatang Ragunan menyediakan kendaraan internal seperti kereta keliling dan sepeda wisata untuk membantu pengunjung berkeliling, pilihan ini mungkin tidak sesuai untuk pengunjung yang ingin menghemat biaya, mengamati satwa liar lebih dekat, atau ingin menjelajahi dengan lebih bebas.

Kebun Binatang Ragunan memiliki empat pintu masuk dan keluar yaitu pintu utara, pintu barat, pintu timur, dan pintu selatan. Berdasarkan akun instagram resmi Ragunan, pintu utara dan pintu barat merupakan pintu masuk utama yang digunakan oleh pengunjung, sedangkan pintu timur dan pintu selatan umumnya digunakan untuk keluar setelah selesai berkeliling. Bagi pengunjung yang ingin berkeliling kebun binatang dan mengunjungi kandang hewan primata dengan waktu yang singkat, penting untuk mengetahui rute terpendek antara pintu utara dan pintu keluar timur.

Di era teknologi dan informasi seperti sekarang ini, menemukan rute terpendek antar kandang hewan primata di Kebun Binatang Ragunan menjadi semakin mudah. Dengan memanfaatkan Google Maps yaitu sebuah jasa peta online hasil karya Google, pencarian informasi geografis dapat dilakukan kapanpun dan dimanapun. Pencarian rute terpendek tentunya membutuhkan suatu algoritma sehingga didapat rute paling efisien. Terdapat banyak algoritma dalam penyelesaian masalah pencarian rute terpendek, salah satunya adalah algoritma A-star (A*).

Algoritma A* pertama kali diperkenalkan oleh Peter Hart, Nils Nilsson, dan Bertram Raphael pada tahun 1968 dengan metode heuristik (Hart dkk., 1968). Algoritma A* adalah algoritma pencarian jalur dalam graf dari sebuah titik awal yang diberikan menuju sebuah titik tujuan yang diberikan. Algoritma ini menggunakan "perkiraan heuristik" h(x) yang memberikan estimasi jalur terbaik yang melalui titik tersebut. Algoritma A* mengunjungi titik-titik berdasarkan urutan dari estimasi heuristik ini (Rachmawati & Gustin, 2020). Algoritma A* dapat memilih jalur terpendek dalam waktu yang lebih cepat, tetapi tidak selalu optimal (Candra dkk., 2020).

Dalam penelitian ini, algoritma A* diterapkan untuk untuk membantu pengunjung menemukan rute terpendek antara Pintu Utara I dan Pintu Keluar Timur melalui kandang hewan primata.

METODE

A*

Fungsi heuristik yang baik memberikan perkiraan biaya mendekati biaya sebenarnya. Salah satu fungsi heuristik yang bisa digunakan dalam permasalahan jalur terpendek adalah *Euclidean Distance* (Candra et al., 2020). A* menggunakan *Euclidean Distance* dan mengevaluasi node dengan menggabungkan $g(n)$ dan $h(n)$ sebagai (Hart dkk., 1968):

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

dimana:

$f(n) =$ biaya evaluasi

$g(n) =$ biaya yang sudah dikeluarkan dari keadaan awal ke $state$ n

$h(n) =$ perkiraan biaya untuk sampai di tujuan dari $state$ n

Berikut adalah langkah-langkah algoritma A* (Candra et al., 2020):

1) Mulai
2) Tetapkan *node* "Mulai" sebagai *successor*.
3) Hitung semua biaya evaluasi yang terhubung dari *successor*.
4) Lakukan pengujian: jika calon *successor* yang menjadi tujuannya, itu akan berhenti, dan jika tidak, lanjutkan ke langkah berikutnya.
5) Menentukan *successor* baru dari evaluasi terbaik biaya (biaya minimum) dan urutan abjad (jika ada dua biaya evaluasi mempunyai nilai yang sama) dari *successor* sebelumnya.
6) Selanjutnya, kembali ke langkah 4.

