TUGAS AKHIR – ES234849

OPTIMASI RUTE *UNMANNED AERIAL VEHICLE* DALAM PENANGANAN *STRESSED REGION* PADA *PRECISION AGRICULTURE* DENGAN MENGGUNAKAN ALGORITMA *SIMULATED ANNEALING*

NAUFAL MAKARIM ASCHAFITZ

NRP 5026211074

Dosen Pembimbing

Amalia Utamima, S.Kom., MBA., Ph.D.

NIP 198612132015042001

Program Studi Sarjana Sistem Informasi

Departemen Sistem Informasi

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2025

TUGAS AKHIR – ES234849

OPTIMASI RUTE *UNMANNED AERIAL VEHICLE* DALAM PENANGANAN *STRESSED REGION* PADA *PRECISION AGRICULTURE* DENGAN MENGGUNAKAN ALGORITMA *SIMULATED ANNEALING*

NAUFAL MAKARIM ASCHAFITZ

NRP 5026211074

Dosen Pembimbing

Amalia Utamima, S.Kom., MBA., Ph.D.

NIP 198612132015042001

Program Studi Sarjana Sistem Informasi

Departemen Sisten Informasi

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2025

FINAL PROJECT – ES234849

ROUTE OPTIMIZATION FOR UNMANNED AERIAL VEHICLE IN MANAGING STRESSED REGION IN PRECISION AGRICULTURE USING THE SIMULATED ANNEALING ALGORITHM

NAUFAL MAKARIM ASCHAFITZ

NRP 5026211074

Advisor

Amalia Utamima, S.Kom., MBA., Ph.D.

NIP 198612132015042001

Bachelor of Information System Program

Department of Information System

Faculty of Intelligent Electrical and Informatics Technology

Institut Teknologi Sepuluh Nopember

Surabaya

2025

LEMBAR PENGESAHAN

OPTIMASI RUTE UNMANNED AERIAL VEHICLE DALAM PENANGANAN STRESSED REGION PADA PRECISION AGRICULTURE DENGAN MENGGUNAKAN ALGORITMA SIMULATED ANNEALING

TUGAS AKHIR

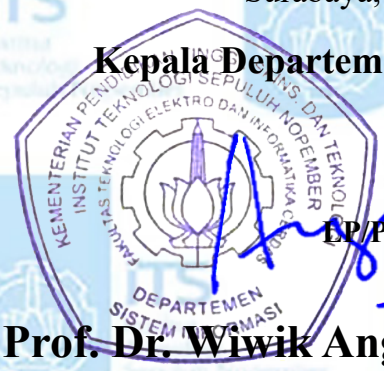
Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer (S.Kom)
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas (ELECTICS)
Institut Teknologi Sepuluh Nopember

Oleh :

Naufal Makarim Aschafitz
NRP: 5026211074

Surabaya, 24 Juli 2025

Kepala Departemen Sistem Informasi



LP/P/25/381

Prof. Dr. Wiwik Anggraeni, S.Si, M.Kom
NIP. 197601232001122002

LEMBAR PERSETUJUAN

OPTIMASI RUTE UNMANNED AERIAL VEHICLE DALAM PENANGANAN STRESSED REGION PADA PRECISION AGRICULTURE DENGAN MENGGUNAKAN ALGORITMA SIMULATED ANNEALING

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Oleh :

Naufal Makarim Aschafitz
NRP: 5026211074

Disetujui Tim Penguji: Tanggal Ujian: 09 Juli 2025
Periode Wisuda: September 2025

Amalia Utamima, S.Kom, MBA, Ph.D (Pembimbing 1)

Ahmad Mukhlason, S.Kom, M.Sc, Ph.D (Penguji 1)

Raras Tyasnurita, S.Kom, MBA (Penguji 2)

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP	:	Naufal Makarim Aschafitz / 5026211074
Program studi	:	S1 Sistem Informasi
Dosen Pembimbing / NIP	:	Amalia Utamima, S.Kom, MBA, Ph.D / 198612132015042001

dengan ini menyatakan bahwa Tugas Akhir dengan judul "OPTIMASI RUTE UNMANNED AERIAL VEHICLE DALAM PENANGANAN STRESSED REGION PADA PRECISION AGRICULTURE DENGAN MENGGUNAKAN ALGORITMA SIMULATED ANNEALING" adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 22 Juli 2025

Mengetahui

Dosen Pembimbing

Amalia Utamima, S.Kom, MBA, Ph.D
NIP. 198612132015042001

Mahasiswa

Naufal Makarim Aschafitz
NRP. 5026211074

ABSTRAK

OPTIMASI RUTE *UNMANNED AERIAL VEHICLE* DALAM PENANGANAN *STRESSED REGION* PADA *PRECISION AGRICULTURE* DENGAN MENGGUNAKAN ALGORITMA *SIMULATED ANNEALING*

Nama Mahasiswa / NRP	**:**	**Naufal Makarim Aschafitz / 5026211074**
Departemen	**:**	**Sistem Informasi FTEIC - ITS**
Dosen Pembimbing	**:**	**Amalia Utamima, S.Kom., MBA., Ph.D.**

Abstrak

Ketahanan pangan menjadi tantangan besar di tengah pertumbuhan penduduk Indonesia yang mencapai 281,6 juta jiwa pada 2024 dengan laju 1,25% per tahun. Sementara itu, luas lahan pertanian terus menurun akibat alih fungsi dan urbanisasi, sehingga memengaruhi kapasitas produksi pangan nasional. *Precision agriculture* hadir sebagai solusi teknologi yang meningkatkan produktivitas dan efisiensi penggunaan sumber daya, salah satunya dengan pemanfaatan *Unmanned Aerial Vehicle* (UAV) untuk pemetaan stressed region serta distribusi agrokimia yang lebih presisi. Namun, penelitian terkait optimasi rute UAV pada sektor ini masih terbatas, khususnya dalam penerapan klasterisasi dan algoritma metaheuristik secara terpadu. Tugas akhir ini mengembangkan metode optimasi rute UAV menggunakan *hybrid Simulated Annealing* (SA) yang menggabungkan inisialisasi greedy dan local search two-opt untuk meminimalkan jarak tempuh distribusi agrokimia. Tahap klasterisasi dilakukan menggunakan *Lloyd algorithm* yang divalidasi melalui *grid search* sehingga diperoleh *coverage ratio* sebesar 97.28% dan *overlap ratio* 19.96%, memastikan seluruh area terdampak terjangkau secara efisien. Pengujian pada data lahan pertanian menunjukkan *hybrid* SA mampu menghasilkan rute terpendek sejauh 1425.22 meter dengan estimasi waktu penerbangan 203.6 detik, hanya berselisih 0.02% dibanding *benchmark* Google OR-Tools. Hasil ini membuktikan bahwa *hybrid* SA merupakan pendekatan kompetitif sekaligus adaptif untuk optimasi rute UAV pada *precision agriculture*, dengan potensi besar diterapkan dalam praktik ditribusi agrokimia di lapangan.

Kata kunci: *Precision Agriculture, Stressed Region, Unmanned Aerial Vehicle, Simulated Annealing,* Optimasi Rute

ABSTRACT

ROUTE OPTIMIZATIPN FOR UNMANNED AERIAL VEHICLE IN MANAGING STRESSED REGION IN PRECISION AGRICULTURE USING THE SIMULATED ANNEALING ALGORITHM

Student Name / NRP	**:**	**Naufal Makarim Aschafitz / 5026211074**
Department	**:**	**Sistem Informasi FTEIC - ITS**
Advisor	**:**	**Amalia Utamima, S.Kom., MBA., Ph.D.**

Abstract

Food security poses a significant challenge amid Indonesia's growing population, which reached 281.6 million in 2024 with an annual growth rate of 1.25%. Meanwhile, the area of agricultural land continues to decline due to land conversion and urbanization, undermining the nation's food production capacity. Precision agriculture emerges as a technology-driven approach to enhance productivity and resource efficiency, including the use of Unmanned Aerial Vehicles (UAVs) for mapping stressed regions and precisely distributing agrochemicals. However, studies focusing on UAV route optimization in this sector remain scarce, especially those that integrate clustering methods with metaheuristic algorithms. This research develops a UAV route optimization approach by modeling agrochemical distribution as a Vehicle Routing Problem (VRP), solved using a hybrid Simulated Annealing (SA) algorithm that combines greedy initialization with two-opt local search to minimize total travel distance. The clustering stage employs Lloyd's algorithm validated by grid search, achieving a coverage ratio of 97.28% with an overlap of 19.96%, ensuring efficient spraying across affected areas. Tests on agricultural land data reveal that the hybrid SA produces a shortest route of 1,425.22 meters with an estimated flight time of 203.6 seconds, differing by only about 0.02% from the Google OR-Tools benchmark. These findings highlight the competitiveness and adaptability of the hybrid SA approach for UAV route optimization in precision agriculture applications.

Keywords: ***Precision Agriculture, Stressed Region, UAV, Simulated Annealing, Route Optimization***

KATA PENGANTAR

Segala puji dan rasa syukur penulis panjatkan kepada Allah SWT, karena berkat rahmat dan karunia-Nya, penulis dapat menyelesaikan tugas akhir yang berjudul *"*Optimasi Rute *Unmanned Aerial Vehicle* Dalam Penanganan *Stressed Region* Pada *Precision Agriculture* Dengan Menggunakan Algoritma *Simulated Annealing"* dengan baik dan tepat waktu. Tugas akhir ini dibuat untuk memenuhi salah satu syarat kelulusan Program Studi S1 Sistem Informasi Fakultas Teknologi Elektro dan Informatika Cerdas di Institut Teknologi Sepuluh Nopember. Penyusunan dan penyelesaian tugas akhir ini tentunya tidak terlepas dari dukungan, bantuan, serta kontribusi berbagai pihak, baik secara langsung maupun tidak langsung. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Keluarga penulis terutama orangtua penulis yang selalu memberikan dukungan dan motivasi dalam menyelesaikan tugas akhir ini.
2. Diri saya sendiri yang selalu berusaha secara maksimal dan selalu menjaga motivasi dari awal hingga penyelesaian tugas akhir.
3. Ibu Dr. Wiwik Anggraeni, S.Si., M.Kom. selaku Kepala Departemen Sistem Informasi ITS.
4. Ibu Amalia Utamima, S.Kom., MBA., Ph.D selaku dosen pembimbing penulis yang telah bersedia membimbing dan terus memberikan masukan kepada penulis sehingga penulis dapat menyelesaikan tugas khir dengan baik.
5. Bapak Ahmad Mukhlason, S.Kom., M.Sc., Ph.D. dan Ibu Raras Tyasnurita, S.Kom., M.BA., Ph.D. selaku dosen penguji yang memberikan saran dan kritik untuk menyempurnakan pengerjaan tugas akhir.
6. Seluruh dosen Departemen Sistem Informasi ITS yang telah memberikan ilmu kepada penulis semasa perkuliahan sehingga penulis dapat menyusun dan menyelesaikan tugas akhir.
7. Daffa Rheza Prayoga selaku rekan pengerjaan tugas akhir yang membantu dalam menyelesaikan tugas akhir.
8. Orang tua dan seluruh keluarga besar penulis yang selalu memberikan dukungan moral, doa, dan semangat yang tiada henti dari awal hingga akhir proses pengerjaan tugas akhir ini.
9. Teman-teman dekat penulis yang selalu memberikan dukungan dan selalu ada sepanjang masa perkuliahan hingga selesainya tugas akhir ini.