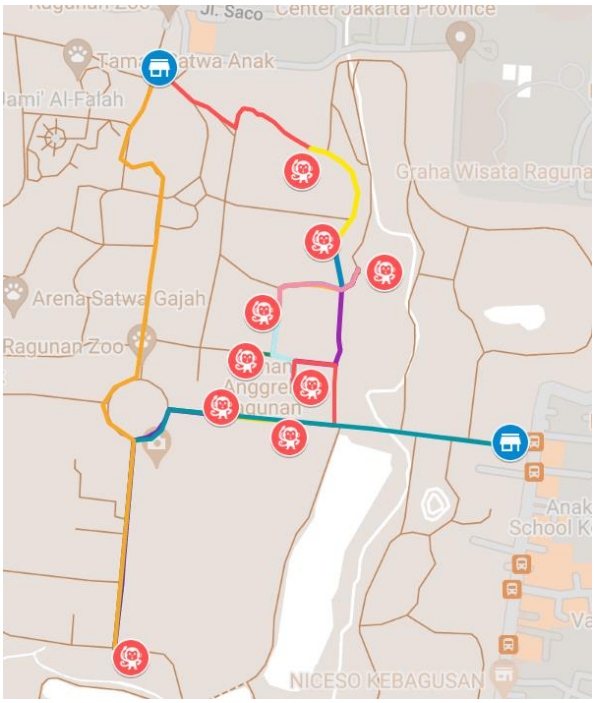
Teknik Pengumpulan Data

Teknik pengumpulan data untuk penelitian ini adalah dokumentasi, yaitu memperoleh data dan informasi berupa buku, catatan, dokumen, gambar tertulis dan visual dalam bentuk laporan, serta data yang dapat menunjang penelitian (Sugiyono, 2017). Kami menggunakan teknik ini dimana kami mengambil koordinat dari semua titik yang diamati menggunakan *google maps*.

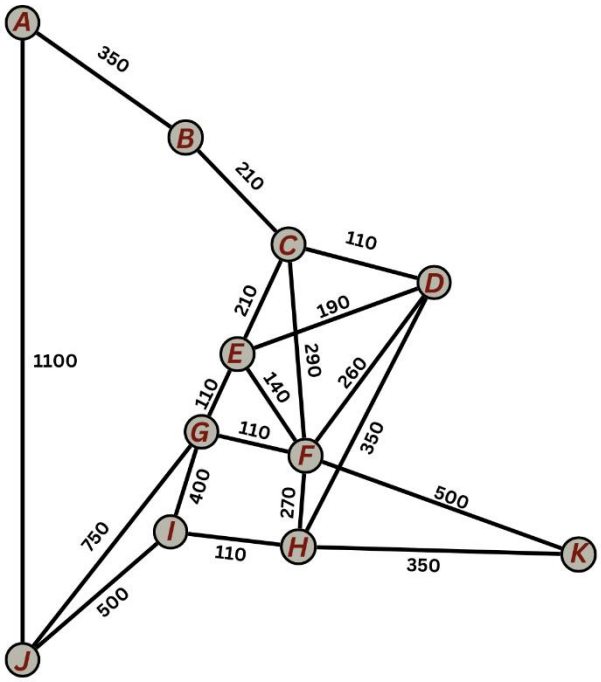
PEMBAHASAN

Pada bagian ini akan mengeksplorasi implementasi dan hasil dari penerapan algoritma A-star dalam mencari rute terpendek dari Pintu Utara I ke Pintu Keluar Timur Kebun Binatang Ragunan melalui Kadang Hewan Primata seperti orang utan, siamang, monyet, dan lainnya. Penelitian ini bertujuan untuk memberikan pemahaman yang lebih dalam tentang efektivitas algoritma A* dalam konteks pemetaan rute pada suatu area kompleks seperti kebun binatang.

Langkah awal yang perlu dilakukan adalah mengumpulkan data-data yang diperlukan dalam menghitung nilai heuristik yang akan digunakan dalam menentukan rute terpendek dengan menggunakan Algoritma A-star. Data yang diperlukan berupa jarak dan titik koordinat dari setiap titik atau *node* yang akan diamati melalui Google Maps sebagai berikut.



Gambar 1 Peta Kebun Binatang Ragunan
Sumber: *Google Maps*



Gambar 2 Graf Peta Kebun Binatang Ragunan dari Pintu Utara I sampai Pintu Timur

Keterangan:

A : Pintu masuk utara 1

B : Kandang primata

C : Kandang orang utan kalimantan

D : Kandang siamang

E : Kandang mamalia kecil

F : Kandang orang utan

G : Kandang orang utan

H : Kandang orang utan kalimantan

I : Kandang orang utan sumatra

J : Kandang siamang terbuka

K : Loket Timur

Jarak atau Bobot antar Titik

Titik	Jarak (m)
A-B	350
B-C	210
C-D	110
C-E	210
C-F	290
D-C	110
D-E	190
D-F	260
D-H	350
E-C	210
E-D	190
E-G	110
E-F	140
F-C	290
F-D	260
F-E	140
F-G	110
F-H	270
F-K	500
G-F	110
G-I	400
G-J	750
H-D	350
H-F	270
H-I	110
H-K	350
I-G	400
I-H	110
I-J	500
J-G	750
J-I	500
J-A	1100

Gambar 3 Jarak antar Titik atau *Node*.

Kemudian kumpulkan data koordinat X dan Y dari setiap titik melalui *Google Maps*.