Surabaya, 8 Juli 2025

Naufal Makarim Aschafitz

DAFTAR ISI

LEMBAR PENGESAHAN i
LEMBAR PERSETUJUAN ii
PERNYATAAN ORISINALITAS iii
ABSTRAK iv
ABSTRACT v
KATA PENGANTAR vi
DAFTAR ISI vii
DAFTAR GAMBAR x
DAFTAR TABEL xi
DAFTAR KODE xii
BAB 1 PENDAHULUAN 13
1.1 Latar Belakang 13
1.2 Rumusan Masalah 14
1.3 Batasan Masalah 14
1.4 Tujuan 15
1.5 Manfaat 15
1.6 Relevansi 15
BAB 2 TINJAUAN PUSTAKA 17
2.1 Hasil Penelitian Terdahulu 17
2.2 Dasar Teori 19
2.2.1 Precision Agriculture 19
2.2.2 Unmanned Aerial Vehicle 20
2.2.3 Stressed Region 21
2.2.4 Lloyd Algorithm 21
2.2.5 Diagram Voronoi 22
2.2.6 *Vehicle Route Planning* 23
2.2.7 Two-opt Local Search 25
2.2.8 Simulated Annealing 26
2.2.9 Evaluasi Hasil 27
BAB 3 METODOLOGI 29
3.1 Metode yang digunakan 29
3.2 Uraian Metodologi 29

3.2.1 Studi Literatur dan Identifikasi Masalah 29
3.2.2 Pengumpulan dan Pengolahan Data 30
3.2.3 Pembentukan Klasterisasi dengan *Lloyd Algorithm* 31
3.2.4 Implementasi Metode Algoritma 32
3.2.5 Pengujian dan Evaluasi 32
3.2.6 Analisis dan Pembahasan 32
3.2.7 Penyusunan Buku Tugas Akhir 32
BAB 4 HASIL DAN PEMBAHASAN 34
4.1 Deskripsi Data 34
4.2 Environment 36
4.3 Implementasi 37
4.3.1 Pengolahan Data 37
4.3.1.1 Pengumpulan Data 37
4.3.1.2 Segmentasi Data dan Pembentukan Poligon Tertutup 37
4.3.2 Klasterisasi Data 38
4.3.2.1 Pembuatan Titik Acak dalam Wilayah (Region-based Sampling) 38
4.3.2.2 Proses Grid Search Evaluasi Overlap-Coverage 39
4.3.2.3 Proses Klasterisasi Hasil Grid Search 41
4.3.2.4 Visualisasi Hasil Klasterisasi 42
4.3.3 Desain Algoritma 43
4.3.3.1 Pembentukan Distance Matrix dan Fungsi Evaluasi 43
4.3.3.2 Greedy Initial Route 44
4.3.3.3 Two-Opt Method 44
4.3.3.4 Simulated Annealing 45
4.3.4 Desain Algoritma Pembanding (Google OR-Tools) 47
4.4 Hasil Uji Coba dan Pembahasan 48
4.4.1 Klasterisasi Data 48
4.4.2 Optimasi Rute UAV 52
4.4.2.1 Uji Coba Awal Algoritma Hybrid Simulated Annealing 53
4.4.2.2 Grid Search dan Multi-Run pada Parameter Terbaik 54
4.4.2.3 Visualisasi Rute Terbaik dari Seluruh Percobaan 56
4.4.2.4 Uji Coba Algoritma dengan OR-Tools 57
4.4.2.5 Uji Coba Algoritma dengan Ant Colony Optimization (ACO) 58
4.4.3 Perbandingan Kinerja Algoritma 60

4.4.3.1 Perbandingan Hybrid SA dengan OR-Tools 61
4.4.3.2 Perbandingan Hybrid SA dengan ACO 63
BAB 5 Kesimpulan dan Saran 67
5.1 Kesimpulan 67
5.2 Saran 68
DAFTAR PUSTAKA 69
LAMPIRAN 72
BIODATA PENULIS 81

DAFTAR GAMBAR

Gambar 1.1 Kerangka Kerja Riset Laboratorium RDIB 16
Gambar 2.1 Ilustrasi Konsep Vehicle Routing Problem 23
Gambar 2.2 Contoh hasil klasterisasi dengan lloyd algorithm 22
Gambar 3.1 Diagram Alir Metodologi 29
Gambar 3.2 Gambar lahan pertanian (data primer) 30
Gambar 3.3 Gambar data stressed region (data sekunder) 30
Gambar 3.4 Hasil penggabungan data 31
Gambar 4.1 Hasil Visualisasi Klasterisasi Menggunakan Lloyd Algorithm 49
Gambar 4.2 Visualisasi Distribusi Coverage dan Overlap pada Stressed Region 51
Gambar 4.3 Visualisasi Jalur UAV pada Rute Terpendek oleh Hybrid SA 56
Gambar 4.4 Grafik Konvergensi Nilai Cost pada Run Terbaik 57
Gambar 4.5 Hasil Visualisasi Rute UAV Terpendek Hasil OR-Tools 58
Gambar 4.6 Visualisasi Rute Terbaik Hasil Hybrid SA 62
Gambar 4.7 Visualisasi Rute Terbaik Hasil OR-Tools 62

DAFTAR TABEL

Tabel 2.1 Literatur 1 (Srivastava et al., 2019) 17
Tabel 2.2 Literatur 2 (Furchi et al., 2023)..... 17
Tabel 2.3 Literatur 3(Utamima et al., 2019) 18
Tabel 2.4 Literatur 4 (Conesa-Muñoz et al., 2016)..... 18
Tabel 2.5 Literatur 5 (Ribeiro Junior et al., 2022) 19
Tabel 2.6 Spesifikasi UAV 20
Tabel 4.1 Data Koordinat Titik pada Lahan Pertanian 34
Tabel 4.2 Perangkat Keras Yang Digunakan..... 36
Tabel 4.3 Perangkat Lunak Dan Pustaka Python yang Digunakan 36
Tabel 4.4 Jumlah Klaster dan Subklaster per-Stressed Region 48
Tabel 4.5 Perhitungan Luas Persentase Coverage dan Overlap Area Tiap Stressed Region . 50
Tabel 4.6 Nilai Persentase Coverage dan Overlap Ratio Secara Global 50
Tabel 4.7 Variasi Parameter Algoritma Hybrid SImulated Annealing..... 53
Tabel 4.8 Rangkuman Hasil Uji Coba Hybrid SA Pervariasi Parameter 53
Tabel 4.9 Rata-rata Jarak Tempuh Berdasarkan Setiap Nilai Parameter..... 54
Tabel 4.10 Tiga Konfigurasi Parameter Terbaik Hasil Grid Search untuk Uji Multi-run..... 55
Tabel 4.11 Hasil Rangkuman Hasil Uji Coba pada Tiga Konfigurasi Parameter Terbaik..... 55
Tabel 4.12 Hasil Optimasi Terbaik Algoritma Hybrid Simulated Annealing 56
Tabel 4.13 Rekap Statistik Hasil Optimasi OR-Tools..... 57
Tabel 4.14 Hasil Optimasi Rute Terbaik OR-Tools 58
Tabel 4.15 Parameter pada Algoritma ACO..... 59
Tabel 4.16 Rekap Statistik Hasil Optimasi ACO..... 59
Tabel 4.17 Hasil Optimasi Rute Terbaik ACO..... 60
Tabel 4.18 Parameter dan Hasil Terbaik Algoritma Hybrid SA..... 61
Tabel 4.19 Perbandingan Hasil Kinerja Terbaik Algoritma dalam Optimasi Rute UAV 63
Tabel 4.20 Perbandingan Hasil Kinerja Hybrid SA dan ACO dalam Optimasi Rute 64

DAFTAR KODE

Kode 4.1 Import Data dari CSV 37
Kode 4.2 Segmentasi Data dan Pembentukan Poligon Tertutup 38
Kode 4.3 Fungsi Untuk Menghasilkan Titik Acak Di Dalam Batas Poligon 38
Kode 4.4 Menghasilkan Titik Acak dalam Masing-masing Inner Region 39
Kode 4.5 Fungsi Utama untuk Klasterisasi, Perhitungan Coverage, dan Overlap UAV 40
Kode 4.6 Grid Search Evaluasi Klasterisasi Stressed Region 41
Kode 4.7 Kode Menjalankan Klasterisasi 42
Kode 4.8 Visualisasi Klaster dan Penympanan Hasil Centroid 43
Kode 4.9 Pembuatan Jarak Matriks Euclidean dan fungsi fitness untuk TSP 43
Kode 4.10 Pembuatan Rute Awal dengan Greedy Nearest Neighbor 44
Kode 4.11 Fungsi Algoritma Two-opt 45
Kode 4.12 Implementasi Algoritma Hybrid Simulated Annealing 47
Kode 4.13 Fungsi Algoritma OR-Tools 48

BAB 1 PENDAHULUAN

Di dalam bab pendahuluan, akan dipaparkan penjelasan awal terkait latar belakang masalah, perumusan masalah, tujuan, serta manfaat dari tugas akhir ini.

1.1 Latar Belakang

Indonesia merupakan salah satu negara dengan jumlah penduduk terbesar di dunia. Menurut Badan Pusat Statistik, jumlah Warga Indonesia mencapai 281,6 juta jiwa pada tahun 2024 dengan tingkat pertumbuhan populasi sebesar 1,11% setiap tahunnya (BPS, 2024). Hal ini menunjukkan adanya peningkatan jumlah penduduk sekitar 2–3 juta jiwa per tahun. Seiring dengan meningkatnya jumlah penduduk, kebutuhan pangan juga pasti mengalami peningkatan signifikan. Hal tersebut tentu menjadi tantangan bagi pemerintah dalam memastikan ketersediaan pangan yang mencukupi bagi seluruh penduduk. Permasalahan tersebut juga didukung dengan adanya penurunan jumlah lahan pertanian secara terus menerus di penjuru daerah dan berdampak terhadap keseimbangan kebutuhan dan suplai pangan negara yang tersedia. Oleh karena itu, diperlukan suatu solusi untuk mengatasi masalah keseimbangan pangan secara berkelanjutan dalam rangka memenuhi kebutuhan pangan nasional secara efektif dalam jangka panjang (Widiastuti et al., 2024).

Sustainable agriculture merupakan salah satu permasalahan utama di sektor pertanian global saat ini yang berfokus terhadap cara kinerja sektor pertanian dalam memenuhi kebutuhan pangan saat ini tanpa mengorbankan kemampuan generasi mendatang. Konsep tersebut tidak hanya sekadar memenuhi kebutuhan pangan bagi para warga saja, tetapi juga menekankan pada pelestarian serta peningkatan kualitas sumber daya yang menjadi tumpuan sistem pertanian negara dalam jangka panjang. Di Indonesia, terdapat beberapa faktor yang menghambat pencapaian *sustainable agriculture*. Beberapa di antaranya adalah kurangnya efektivitas pemeliharaan lahan pertanian, mulai dari serangan hama yang tidak terkendali, kurangnya penggunaan pupuk, pasokan air yang terbatas, serta ketidakpastian akibat perubahan iklim (Srivastava et al., 2019). Selain itu, penurunan drastis jumlah petani dan kurangnya minat pada profesi tersebut di kalangan anak-anak muda juga menjadi kendala serius yang menghambat tercapainya tujuan tersebut. Dalam menghadapi semua permasalahan tersebut, konsep *sustainable agriculture* dirancang agar sektor pertanian tidak hanya berfokus pada hasil produksi saja, tetapi juga pada pemberdayaan kehidupan para petani pemeliharaan lahan secara berkelanjutan panjang (Muhie, 2022).

Upaya-upaya dalam mengatasi tantangan *sustainable agriculture* telah dilakukan melalui berbagai inovasi teknologi. Salah satu inovasi yang telah diterapkan adalah penggunaan UAV (Unmanned Aerial Vehicle) dalam menerapkan *precision agriculture* pada aktivitas penebaran agrokimia di kegiatan sehari-hari. Dalam konteks tersebut, UAV ini merujuk pada sebuah perangkat udara tanpa awak seperti drone. UAV digunakan untuk berbagai tujuan dalam sektor pertanian, termasuk pemetaan *stressed region*, penyiraman pupuk, dan aplikasi pestisida. Dalam hal pemetaan *stressed region*, UAV melakukan penangkapan citra lahan pertanian dan mendeteksi tingkat kesehatan tanaman melalui tingkat stress tanaman, kadar air, kondisi nutrisi tanaman, dan lainnya. teknologi ini terbukti dapat meningkatkan presisi serta mengurangi waktu dan tenaga yang dibutuhkan. Dengan menggunakan UAV, pestisida dapat ditebarkan lebih tepat sasaran, terutama pada tanaman yang lebih membutuhkan perawatan intensif sehingga meminimalkan penggunaan bahan kimia dan efisiensi tenaga (Srivastava et al., 2019).

Pada penelitian sebelumnya, telah dikembangkan sebuah solusi untuk melakukan optimasi rute dalam sebuah lahan pertanian menggunakan metode *Lloyd Clustering.* Metode *Lloyd Clustering* bekerja dengan mengelompokkan data berdasarkan kesamaan tertentu melalui proses iteratif yang memindahkan titik centroid (pusat klaster) hingga posisi efisien tercapai. Dalam penelitian tersebut, teknik klasterisasi akan melakukan pemetaan terhadap data *stressed region* yang telah dibuat sebelumnya. Kemudian, hasil klasterisasi akan dilakukan optimasi rute dengan memanfaatkan titik-titik *subcentroid* sebagai referensi dalam menentukan rute intervensi yang efisien. Dengan fokus pada area yang memiliki tingkat stres lebih tinggi, penggunaan sumber daya dapat dikelola lebih efisien dan tepat sasaran, sekaligus mengurangi biaya yang diperlukan.

Meskipun penelitian mengenai permasalahan optimasi rute pada pertanian sudah banyak beredar, tetapi belum ada penelitian yang membahas secara spesifik mengenai optimasi rute dalam penanganan area *stressed region*. Penelitian mengenai *stressed region* yang sudah ada hanya membahas pemetaan area *stressed region* saja, tidak mencakup optimasi rutenya. Dengan kata lain, studi penelitian yang membahas topik tugas akhir ini masih terbatas. Melalui pendekatan tersebut, tugas akhir ini bertujuan dalam menjembatani celah antara teknologi pemetaan *stressed region* dan kebutuhan solusi logistik pada UAV dalam konteks *precision agriculture*.

Berdasarkan hal-hal tersebut, tugas akhir ini bertujuan untuk mengimplementasikan algoritma lain dalam permasalahan optimasi rute UAV. Dengan begitu pengerjaan tugas akhir ini diharapkan dapat menjadi referensi pembanding dengan metode Solusi yang sudah ada. Pengembangan ini diharapkan dapat memberikan kontribusi terhadap pencapaian *sustainable agriculture* sehingga dapat mendukung ketahanan pangan dengan memaksimalkan penggunaan teknologi dalam sektor pertanian.

1.2 Rumusan Masalah

Berdasarkan latar belakang permasalahan di atas, maka rumusan permasalahan dalam tugas akhir ini adalah sebagai berikut

1. Bagaimana menentukan titik penyemprotan agrokimia yang harus dilalui UAV dengan menggunakan *Lloyd algorithm*
2. Bagaimana membangun algoritma *simulated annealing* untuk diterapkan dalam perencanaan rute UAV dalam penanganan *stressed region* secara efisien

1.3 Batasan Masalah

Untuk memfokuskan pengerjaan tugas akhir ini agar lebih terarah dan mendalam, beberapa batasan masalah yang diterapkan adalah sebagai berikut

1. Cakupan dari tugas akhir hanya akan berfokus pada pengembangan metode perencanaan rute UAV untuk mendukung *precision agriculture*
2. Tugas akhir ini hanya mencakup penebaran agrokimia pada lahan pertanian berbasis pemetaan *stressed region*, dan tidak mencakup penggunaan UAV untuk fungsi lain di sektor pertanian
3. Data yang digunakan, yaitu data ladang pertanian dari data suatu daerah petak asli dan data stressed region sebagai data sekunder yang diambil dari paper rujukan

4. Tugas akhir ini tidak mempertimbangkan kondisi cuaca, arah angin, serta topologi lahan yang melibatkan tanjakan atau elevasi. Simulasi dilakukan pada ruang dua dimensi untuk menyederhanakan proses optimasi rute

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut

1. Mengembangkan metode *lloyd algorithm* untuk menentukan titik-titik efisien penyemprotan agrokimia yang harus dilalui UAV pada area *stressed region* di ladang pertanian
2. Merancang algoritma *Simulated annealing* untuk merencanakan rute UAV dalam menangani *stressed region* secara efisien

1.5 Manfaat

Tugas akhir ini diharapkan memberikan manfaat sebagai berikut

1. Memberikan kontribusi dalam meningkatkan produktivitas dalam hal perencanaan rute UAV di sektor pertanian melalui pengoptimalan jarak tempuh UAV pada suatu lahan menggunakan Algoritma *Simulated Annealing*
2. Menjadi referensi ataupun landasan untuk pengembangan inovasi teknologi UAV dan *precision agriculture* dalam merumuskan solusi dalam permasalahan rute pemupukan dan jarak tempuh melalui Algoritma *Simulated Annealing*

1.6 Relevansi

Tugas akhir ini memiliki relevansi yang erat dengan fokus dalam Laboratorium Rekayasa Data dan Intelijensi Biisnis (RDIB), khususnya dalam domain *ICT for rural development*. Dalam *roadmap* penelitian laboratorium, topik yang diambil pada tugas akhir ini, yaitu optimasi rute kendaraan pada pertanian sehingga hal tersebut tergolong dalam kategori *agriculuture route planning*. Pengerjaan ini memperluas penerapan perencanaan rute dengan menggunakan pendekatan algoritma metaheuristik, seperti *Simulated Annealing* dalam menerapkan penerapan sistem pertanian presisi. Selain itu, tugas akhir ini juga mendukung salah satu tujuan laboratorium dalam mengaplikasikan teknologi cerdas untuk memecahkan permasalahan praktis di berbagai sektor, termasuk pertanian. Dengan fokus pada efisiensi penebaran agrokimia, tugas akhir ini diharapkan dapat memberikan kontribusi terhadap Solusi yang lebih efisien dalam mendukung solusi *sustainable agriculture.*

Gambar 1.1 Kerangka Kerja Riset Laboratorium RDIB

BAB 2 TINJAUAN PUSTAKA

Dalam bab ini, terdapat kajian referensi dari penelitian-penelitian sebelumnya, dimana *paper-paper* tersebut digunakan sebagai rujukan atau referensi dalam tugas akhir. Selain itu, dipaparkan juga dasar teori yang relevan untuk mendukung isi dari pengerjaan tugas akhir.

2.1 Hasil Penelitian Terdahulu

Terdapat beberapa penelitian terdahulu yang memiliki keterkaitan dengan tugas akhir ini, diantaranya akan dijelaskan pada tabel beirkut ini

Tabel 2.1 Literatur 1 (Srivastava et al., 2019)

Judul	*UAVs technology for the development of GUI based application for precision agriculture and environmental research*
Nama, Tahun	Srivastava K, dkk 2019
Gambaran umum penelitian	Penelitian ini mengembangkan aplikasi GUI berbasis teknologi UAV untuk mendeteksi dan memetakan area tanaman yang mengalami stres. Proses penelitian diawali dengan pengumpulan data lapangan melalui UAV yang dilengkapi sensor khusus untuk menangkap gambar multispektral. Penelitian ini melalui beberapa tahap, mulai dari desain aplikasi, pengujian lapangan untuk kalibrasi data, hingga validasi hasil dengan membandingkan kondisi lapangan nyata. Hasil penelitian menunjukkan bahwa aplikasi GUI ini dapat membantu petani memahami kondisi lahan secara visual, mempermudah keputusan manajemen lahan berdasarkan data yang ditangkap oleh UAV.
Keterkaitan penelitian	Penelitian ini berkaitan erat dengan tugas akhir dalam menggunakan teknologi UAV untuk pemetaan area stres pada tanaman. Namun, pada tugas akhir ini, terdapat sebuah penambahan aspek optimasi rute UAV dengan algoritma *Simulated Annealing* untuk meningkatkan efisiensi distribusi agrokimia yang lebih presisi. Selain itu, artikel penelitian ini juga digunakan sebagai *baseline* pengerjaan tugas akhir mengenai optimasi rute UAV.