No	Titik	Nama Titik	Koordinat	
			X	Y
1	A	Pintu Masuk Utara 1	106,82022	-6,30665
2	B	Primata	106,82229	-6,30816
3	C	Orang Utan Kalimantan	106,82259	-6,30917
4	D	Siamang	106,82349	-6,30962
5	E	Mamalia Kecil	106,82171	-6,31019
6	F	Orang Utan 1	106,82243	-6,31128
7	G	Orang Utan 2	106,82149	-6,3109
8	H	Orang Utan Kalimantan	106,82210	-6,312
9	I	Orang Utan Sumatra	106,82112	-6,31157
10	J	Siamang Terbuka	106,81980	-6,31518
11	K	Loket Timur	106,82532	-6,31206

Gambar 4 Koordinat Setiap Titik atau *Node*.

Mencari Nilai Heuristik

Untuk mencari nilai heuristik, dapat menggunakan rumus berikut:

$$h(n) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

kemudian cari nilai heuristik dari setiap titik yang sudah dikumpulkan dengan rumus tersebut sehingga didapatkan hasil perhitungan setiap titik menuju titik akhir sebagai berikut:

A ke K		
No	Titik	h(n)
1	AK	0,007435
2	BK	0,004938
3	CK	0,003976
4	DK	0,003050
5	EK	0,004066
6	FK	0,002993
7	GK	0,004002
8	HK	0,003221
9	IK	0,004228
10	JK	0,006341

Gambar 5 Perhitungan Nilai Heuristik

Simulasi Algoritmat A* (A-Star)

Iterasi	Node	f(n)	
1	A-B	350,004938	
	A-J	1100,006341	HOLD
2	A-B-C	560,003976	
3	A-B-C-D	670,003050	
	A-B-C-E	770,004066	HOLD
	A-B-C-F	850,002993	HOLD
4	A-B-C-D-E	860,004066	
	A-B-C-D-F	930,002993	HOLD
	A-B-C-D-H	1020,003221	HOLD
5	A-B-C-D-E-G	970,004002	
	A-B-C-D-E-F	1000,002993	HOLD
6	A-B-C-D-E-G-F	1080,002993	
	A-B-C-D-E-G-I	1370,004228	HOLD
	A-B-C-D-E-G-J	1720,006341	HOLD
7	A-B-C-D-E-G-F-H	1350,003221	HOLD
	A-B-C-D-E-G-F-K	1580	

Gambar 6 Langkah Pertama

Sebelum melakukan langkah pertama pada simulasi Algoritma A-Star, kita memerlukan f(n) dalam menentukan jarak terpendek dari titik awal hingga titik akhir dimana,

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

Keterangan:

g(n) = jarak antar titik

h(n) = nilai heauristik

Pada langkah pertama, ambil titik awal A kemudian perbandingkan antara f(n) A ke B dengan A ke J lalu cari f(n) terkecil yaitu titik A ke B sehingga dapat dilanjutkan ke percabangan titik B. Untuk f(n) yang lebih besar di*hold* (tahan) yang digunakan untuk dibandingkan dengan f(n) akhir. Kemudian lakukan cara yang sama sehingga didapatkan rute dari A ke K. Pada langkah pertama ini didapatkan rute A-B-C-D-E-G-F-K yang memiliki f(n) = 1580 antara titik A dan titik K. Bandingkan f(n) tersebut dengan f(n) yang tertahan, apabila terdapat f(n) yang lebih kecil makan ambil rute tersebut dan lanjutkan ke langkah berikutnya untuk mendapatkan kemungkinan rute dari A ke K yang lain.

Dari langkah pertama, terdapat 2 rute yang dapat dilanjutkan ke langkah kedua yaitu A-B-C-E dan A-B-C-F

Iterasi	Node	f(n)	
1	A-B-C-E	770	
2	A-B-C-E-D	960,003050	HOLD
	A-B-C-E-F	910,002993	HOLD
	A-B-C-E-G	880,004002	
3	A-B-C-E-G-F	990,002993	
	A-B-C-E-G-I	1280,004228	HOLD
	A-B-C-E-G-J	1630,006341	HOLD
4	A-B-C-E-G-F-D	1250,003050	
	A-B-C-E-G-F-H	1260,003221	HOLD
	A-B-C-E-G-F-K	1490	HOLD
5	A-B-C-E-G-F-D-H	1600,003221	
6	A-B-C-E-G-F-D-H-K	1950	

Gambar 7 Langkah Kedua

Pertama, eksekusi rute A-B-C-E dengan cara yang sama pada langkah pertama untuk mencari kemungkinan rute lain yang bisa ditemukan. Pada perhitungan tersebut, didapatkan dua rute lain dari A ke K yaitu A-B-C-E-G-F-K dengan f(n) = 1490 dan A-B-C-E-G-F-D-H-K dengan f(n) = 1950. Kemudian ketika dibandingkan dengan f(n) lain yang sudah di*hold*, terdapat satu rute yang dapat dieksekusi pada langkah berikutnya yaitu A-B-C-E-F dengan f(n) = 910,003.