Tabel 2.2 Literatur 2 (Furchi et al., 2023)

Judul	*Route Optimization in Precision Agriculture Settings: A Multi-Steiner TSP Formulation*
Nama, Tahun	Furchi A, dkk 2023
Gambaran umum penelitian	Penelitian ini mengembangkan strategi perencanaan rute untuk robot mobile heterogen dalam pengaturan *Precision Agriculture* (PA). Dengan menggunakan formulasi *multi-Steiner Traveling Salesman Problem* (TSP), penelitian ini mengoptimalkan penugasan tugas pertanian pada robot, memperhitungkan waktu, energi, dan biaya manuver.
Keterkaitan penelitian	Penelitian ini berkaitan erat dengan tugas akhir mahasiswa mengenai optimasi rute dalam aplikasi pertanian presisi. Perbedaannya adalah penelitian ini lebih berfokus pada optimasi penggunaan beberapa robot dengan berbagai keterbatasan energi dan kebutuhan manuver, sedangkan dalam tugas akhir lebih difokuskan pada UAV untuk distribusi penebaran bahan agrokimia.

Tabel 2.3 Literatur 3(Utamima et al., 2019)

Judul	*Optimisation of Agricultural Routing Planning in Field Logistics with Evolutionary Hybrid Neighbourhood Search*
Nama, Tahun	Amalia Utamima, dkk 2019
Gambaran umum penelitian	Penelitian ini berisi optimalisasi perencanaan rute logistik di bidang pertanian dengan menggunakan *Evolutionary Hybrid Neighbourhood Search* (EHNS). Tujuan dari perancangan algoritma adalah untuk meminimalisir total jarak tempuh kendaraan otonom yang digunakan untuk distribusi sumber daya, seperti herbisida dan pupuk di area pertanian yang luas. Penelitian juga membandingkan kinerja EHNS dengan algoritma lain seperti *Genetic Algorithm* (GA) dan *Tabu Search* (TS). Hasil penelitian menunjukkan bahwa EHNS mampu memberikan solusi rute yang lebih efisien dengan waktu komputasi yang lebih singkat dibandingkan metode konvensional.
Keterkaitan penelitian	Penelitian ini relevan dengan tugas akhir karena sama-sama membahas optimasi rute di sektor pertanian. Namun, penelitian ini lebih berfokus pada armada kendaraan pertanian, sementara tugas akhir ini berfokus pada optimasi rute UAV untuk distribusi agrokimia di area *stressed region*. Selain itu, data lahan yang digunakan dalam penelitian ini juga menjadi bahan acuan sebagai data primer dalam tugas akhir.

Tabel 2.4 Literatur 4 (Conesa-Muñoz et al., 2016)

Judul	*Route planning for agricultural tasks: A general approach for fleets of autonomous vehicles in site-specific herbicide applications*
Nama, Tahun	Conesa-Muñoz J, dkk 2016
Gambaran umum penelitian	Penelitian ini mengembangkan metode optimasi rute untuk beberapa kendaraan otonom dalam aplikasi herbisida pada daerah lahan pertanian spesifik. Proses ini melibatkan penggunaan algoritma metaheuristik, yaitu *Simulated Annealing* (SA) dalam mengatasi masalah optimasi rute yang kompleks dengan mempertimbangkan faktor seperti biaya input, waktu, dan karakteristik unik kendaraan (kecepatan, radius putar, kapasitas tangki). Metode ini melibatkan perumusan masalah optimasi kombinatorial dan penentuan urutan trek optimal di lahan, termasuk pengaturan titik pengisian ulang. Algoritma SA mengontrol penurunan parameter suhu untuk menghindari perangkap minimum lokal, yang memungkinkan solusi optimal dengan mempertimbangkan variabel-variabel lapangan dan batasan kendaraan. Hasil dari metode ini dibandingkan dengan pendekatan lain, menunjukkan peningkatan dalam penghematan jarak dan efisiensi waktu operasional kendaraan di lahan yang ditentukan.
Keterkaitan penelitian	Penelitian ini memiliki relevansi terhadap tugas akhir dimana keduanya sama-sama menggunakan algoritma *Simulated Annealing* untuk optimasi rute di sektor pertanian. Namun, dalam penelitian ini, metode SA diterapkan pada armada kendaraan dengan karakteristik yang bervariasi, sedangkan penelitian ini lebih berfokus pada penggunaan UAV tunggal untuk distribusi agrokimia di lahan yang memiliki area stres tertentu.

	Kedua penelitian ini berkontribusi pada optimasi operasional di pertanian presisi.

Tabel 2.5 Literatur 5 (Ribeiro Junior et al., 2022)

Judul	*Data Reduction Based on Machine Learning Algorithms for Fog Computing in IoT Smart Agriculture*
Nama, Tahun	Ribeiro Junior F, dkk 2022
Gambaran umum penelitian	Penelitian ini berfokus pada optimasi data yang dikumpulkan melalui sensor IoT dalam sistem *smart agriculture* (*fog computing*). Data diklasifikasikan dengan algoritma *unsupervised learning*, mulai dari *K-Means*, *Hierarchical Tree*, dan *Gaussian Mixture Model*. Lalu, dilakukan kompresi data sehingga mengurangi volume data yang dikirim dari *fog node* ke *cloud*.
Keterkaitan penelitian	Penelitian memiliki relevansi dengan tugas akhir dimana keduanya menggunakan pengelompokkan klasterisasi menggunakan *K-Means Clustering* atau dapat disebut *Lloyd algorithm*.

2.2 Dasar Teori

Subbab ini berisikan paparan hingga penjelasan dasar teori yang akan digunakan dalam tugas akhir dan mendukung isi dari tugas akhir tersebut.

2.2.1 Precision Agriculture

Precision Agriculture merupakan salah satu strategi manajemen pertanian berbasis teknologi informasi untuk mengumpulkan dan menganalisis data dari berbagai sumber dengan tujuan mendukung pengambilan keputusan yang lebih baik mengenai startegi produksi pertanian. Strategi ini memungkinkan petani untuk mengetahui kondisi spasial dan temporal dalam suatu lahan, dengan begitu petani dapat melakukan tindakan yang berbeda-beda untuk setiap area spesifik yang diperlukan. Implementasi tersebut dapat meningkatkan efisiensi penggunaan sumber daya, seperti air, pupuk, dan pestisida, serta meminimalkan limbah dan dampak negatif terhadap lingkungan sehingga menghasilkan hasil produksi yang lebih baik (Cisternas et al., 2020).

Dalam *Precision Agriculture*, teknologi seperti sensor, sistem navigasi satelit (*Global Positioning System* - GPS), drone atau *Unmanned Aerial Vehicles* (UAV), serta *Internet of Things* (IoT) digunakan untuk mengumpulkan informasi detail terkait kondisi lahan, tanaman, dan cuaca. Penerapan teknologi-teknologi tersebut memberikan solusi inovatif dalam menghadapi tantangan dalam sektor pertanian, seperti perubahan iklim, keterbatasan lahan, dan peningkatan kebutuhan pangan. Dengan teknologi seperti prediksi cuaca, pemantauan kelembaban tanah, dan deteksi hama, *precision agriculture* memungkinkan peningkatan efisiensi dan keberlanjutan produksi. Misalnya, penggunaan UAV untuk pemetaan lahan dan pemantauan tanaman, serta aplikasi kecerdasan buatan untuk mengidentifikasi penyakit dan gulma, telah terbukti mampu meningkatkan produktivitas dan mengurangi dampak lingkungan yang merugikan (Sharma et al., 2021).

Implementasi precision agriculture merupakan salah satu Solusi dalam mencapai tujuan *sustainable agriculture*. Pengelolaan lahan dengan lebih presisi juga mengartikan bahwa setiap area lahan dapat dioptimalkan sesuai kebutuhannya, baik dari segi nutrisi, air, maupun

perlindungan dari hama dan penyakit. Precision agriculture juga memungkinkan deteksi lebih dini terhadap ancaman seperti penyakit tanaman, kekurangan nutrisi, atau serangan hama, sehingga intervensi dapat dilakukan lebih cepat dan tepat sebelum masalah semakin memburuk. Dengan mengurangi penggunaan input yang tidak perlu, precision agriculture juga berkontribusi pada pelestarian kesehatan tanah, menjaga biodiversitas, serta mengurangi jejak karbon pertanian, menjadikannya salah satu solusi kunci untuk pertanian berkelanjutan di masa depan (Srivastava et al., 2019).

2.2.2 Unmanned Aerial Vehicle

Secara umum, *Unmanned Aerial Vehicle* (UAV) adalah sebuah kendaraan udara tanpa awak yang beroperasi secara otonom atau melalui kendali jarak jauh dari sebuah *control station.* UAV digunakan untuk berbagai hal, mulai dari pemantauan, perencanaan rute, pencarian dan penyelamatan, serta inspeksi infrastruktur. Semenjak penemuannya, UAV telah berkembang secara pesat, baik dalam hal struktur, metode kerja, fitur penerbangan, maupun kontrol navigasi. Awalnya, UAV dikembangkan untuk berbagai keperluan militer dan sipil. Namun, penggunaannya kini telah meluas ke berbagai sektor termasuk pertanian, transportasi, dan pengelolaan bencana. Dengan kemampuan manuver yang fleksibel dan portabilitas tinggi, UAV juga dapat digunakan untuk mendeteksi kerusakan bangunan, memantau proyek konstruksi, hingga mengantarkan material ke lokasi yang sulit dijangkau (Ahmed et al., 2022).

Dalam pertanian, UAV telah merevolusi praktik-praktik konvensional yang telah ada melalui *precision agriculture.* Alat ini memungkinkan pemantauan lahan secara *real time* dan penggunaan sumber daya secara efisien dengan menggabungkan teknologi pencitraan dan kecerdasan buatan (Nazeer et al., 2024). Selain itu, integrasi sistem dalam UAV juga memungkinkan UAV dalam mendeteksi area tanaman yang stres untuk mengidentifikasi area tanaman yang memerlukan intervensi khusus. Dengan begitu, para petani dapat mengambil tindakan atau hasil keputusan yang tepat demi meningkatkan produktivitas hasil panen dan keberlanjutan lahan pertanian (Srivastava et al., 2019).