Selanjutkan kita eksekusi rute A-B-C-F terlebih dahulu.

Iterasi	Node	f(n)	
1	A-B-C-F	850	
2	A-B-C-F-D	1110,003050	
	A-B-C-F-K	1350	HOLD
	A-B-C-F-H	1120,003221	HOLD
3	A-B-C-F-D-H	1460,003221	
4	A-B-C-F-D-H-K	1810	

Gambar 8 Langkah Ketiga

Pada perhitungan langkah ketiga, didapatkan rute baru yaitu A-B-C-F-D-H-K dengan f(n) = 1810 dan A-B-C-F-K dengan f(n) = 1350. Kemudian bandingkan dengan f(n) lain yang sudah

di*hold*, terdapat satu rute kemungkinan yang bisa dieksekusi pada langkah berikutnya yaitu A-B-C-F-H dengan f(n) = 1120,003.

Selanjutnya eksekusi rute A-B-C-E-F dan A-B-C-F-H dengan cara yang sama.

Iterasi	Node	f(n)
1	A-B-C-E-F	910
2	A-B-C-E-F-K	1410

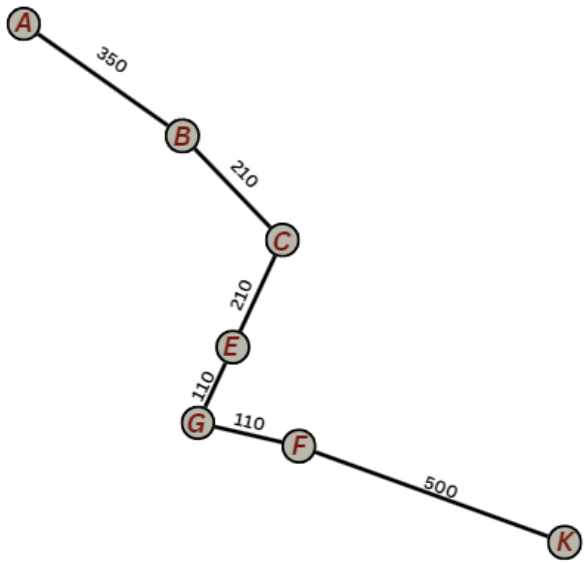
Gambar 9 A-B-C-E-F

Iterasi	Node	f(n)
1	A-B-C-F-H	1120
2	A-B-C-F-H-K	1470

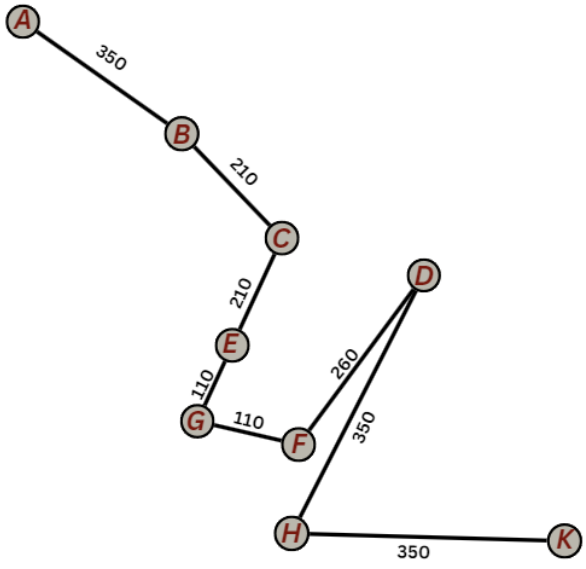
Gambar 10 A-B-C-F-H

Didapatkan dua rute baru yaitu A-B-C-E-F-K dengan f(n) = 1410 dan A-B-C-F-H-K dengan f(n) = 1470.

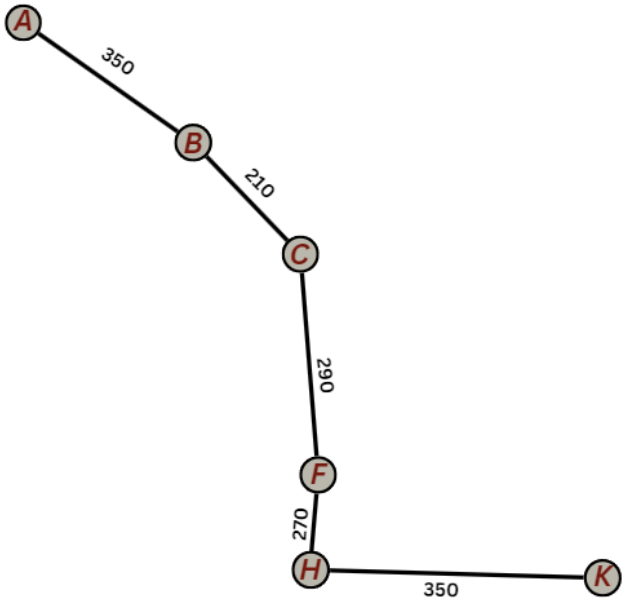
Dari seluruh perhitungan, diperoleh 7 rute dari A (Pintu Masuk Utara I) menuju K (Pintu Timur). Berikut adalah rute dan grafnya.



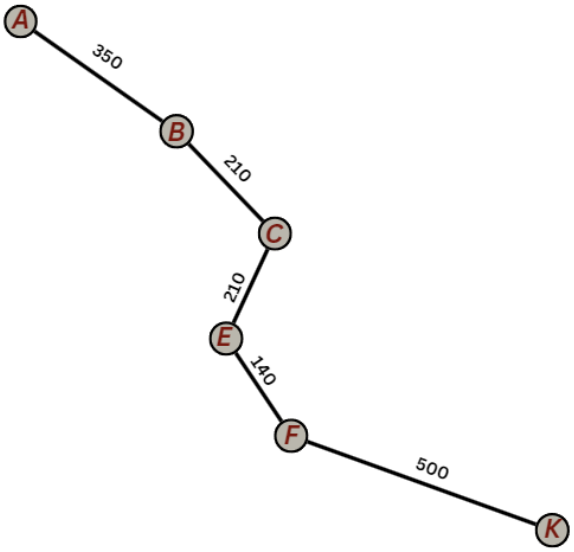
Gambar 11
Graf A -B-C-E-G-F-K dengan f(n) = 1490



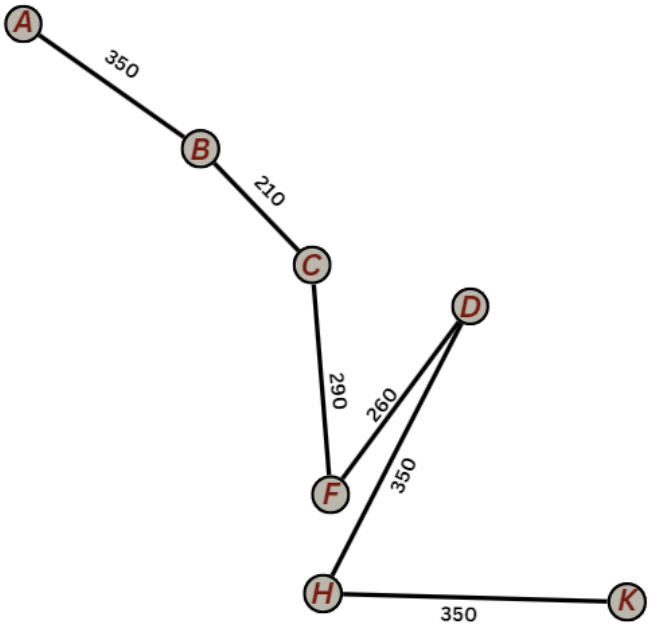
Gambar 12
A-B-C-E-G-F-D-H -K dengan f(n) = 1950