Pada implementasi dalam tugas akhir, model UAV yang digunakan adalah Dji Agras T-30 dari DJI. Drone ini dirancang khusus untuk aplikasi pertanian pada lahan berskala besar dan digunakan dalam berbagai pekerjaan, termasuk penyemprotan bahan agrokimia. Berikut rincian spesifikasi yang relevan untuk penelitian ini.

Tabel 2.6 Spesifikasi UAV

Fitur	Spesifikasi
Kapasitas tangki	40 liter
Waktu terbang efektif	20 menit
Laju semprot maksimum	8 liter/menit
Jumlah nozzle	16
Lebar penyemprotan	9 meter
Kecepatan terbang rata-rata	7 meter/detik

Berdasarkan, spesifikasi tersebut, dengan waktu terbang efektif sekitar 20 menit dan kecepatan rata-rata 7 m/s, Uav dapat menempuh llintasan sepanjang 8400 meter dalam satu kali penerbangan. Dari sisi kapasitas tangki dan laju semprotan, dengan kapasitas sebesar 40 liter dan laju semprot sebesar 8 L/menit, maka UAV akan menghabiskan satu tangki larutan dalam waktu sekitar **5 menit**. Dalam waktu tersebut, didapatkan total lintasan yang dapat ditempuh

berdasarkan kecepatan 7 m/s mencapai **2100 meter** sehingga area yang dapat disemprot pada satu kali jalan mencapai **18900 m^2**. Nilai ini menunjukkan bahwa dalam satu siklus semprot, *drone* ini mampu menyelesaikan penyemprotan pada area yang bahkan melebihi luas area *stressed region* sebesar 16289 m^2 sehingga mendukung model optimasi VRP pada tugas akhir ini, dari segi kapasitas baterai dan tangki untuk difokuskan pada perencanaan lintasan tunggal (*single tour*) tanpa perlu mempertimbangkan skenario *multi trip* atau *multi vehicle*.

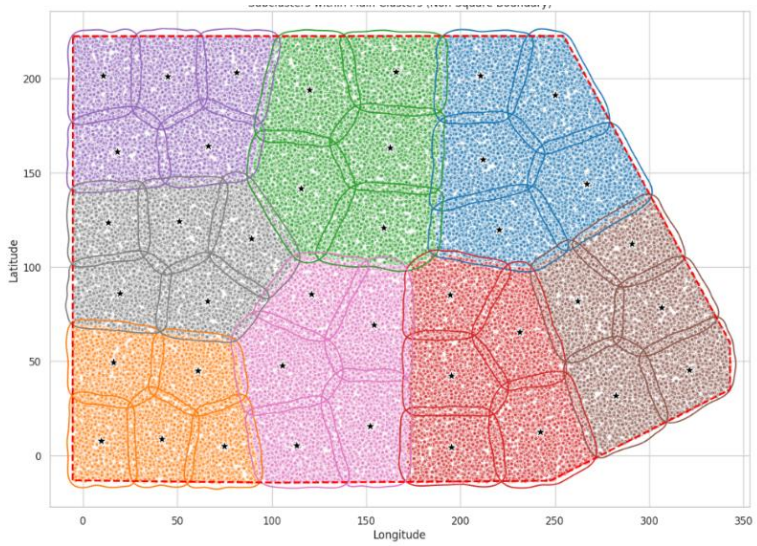
Selain spesifikasi teknis, *drone* Dji Agras T-30 juga telah mendukung fitur *autonomous operation*, dimana sistem *drone* secara otomatis dapat memantau indikator waktu estimasi hingga tangki habis secara *real time* sehingga *drone* dapat menjaga keseimbangan antara kapasitas baterai dan daya baterai. UAV ini juga dilengkapi dengan sistem *automatic edge sweeping* yang memungkinkan cakupan penyemprotan pada tepi lahan lebih efektif dan mempermudah operasi penyemprotan secara menyeluruh. Dengan demikian, model UAV ini sangat cocok untuk digunakan sebagai dasar perencanaan distribusi agrokimia berbasis optimasi rute dalam penelitian ini.

2.2.3 Stressed Region

Dalam pengerjaan tugas akhir, data *stressed region* digunakan sebagai bahan data sekunder. *Stressed Region* ini merujuk pada sebagian area tanaman pada lahan yang mengalami tekanan atau stress. Hal tersebut dapat disebabkan oleh berbagai faktor seperti kekurangan air, kekurangan nutrisi, atau serangan hama. Pemetaan *stressed region* ini berperan penting dalam mengambil tindakan perbaikan yang tepat pada area lahan yang memerlukan perhatian lebih. Pentingnya deteksi area stres pada tanaman telah diakui dalam penelitian modern, dimana teknologi UAV digunakan untuk mendukung praktik *precision agriculture* dengan menyediakan data pencitraan spektral beresolusi tinggi. Dengan adanya penerapan ini, para petani dapat memperoleh data akurat mengenai lokasi spesifik area yang mengalami tekanan sehingga memungkinkan pengambilan Keputusan yang lebih cepat dan tepat, terutama dalam konteks *precision agriculture*, Dimana UAV digunakan untuk mendeteksi dan memetakan *stressed region* tersebut (Srivastava et al., 2019).

2.2.4 Lloyd Algorithm

Lloyd algorithm atau biasa dikenal sebagai *k-means clustering*, merupakan sebuah metode algoritma yang digunakan untuk membagi Kumpulan data ke dalam sejumlah klaster. Algoritma ini bekerja secara iteratif menentukan pusat klaster (*centroid*) secara acak, lalu mengelompokkan titik data yang berdekatan menjadi satu klaster. Setelah pengelompokkan dilakukan, pusat klaster diperbarui berdasarkan posisi rata-rata titik data dalam klaster dan terus berlanjut hingga perubahan posisi *centroid* menjadi sangat kecil dan stabil. Hal tersebut menandakan hasil algoritma telah mencapai konvergensi. Untuk lebih jelasnya, berikut adalah contoh gambar dari *lloyd algorithm* (Ribeiro Junior et al., 2022).



Gambar 2.1 Contoh hasil klasterisasi dengan *lloyd algorithm*

Pada gambar ini, terlihat hasil penerapan *lloyd algorithm* dalam membagi lahan pertanian menjadi beberapa sub klaster. Setiap warna dalam gambar mewakili klaster-klaster utama yang terbentuk dan garis-garis yang membatasi di dalam klaster tersebut menunjukkan sub klaster yang dibentuk. Titik-titik dalam gambar menunjukkan posisi data individu yang terbentuk.

Namun demikian, pada tugas akhir ini, digunakan pendekatan klasterisasi satu tingkat dengan menerapkan *lloyd algorithm* secara langsung pada *stressed region* tanpa melakukan pembagian lebih lanjut menjadi subklaster. Penentuan jumlah klaster dilakukan melalui proses *grid search* dengan mempertimbangkan dua kriteria utama, yaitu rasio cakupan (*coverage ratio*) yang diharapkan melebihi 97%, serta rasio daerah tumpang tindih (*overlap ratio*) yang dijaga agar berada di bawah 20% (Srivastava et al., 2020). Dengan cara ini diharapkan algoritma dapat menentukan jumlah klaster optimal yang memenuhi target distribusi UAV pada area *stressed region*, sehingga membantu memaksimalkan efisiensi penyemprotan agrokimia di lahan pertanian.

2.2.5 Diagram Voronoi

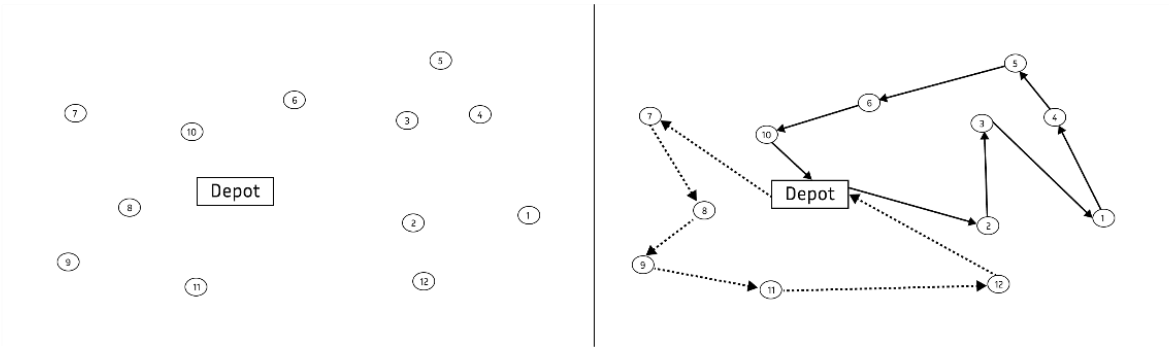
Diagram voronoi merupakan metode dalam komputasi geometrik yang membagi ruang kerja menjadi beberapa wilayah berdasarkan kedekatan terhadap titik-titik tertentu. Diagram ini membentuk sebuah poligon voronoi Dimana setiap poligon akan merepresentasikan area yang lebih dekat ke satu titik dibandingkan dengan titik lainnya. Secara fungsional, diagram voronoi sering digunakan dalam pemodelan suatu ruang kerja atau sebagai metode visualisasi yang efektif dalam penyajian hubungan spasial secara intuitif. Dalam konteks *path planning*, diagram voronoi biasa digunakan dalam memodelkan ruang bebas dari halangan sehingga menyediakan jalur dengan jarak yang maksimum dari objek-objek penghalang (Samuel et al., 2023).

Pada tugas akhir ini, diagram Voronoi digunakan untuk memvisualisasikan hasil klasterisasi dan optimasi rute UAV dalam proses distribusi agrokimia. Titik-titik *centroid* hasil klasterisasi dipetakan sebagai titik *generator* untuk membagi lahan pertanian ke dalam wilayah semprot masing-masing UAV. Setiap wilayah klaster diberi warna berbeda untuk membedakan domain kerjanya. Area *overlap* antar klaster ditandai dengan warna biru untuk menunjukkan kemungkinan tumpang tindih distribusi, sementara wilayah yang tercakup (*coverage*) ditandai dengan warna hijau. Adapun area yang tidak tercakup sama sekali oleh jangkauan radius UAV

(*uncovered*) divisualisasikan menggunakan warna merah untuk mengidentifikasi kekurangan distribusi. Diagram ini juga dilengkapi dengan anotasi luas area dan persentase cakupan tiap klaster guna mendukung analisis kuantitatif. Dengan demikian, diagram Voronoi dalam penelitian ini tidak hanya menjadi alat visualisasi spasial, tetapi juga media evaluasi performa distribusi agrokimia secara menyeluruh (Tahilyani et al., 2022).

2.2.6 *Vehicle Route Planning*

Vehicle Route Planning (VRP) merupakan sebuah pendekatan terkait optimasi rute yang bertujuan untuk mengoptimalkan rute kendaraan dalam meminimalkan biaya, waktu, dan jarak tempuh yang dibutuhkan untuk mencapai tujuan perjalanan (Arifuddin et al., 2024). Dengan mempertimbangkan kendala-kendala yang ada, seperti kapasitas kendaraan, permintaan dinamis, dan ketersediaan waktu operasional, VRP memiliki peran dalam memastikan setiap titik pelanggan dilayani tepat satu kali dengan meminimalkan biaya operasional secara keseluruhan. Hal ini digambarkan dengan gambar berikut ini.



Gambar 2.2 Ilustrasi Konsep *Vehicle Routing Problem*

Pada gambar di atas, proses VRP digambarkan dengan ilustrasi rute dua kendaraan yang telah melayani seluruh pelanggan yang ada secara efisien. Hal tersebut menandakan bahwa VRP merupakan elemen penting dalam optimisasi rute perjalanan.

Dalam konteks pertanian, VRP diterapkan untuk meningkatkan efisiensi operasional melalui penentuan jalur optimal bagi kendaraan, seperti traktor dan UAV dalam pelaksanaan tugas-tugas, seperti penyemprotan pestisida, penyebaran pupuk, dan pemanenan. Pendekatan ini mendukung tujuan pertanian secara presisi melalui pemetaan lahan sebagai graf berarah, di mana simpul-simpul mewakili lokasi tugas dan jalur di antaranya menunjukkan rute yang harus ditempuh (Bakhtiari et al., 2011). Selain itu, dalam implementasi VRP, terdapat beberapa faktor lain yang memengaruhi efisiensi pengerjaan operasional pertanian. Hal-hal tersebut dimulai dari konsumsi energi dan kemampuan manuver kendaraan pada lahan dengan kondisi medan yang berbeda-beda. Hal tersebut penting karena hambatan fisik seperti topografi dan vegetasi mempengaruhi pergerakan kendaraan. Optimasi rute menjadi salah satu metode yang efektif untuk meningkatkan efisiensi operasional pertanian. Dengan begitu, fungsi VRP dalam pertanian tidak hanya membantu pengurangan biaya operasional, tetapi juga mendukung praktik *Sustainable Agriculture* dalam penerapannya (Furchi et al., 2023)

Dalam menemukan solusi dalam permasalahan VRP, terdapat beberapa formulasi teknis yang perlu digunakan. Formulasi matematis ini mengacu pada pendekatan yang diterapkan oleh Utamima et al. (2019), yang berfokus pada perencanaan rute logistik dengan menggunakan model graf berbobot. Pendekatan ini mengintegrasikan konsep VRP dan *coverage path planning* (CPP) dengan menyesuaikan karakteristik UAV di medan pertanian. Formulasi ini

mencakup fungsi tujuan, variabel keputusan, dan batasan-batasan yang ada dalam mempertimbangkan kondisi lapangan.

Fungsi tujuan bertujuan untuk meminimalkan jarak tempuh total UAV dalam melayani seluruh area yang perlu dituju. Fungsi tujuan dapat dirumuskan sebagai berikut.

$$Min\ Z = \sum_{i \in N} \cdot \sum_{j \in N} d_{ij} \cdot x_{ij} \tag{1}$$

Dimana d_{ij} merupakan jarak antara *node* i dan j, serta x_{ij} adalah variabel biner yang memutuskan apakah UAV bergerak dari *node* i ke j. Berikutnya adalah variabel Keputusan yang digunakan dalam permasalahan VRP adalah sebagai berikut.

$$x_{ij}\ ; y_i \in \{0,1\} \tag{2}$$

Dalam permasalahan ini, variabel keputusan x_{ij} menentukan apakah UAV akan melakukan perjalanan dari *node* i ke j dan variabel y_i menentukan apakah UAV akan mengunjungi simpul i. Variabel ini digunakan untuk menentukan rute optimal UAV dalam melayani seluruh area *stressed region* sesuai batasan operasional. Yang terakhir, yaitu formulasi batasan. Batasan ini dirancang untuk menceeminkan kendala UAV di lapangan. Berikut 4 batasan utama yang diformulasikan sebagai berikut.

$$\sum_{j \in N} x_{ij} = 1, \forall i \in N \tag{3}$$

$$\sum_{j \in N} x_{ij} = \sum_{j \in N} x_{ji}\ , \forall i \in N \tag{4}$$

$$\sum_{i \in N} c_i \cdot y_i\ \leq C_{max} \tag{5}$$

$$\sum_{i \in N} \cdot \sum_{j \in N} t_{ij} \cdot x_{ij}\ \leq T_{max} \tag{6}$$

Batasan pertama (3) memastikan bahwa setiap *node* dalam *stressed region* harus dikunjungi tepat satu kali oleh UAV. Selanjutnya, batasan kedua (4) menjamin kesinambungan rute, di mana UAV yang memasuki simpul tertentu harus meninggalkannya kembali. Batasan ketiga (5) akan membatasi total kebutuhan sumber daya yang di bawa UAV agar tidak melebihi kapasitas maksimumnya. Terakhir, batasan keempat (6) mengatur waktu operasional UAV yang dihabiskan harus berada dalam batas waktu yang telah ditentukan. Kombinasi batasan ini memastikan bahwa rute yang dirancang efisien dan dapat dijalankan secara operasional mendekati kondisi di dunia nyata (Utamima et al., 2019).

2.2.7 Two-opt Local Search

Two-opt merupakan salah satu algoritma *local search* yang banyak digunakan untuk menyelesaikan masalah *Traveling Salesman Problem* (TSP) maupun *Vehicle Routing Problem* (VRP). Algoritma ini bekerja dengan memeriksa kemingkinan perbaikan solusi melalui pertukaran dua *edge* (*arc*) pada jalur yang sedang dievaluasi, dengan tujuan mengurangi total jarak rute yang dilalui. Prinsip kerja *two-opt* adalah memotong dua *edge* pada suatu jalur kemudian menyambungnya kembali dengan cara yang berbeda sehingga segmen jalur yang tadinya mungkin bersilangan (*crossing*) menjadi lebih pendek. Hal ini efektif untuk menghilangkan rute tidak efisien yang menambah panjang rute sehingga menjadi lebih efisien. Konsep ini sering digunakan dalam kombinasi dengan algoritma metaheuristik, seperti *Simulated Annealing* (SA) atau *Genetic Algorithm* (GA), untuk dapat menghindari jebakan solusi minimum lokal yang mudah terjadi pada masalah optimasi kombinatorial (Chandomi-Castellanos et al., 2022).

Chandomi-Castellanos et al. (2022) menerapkan *two-opt* sebagai tahap *local search* setelah proses *simulated annealing* pada penyelesaian TSP. *Pseudocode* algoritma *two-opt* yang digunakan dalam penelitian mereka ditunjukkan pada berikut ini:

ALGORITMA : *Pseudo-code* 2-opt

```
Input: vector mindx, Distance Matrix (D)
Output: Best route
minchange = −1
while minchange < 0 do
    mincambio = 0
    i = 0
    b = mindx(n)
    while i < n − 2 do
        a = b
        i = i + 1
        b = mindx(j)
        j = i + 1
        d = mindx(j)
        while i < n − 2 do
            c = d
            j = j + 1
            d = mindx(j)
            cambio = (D(a,c) − (D(c,d)) + D(b,d) − D(a,b)
            if mincambio < 0 then
                m(mini : minj − 1) = m(minj − 1 : −1 : mini)
```

Algoritma tersebut bekerja dengan dua *loop* bersarang yang memeriksa semua pasangan *edge* $(i,\ i+1)$ dan $(j,\ j+1)$ pada jalur $mindx$. Pada setiap iterasi, algoritma menghitung perubahan total jarak lintasan ($cambio$) jika dua *edge* tersebut ditukar. Jika $cambio$ bernilai negatif, artinya pertukaran tersebut menghasilkan lintasan yang lebih pendek. Indeks dari *edge* yang dibalik ($mini, minj$) kemudian disimpan. Setelah seluruh kombinasi diperiksa, apabila ditemukan perbaikan ($minchange < 0$), maka segmen jalur pada indeks tersebut dibalik (*reversed*) sehingga lintasan baru menjadi lebih efisien. Proses ini diulangi hingga tidak ditemukan lagi pertukaran *edge* yang dapat memperbaiki total jarak lintasan sehingga diperoleh

solusi minimum lokal (Chandomi-Castellanos et al., 2022). Konsep ini diadopsi dalam tugas akhir ini sebagai strategi *local search* yang terintegrasi pada algoritma *hybrid simulated annealing* unutk optimasi rute UAV pada distribusi agrokimia.

2.2.8 Simulated Annealing

Dalam tugas akhir ini, mahasiswa menggunakan *simulated annealing* (SA) sebagai metodologi algoritma dalam penyelesaian optimasi rute. SA merupakan sebuah algoritma metaheuristik yang umum digunakan untuk menyelesaikan permasalahan optimasi global dan kombinatorial dengan mengadaptasi proses fisik pendinginan logam dan metalurgi (Ariyani et al., 2018). Dalam proses ini, material logam dipanaskan pada suhu tinggi untuk mencapai keadaan dimana atom-atom dapat bergerak bebas dan berpindah posisi. Kemudian didinginkan secara perlahan untuk memungkinkan atom-atom untuk mencapai konfigurasi minimum sehingga struktur atom menjadi stabil. Konsep ini diadopsi ke dalam algoritma SA untuk mencari solusi optimal dengan menghindari jebakan pada solusi minimum lokal yang mudah didapatkan (Rodríguez-Esparza et al., 2024).

Dalam implementasi algoritma, SA dimulai dengan solusi awal yang dihasilkan secara acak pada suhu tinggi. Pada suhu ini, algoritma memiliki fleksibilitas untuk menerima solusi seadanya dengan probabilitas tertentu dimana angka suhu telah diatur sebelumnya. Seiring proses iterasi terus berjalan, suhu akan menurun secara bertahap sesuai dengan skema pendinginan yang telah ditentukan. Dengan cara ini, algoritma SA akan menemukan solusi yang optimal atau mendekati optimal (Rodríguez-Esparza et al., 2024).