Gambar 15
Graf A-B-C-F-D-H-K dengan f(n) = 1810



Gambar 16
Graf A-B-C-E-F-K dengan f(n) = 1410



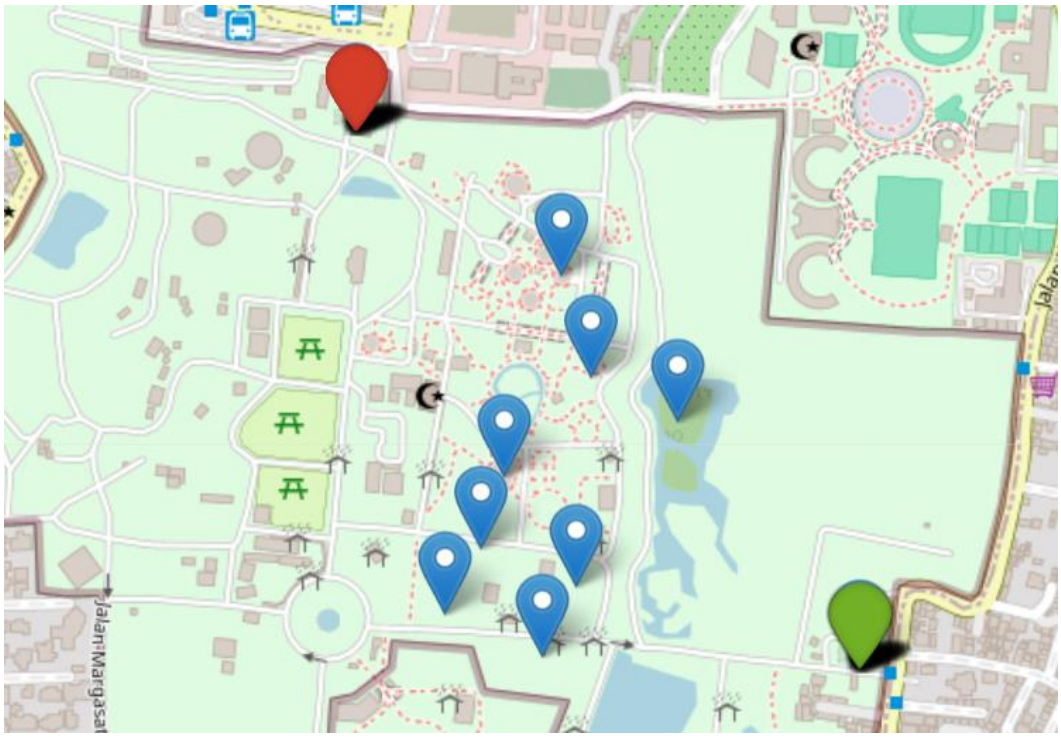
Gambar 17
Graf A-B-C-F-H-K dengan f(n) = 1470

Dari ketujuh graf di atas, terlihat bahwa jarak terpendek yang dapat dilalui untuk sampai ke titik K (Pintu Timur) dari titik A (Pintu Utara I) yaitu pada Gambar 14 graf A-B-C-F-K dengan jarak tempuh 1350 meter.

Simulasi Algoritma A-Star menggunakan *Google Collab*

Visualisasi peta

```
# Gambaran titik A, B, C, D, E, F, G, H, I, J, K pada google maps
map_ = fl.Map(location=[-6.310804742573237, 106.82004172812191],
zoom_start=16)

for (index, row) in dataset.iterrows():
  fl.Marker(location=[row.loc['Y'], row.loc['X']],
            popup=row.loc['Nama Titik'],
            tooltip=row.loc['Titik']).add_to(map_)
  fl.Marker(location=[-6.30665, 106.82022], icon=fl.Icon(color='red',
prefix='fa')).add_to(map_)
  fl.Marker(location=[-6.31206, 106.82532], icon=fl.Icon(color='green',
prefix='fa')).add_to(map_)
map_
```



Gambar 18 Output Visualisasi Peta

```
# Algoritma A-star
def aStarAlgo(start_node, stop_node):
    open_set = set(start_node)
    closed_set = set()
    g = {}                # Menyimpan jarak dari simpul awal
    parents = {}          # Variabel parents menyimpan pemetaan
ketetanggaan untuk semua simpul
    distance = {}         # Menyimpan jarak yang ditempuh untuk mencapai
setiap node
```