Pada setiap iterasi, algoritma SA akan menciptakan solusi tetangga (Sn) dan solusi saat ini (S). Nilai solusi baru akan dilakukan seleksi berdasarkan besaran nilai *fitness* yang tercipta lebih baik daripada nilai solusi saat ini ($d2 < d1$). Jika solusi tetangga memiliki nilai *fitness* yang lebih buruk ($d2 > d1$), solusi tersebut dapat diterima dengan probabilitas tertentu yang dihitung menggunakan persamaan:

$$prob = \frac{1}{e^{\frac{(d2-d1)k}{d1 \cdot T}}} \tag{7}$$

Dari rumus (7), terdapat beberapa variabel kunci yang memengaruhi jalannya proses optimasi. Variabel pertama adalah nilai *fitness*, yang terdiri dari $d1$ sebagai nilai fitness solusi saat ini dan $d2$ sebagai nilai *fitness* dari solusi tetangga yang dihasilkan. Selanjutnya, suhu (T) dimana variabel tersebut yang mengatur probabilitas penerimaan solusi yang lebih buruk. Pada nilai suhu yang tinggi, algoritma cenderung menerima solusi kurang optimal. Konstanta penyesuaian (k) berperan dalam mengatur sensitivitas algoritma terhadap perubahan nilai *fitness* dan suhu yang mempengaruhi seberapa besar dampak suhu terhadap keputusan penerimaan solusi. Kombinasi variabel ini membantu menciptakan keseimbangan antara eksplorasi awal dan eksploitasi solusi optimal di tahap akhir (Ariyani et al., 2018).

Untuk menyelesaikan permasalahan optimasi pada tugas akhir, cara kerja algoritma SA digambarkan dalam *pseudo-code* berikut ini:

ALGORITMA : *Pseudo-code* SA

Inputs: I_{Iter}, α, T_0, M_{Acc}
$s \leftarrow$ Create initial solution
$T \leftarrow T_0$

```
4   while acc < M_Acc do
5     | for (k ← 1 to I_Iter) do
6     |   | s' ← Create neighbor solution(s)
7     |   | Δ = f(s') − f(s)
8     |   | if Δ ≤ 0 then
9     |   |   | s ← s'
10    |   | else
11    |   |   | if p^k > rand then
12    |   |   |   | s ← s'
13    |   | acc ← acc + 1
14    |   | k ← k + 1 ;
15    | T = T · α
16  Return: best solution found
```

Berdasarkan *pseudo-code* di atas, terlihat bahwa algoritma SA dimulai dengan pembuatan solusi awal s dan suhu awal T_0. Selama jumlah akses (acc) masih lebih kecil dari batas maksimum akses (M_{acc}), algoritma akan terus berjalan. Di setiap iterasinya (I_{Iter}), algoritma akan membuat solusi tetangga s' dan menghitung perubahan nilai *fitness* (Δ) yang ada. Jika solusi tetangga lebih baik atau sama dengan solusi saat ini, maka s' akan diadopsi menjadi solusi baru. Setelah iterasi dalam satu siklus telah selesai, suhu T akan diperbarui dengan dikalikan bersama faktor pendinginan (α) untuk mengurangi suhu secara periodik sehingga algoritma dapat mengarahkan solusi untuk lebih fokus pada solusi yang lebih optimal. Proses tersebut akan terus berulang hingga batas maksimum akses (M_{acc}) dan solusi terbaik didapatkan (Rodríguez-Esparza et al., 2024).

2.2.9 Evaluasi Hasil

Dalam optimasi rute, evaluasi hasil kinerja berperan dalam mengukur angka efektivitas algoritma yang digunakan untuk menyelesaikan masalah yang dihadapi. Pada optimasi rute dalam lahan pertanian, metrik kinerja utama yang sering digunakan dalam evaluasi mencakup dua metrik indikator, yaitu total jarak tempuh dan waktu perjalanan. Kedua parameter tersebut biasa digunakan dalam mencerminkan bentuk efisiensi dan kualitas rute perencanaan yang dihasilkan.

Total jarak tempuh (L) sebagai salah satu parameter yang biasa digunakan, menunjukkan panjang total lintasan yang harus ditempuh oleh kendaraan dalam menyelesaikan rute. Pada parameter ini, nilai jarak yang lebih kecil menunjukkan efisiensi yang lebih tinggi dalam penggunaan energinya. Jarak tempuh dapat dihitung dengan rumus berikut:

$$L = \sum_{i=1}^{K-1} L_i \tag{8}$$

$$L = \sum_{i=1}^{K-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{9}$$

Dalam rumus (9), x_i dan y_i merupakan koordinat dari titik i yang dikunjungi pada rute. Selain itu, K melambangkan jumlah total titik persinggahan yang dilewati pada rute yang terbuat.

Untuk parameter kedua, yaitu waktu perjalanan (T) merupakan parameter kinerja yang menggambarkan total waktu yang dibutuhkan untuk menyelesaikan satu rute yang terbuat dari titik awal dan titik akhir yang telah ditentukan. Dalam sektor pertanian, total waktu juga dipengaruhi oleh kondisi medan lahan dan jumlah rotasi yang dilakukan oleh kendaraan di setiap titik belok pada rute. Parameter tersebut dirumuskan sebagai berikut:

$$T = \sum_{i=1}^{K-1} T_{L_i} + \sum_{i=1}^{K-1} T_{\theta_i} \tag{10}$$

T_{L_i} dalam rumus di atas mewakili waktu untuk perjalanan linear dari titik i ke titik $i+1$ dan T_{θ_i} adalah total waktu rotasi yang terjadi selama perjalanan satu rute untuk menyesuaikan kendaraan terhadap rute yang harus ditempuh. Kedua parameter di atas memberikan gambaran komprehensif mengenai efektivitas kinerja algoritma yang digunakan dalam optimasi rute. Dalam optimasi rute, algoritma seperti *simulated annealing* bertujuan untuk mencoba sebisa mungkin meminimalisir angka kedua parameter tersebut secara bersamaan. Evaluasi nilai hasil kinerja ini dilakukan dengan membandingkan nilai-nilai tersebut terhadap batasan dan target yang ingin dicapai (Zangina et al., 2021).

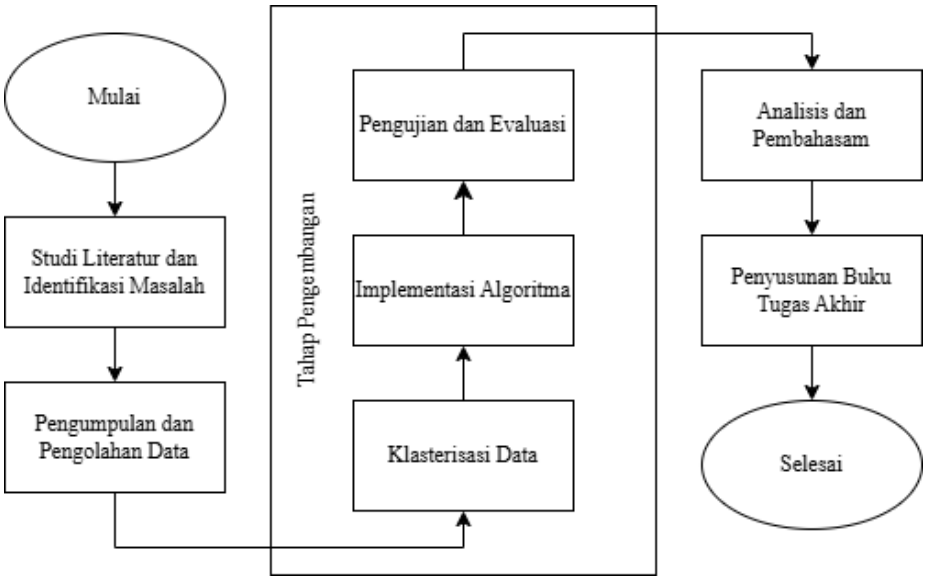
Dalam pelaksanaan evaluasi pada tugas akhir, *Google OR-Tools* digunakan sebagai alat pendukung untuk memperoleh hasil optimasi. *Google OR-Tools* adalah sebuah Pustaka *open source* yang dirancang menyelesaikan beberapa masalah, termasuk optimasi rute. *OR-Tools* mampu menghitung dan mengoptimalkan solusi VRP secara efisien dengan memperhitungkan batasan kapasitas kendaraan dan jarak tempuh total yang dimiliki. Dalam tugas akhir ini, *OR-Tools* digunakan untuk mengevaluasi kinerja algoritma optimasi berdasarkan parameter jarak, waktu tempuh, dan efisiensi distribusi sumber daya, yang diintegrasikan ke dalam formula perhitungan biaya. Dengan begitu, rute UAV yang terbuat akan lebih terukur dan efisien (Muriyatmoko et al., 2024).

BAB 3 METODOLOGI

Dalam bab metodologi, terdapat penjelasan mengenai metodologi yang akan diterapkan dalam penyusunan tugas akhir ini. Metode yang dijelaskan ini berfungsi sebagai arahan atau *guideline* dalam memastikan bahwa tugas akhir dapat dijalankan dengan terarah dan terorganisir. Penjelasan tersebut meliputi penjelasan metode melalui diagram alir hingga rincian langkah-langkah implementasi di dalam tugas akhir.

3.1 Metode yang digunakan

Subbab ini menjelaskan mengenai metode atau langkah-langkah teknis yang akan dilakukan di dalam tugas akhir. Metode-metode tersebut akan diuraikan melalui sebuah diagram alir yang akan ditunjukkan di bawah berikut ini:



Gambar 3.1 Diagram Alir Metodologi

3.2 Uraian Metodologi

Subbab ini akan membahas mengenai bahan dan peralatan yang digunakan pada tugas akhir. Dalam tugas akhir ini, akan dijabarkan lebih lanjut mengenai setiap proses pada tugas akhir yang telah ditunjukkan pada diagram alir sebelumnya.