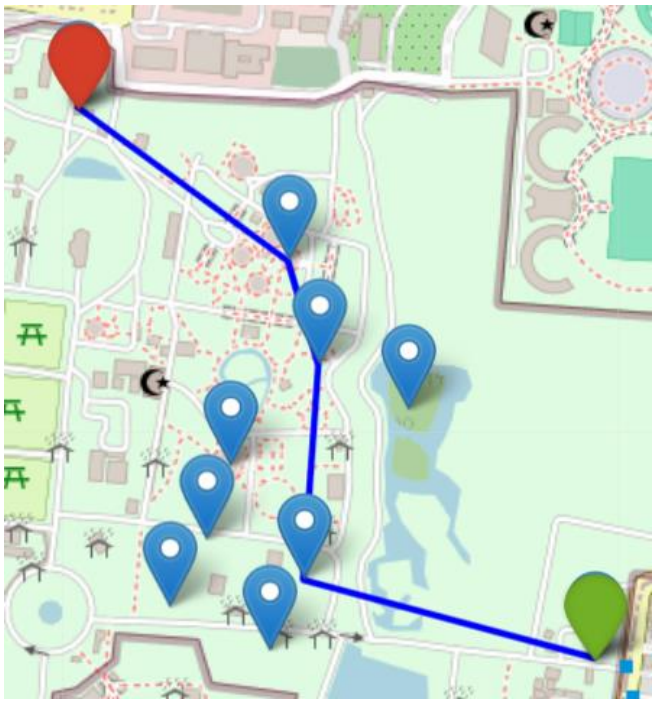
```
    #  Jarak simpul awal dari dirinya sendiri adalah nol
    g[start_node] = 0
    distance[start_node] = 0
    # start_node adalah simpul awal yaitu simpul yang tidak memiliki
simpul parents
    # Jadi, start_node ditetapkan sebagai simpul parents
    parents[start_node] = start_node
    while len(open_set) > 0:
        n = None
        # Simpul dengan nilai f() terkecil ditemukan
        for v in open_set:
            if n == None or g[v] + heuristic(v, stop_node) < g[n] +
heuristic(n, stop_node):
                n = v
        if n == stop_node or Graph_nodes[n] == None:
            pass
        else:
            for (m, weight) in get_neighbors(n):
                # Simpul-simpul 'm' yang tidak terdapat pada set
pertama dan terakhir, ditambahkan ke set pertama
                # Simpul n ditetapkan sebagai simpul parents
                if m not in open_set and m not in closed_set:
                    open_set.add(m)
                    parents[m] = n
                    g[m] = g[n] + weight
                    distance[m] = distance[n] + weight
                # Untuk setiap simpul m, bandingkan jaraknya dari
simpul awal
                # Dari simpul awal melewati simpul n
                else:
                    if g[m] > g[n] + weight:
                        # Perbarui g(m)
                        g[m] = g[n] + weight
                        # Perbarui jarak ke simpul m
                        distance[m] = distance[n] + weight
                        # Ganti parent dari m ke n
                        parents[m] = n
                        # Jika simpul m berada di set tertutup,
pindahkan ke set terbuka.
                        if m in closed_set:
                            closed_set.remove(m)
                            open_set.add(m)
        if n == None:
            print('Rute tidak ada!')
            return None

        # Jika simpul saat ini adalah simpul akhir
```

```
        # Maka kita mulai merekonstruksi rute dari simpul tersebut ke
simpul awal.
        if n == stop_node:
            path = []
            total_distance = distance[stop_node] + heuristic(stop_node,
stop_node)  # total distance from start to stop_node
            while parents[n] != n:
                path.append(n)
                n = parents[n]
            path.append(start_node)
            path.reverse()
            print('Rute ditemukan:', path)
            print('Jarak antara simpul asal dan tujuan:',
total_distance, 'meter')
            return path
        # Pindahkan simpul n dari open_list dan tambahkan ke
closed_list karena semua simpul tetangganya telah dicek.
        open_set.remove(n)
        closed_set.add(n)
    print('Rute tidak ada!')
    return None

# Mendefinisikan fungsi untuk mengembalikan tetangga dan jaraknya dari
simpul yang telah dilewati.
def get_neighbors(v):
    if v in Graph_nodes:
        return Graph_nodes[v]
    else:
        return None
# Fungsi ini mengembalikan jarak heuristik untuk semua node
def heuristic(n, stop_node):
  H_dist = {
      'A': 0.007435,
      'B': 0.004938,
      'C': 0.003976,
      'D': 0.003050,
      'E': 0.004066,
      'F': 0.002993,
      'G': 0.004002,
      'H': 0.003221,
      'I': 0.004228,
      'J': 0.006341,
      'K': 0
  }
  return H_dist[n]

# Penjelasan Graph
Graph_nodes = {
```

```
    'A': [('B', 350), ('J', 1100)],
    'B': [('C', 210)],
    'C': [('D', 110), ('E', 210), ('F', 290)],
    'D': [('C', 110), ('E', 190), ('F', 260), ('H', 350)],
    'E': [('C', 210), ('D', 190), ('F', 140), ('G', 110)],
    'F': [('C', 290), ('D', 260), ('E', 140), ('G', 110), ('H', 270),
('K', 500)],
    'G': [('E', 110), ('F', 110), ('I', 400), ('J', 750)],
    'H': [('D', 350), ('F', 270), ('I', 110), ('K', 350)],
    'I': [('G', 400), ('H', 110), ('J', 500)],
    'J': [('A', 1100), ('G', 750), ('I', 500)],
}

path = aStarAlgo('A', 'K')
```

```
Rute ditemukan: ['A', 'B', 'C', 'F', 'K']
Jarak antara simpul asal dan tujuan: 1350 meter
```

Gambar 19 Output Algoritma A*



Gambar 20 Output Visualisasi Rute Terpendek pada Peta

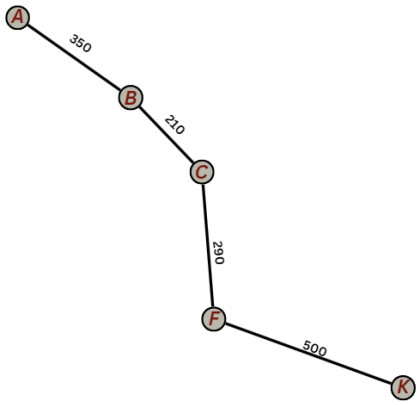
KESIMPULAN

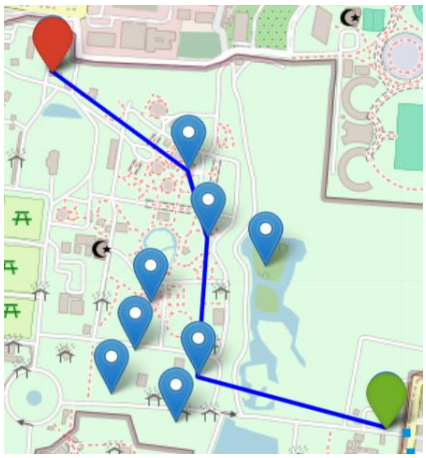
Berdasarkan hasil penelitian dan simulasi yang dilakukan menggunakan algoritma A*, dapat disimpulkan bahwa terdapat tujuh rute yang dapat digunakan untuk menempuh perjalanan dari titik A (Pintu Utara I) ke titik K (Pintu Timur) di Kebun Binatang Ragunan. Rute-rute tersebut adalah:

1. A-B-C-D-E-G-F-K dengan f(n) = 1580
2. A-B-C-E-G-F-K dengan f(n) = 1490
3. A-B-C-E-G-F-D-H-K dengan f(n) = 1950
4. A-B-C-F-K dengan f(n) = 1350
5. A-B-C-F-D-H-K dengan f(n) = 1810
6. A-B-C-F-H-K dengan f(n) =1470
7. A-B-C-E-F-K dengan f(n) = 1410

Dari perhitungan manual dan simulasi menggunakan Python, diperoleh kesimpulan bahwa rute terpendek yang optimal adalah rute A-B-C-F-K dengan f(n) = 1350. Rute ini melalui beberapa kandang di Kebun Binatang Ragunan, yaitu:

1. Kandang primata (B)
2. Kandang orang utan kalimantan (C)
3. Kandang orang utan (F)





Hasil simulasi yang konsisten antara perhitungan manual dan penggunaan algoritma A* menegaskan bahwa rute ini memang merupakan solusi optimal dalam pencarian rute terpendek dari Pintu Utara I ke Pintu Timur di Kebun Binatang Ragunan melalui Kadang Hewan Primata. Dengan demikian, penerapan algoritma A* terbukti efektif dalam mencari solusi optimal dalam konteks ini, yang dapat membantu meningkatkan efisiensi dan pengalaman navigasi pengunjung di Kebun Binatang Ragunan.

REFERENSI

Azwir, A., Jalaluddin, J., & Faisal, S. (2021). Observasi perilaku harian primata monyet ekor panjang (Mascaca fascicularis) berdasarkan etno ekologi di Kawasan Gunung Geurutee Kabupaten Aceh Jaya. *Jurnal Biology Education*, 9(1), 8-16.

Candra, A., Budiman, M. A., & Hartanto, K. Dijkstra's and A-Star in Finding the Shortest Path: a Tutorial. *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*. https://doi.org/10.1109/DATABIA50434.2020.9190342.

Hart, P. E., Nilsson, N. J, & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107. https://doi.org/10.1109/TSSC.1968.300136.

Pawestri, K. N., Wulandari, E., & Yulianti, E. (2021). Pengembangan Media Puzzle untuk Pembelajaran Biologi pada Materi Primata Siswa Kelas X SMA. *BIOCOLONY*, 4(2), 22-26.

Peta & Petunjuk Arah. *Taman Margasatwa Ragunan*. (n.d.). Diakses pada 15 April 2024. https://ragunanzoo.jakarta.go.id/info-pengunjung/map-direction-2/

Rachmawati, D., & Gustin, L. (2020). Analysis of Dijkstra's algorithm and A* algorithm in shortest path problem. *Journal of Physics: Conference Series*, 1566(1). https://doi.org/10.1088/1742-6596/1566/1/012061.

Sejarah Singkat. *Taman Margasatwa Ragunan*. (n.d.). Diakses pada 15 April 2024. https://ragunanzoo.jakarta.go.id/tentang/short-history/

Ragunan, T. M. [@ragunanzoo]. (2023). Akses Pintu Masuk Taman Margasatwa Ragunan. [Poster]. https://www.instagram.com/ragunanzoo/p/CrW9N-4yvDz/?img_index=1

Sugiyono. (2017). Metode Penelitian Kuantitatif, Kualitatif, dan R&D. Jakarta: Alfabeta.