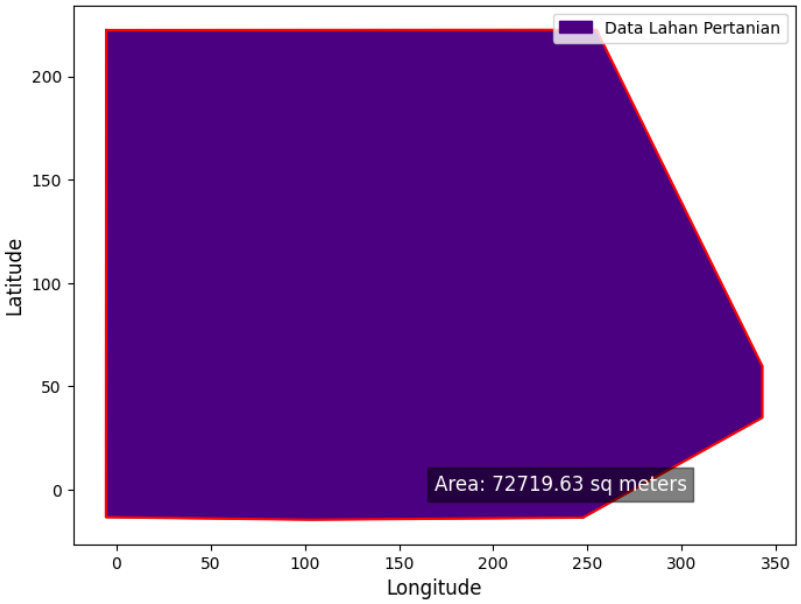
3.2.1 Studi Literatur dan Identifikasi Masalah

Pada tahap ini, dilakukan pembelejaran pada berbagai literatur tugas akhir. Studi ini bertujuan untuk penentuan topik tugas akhir yang akan dilakukan dalam tugas akhir ini. Setelah topik tugas akhir telah ditentukan, dilakukan pengumpulan dan pengkajian artikel tugas akhir terdahulu yang relevan dengan topik. Pengkajian pada literatur tugas kahir tersebut dilakukan untuk mendapatkan pemahaman mengenai teori dan metode yang akan mendukung tugas akhir. Setelah itu, topik permasalahan akan disesuaikan dengan kondisi dan kemampuan mahasiswa. Topik permasalahan yang telah disesuaikan, akan dirumuskan dalam penulisan bab

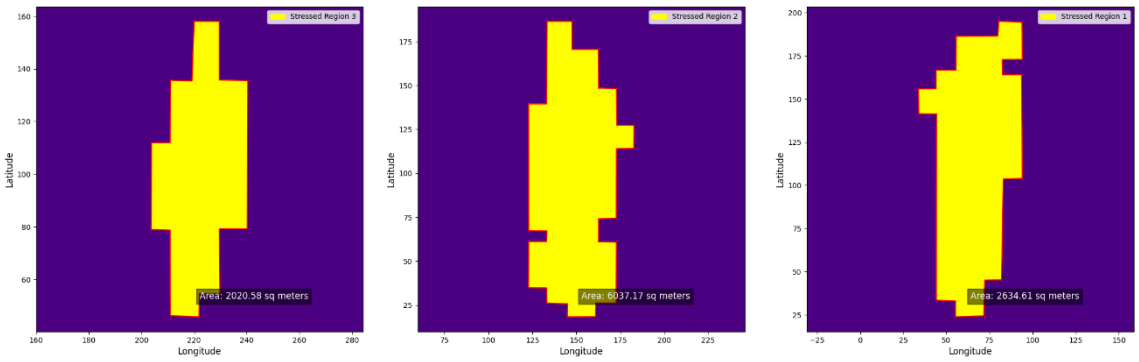
pendahuluan pada dokumen, meliputi penulisan latar belakang, rumusan masalah, tujuan, dan batasan masalah pada tugas akhir.

3.2.2 Pengumpulan dan Pengolahan Data

Pada tahap ini, dilakukan pengumpulan dan pengolahan data yang akan digunakan dalam analisis tugas akhir. Terdapat dua jenis data utama yang digunakan, yaitu data primer dan data sekunder. Kedua data tersebut diambil dari dua artikel penelitian yang berbeda. Data primer berupa data ladang pertanian yang berisi koordinat geografis lahan tersebut(Utamima et al., 2019). Data sekunder berupa informasi tentang *stressed region* yang diperoleh dari penelitian sebelumnya (Srivastava et al., 2019).



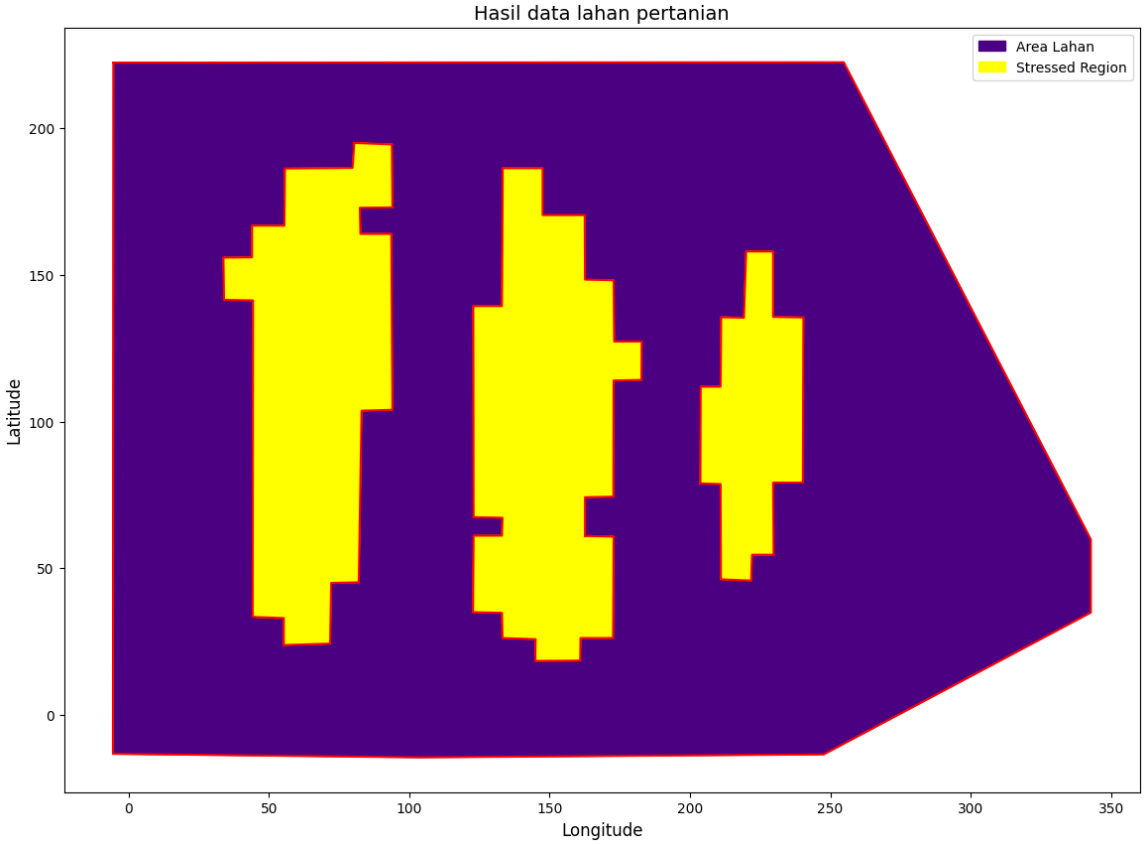
Gambar 3.2 Gambar lahan pertanian (data primer)



Gambar 3.3 Gambar data *stressed region* (data sekunder)

Setelah data terkumpul, data diproses untuk memastikan kesesuaiannya dengan kebutuhan tugas akhir. Proses ini dilakukan menggunakan perangkat lunak Desmos. Desmos merupakan sebuah perangkat lunak yang efektif untuk analisis dan visualisasi data geografis. Pada aplikasi tersebut, data *stressed region* digabungkan dengan data ladang pertanian untuk menghasilkan koordinat geografis berupa *longitude* dan *latitude* yang telah disesuaikan dengan data lapangan

sebenarnya. Hasil ini menghasilkan data yang siap digunakan untuk analisis lebih lanjut dalam proses optimasi.



Gambar 3.4 Hasil penggabungan data

3.2.3 Pembentukan Klasterisasi dengan *Lloyd Algorithm*

Dalam tahap ini, dilakukan klasterisasi pada data *stressed region* yang telah diproses atau pengelompokkan area lahan menjadi beberapa area daerah (klaster). Proses klasterisasi akan dilakukan menggunakan metode *lloyd algorithm* atau *K-Means clustering.* Pada waktu klasterisasi dilakukan, inisiasi klaster dalam area akan dibuat secara acak. Setelah itu, dalam setiap kelompok klaster akan terbentuk titik centroid yang menjadi pusat titik dari klaster tersebut. Proses ini akan dilakukan berulang sesuai dengan angka iterasi yang diinginkan hingga terbentuk sebuah kelompok klaster yang sesuai.

Untuk menentukan jumlah *cluster* yang efektif pada setiap area *stressed region*, dilakukan proses *grid search* dengan mencoba berbagai jumlah *cluster* dan mengevaluasinya menggunakan dua indikator utama, yaitu rasio cakupan area (*coverage ratio*) yang ditargetkan minimal 97%, serta persentase tumpang tindih area (*overlap ratio*) yang dijaga tidak melebihi 20%. Metode evaluasi ini mengadaptasi pendekatan yang digunakan oleh Srivastava (2020), dimana pada penelitian tersebut mempertimbangkan cakupan area minimum dan menimalkan area *overlap* agar ditribusi input pertanian lebih efisien. Dengan pendekatan tersebut, diperoleh konfigurasi klaster yang tidak hanya membagi area lahan secara spasial, tetapi juga mendukung efektivitas distribusi agrokimia yang lebih presisi (Srivastava et al., 2020).

3.2.4 Implementasi Metode Algoritma

Pada tahap ini, algoritma optimasi yang dipilih diterapkan untuk menyelesaikan permasalahan optimasi rute UAV dalam distribusi agrokimia pada area *stressed region*. Setelah klasterisasi dilakukan dan diperoleh *centroid* sebagai representasi titik semprot, algoritma *hybrid simulated annealing* (SA) dikembangkan uuntuk mengoptimasi urutan kunjungan UAV terhadap titik *centroid* tersebut. Algoritma *hybrid* SA ini mengombinasikan mekanisme inisialisasi solusi awal secara *greedy* dengan proses *local search two-opt* yang dijalankan secara periodik sehingga dapat mengeksplorasi ruang solusi sekaligus memperbaiki solusi pada tingkat lokal dengan lebih efektif.

Implementasi algoritma dilakukan dengan mengikuti langkah-langkah pada *pseudocode* yang telah disusun, dengan penentuan parameter-parameter, seperti *initial temperature*, *cooling rate*, *two-opt interval*, dan jumlah iterasi maksimum yang disesuaikan berdasarkan kebutuhan eksperimen. Selanjutnya, proses optimasi akan disimulasikan untuk mengevaluasi performa algoritma dalam menghasilkan lintasan UAV yang efisien.

3.2.5 Pengujian dan Evaluasi

Pada tahap pengujian dan evaluasi, hasil kinerja algoritma dalam optimasi rute UAV akan dianalisis. Untuk menilai efektivitas algoritma, digunakan dua parameter utama, yaitu total jarak tempuh yang menunjukkan panjang lintasan UAV dalam menjangkau seluruh *centroid* dan estimasi waktu penerbangan yang dihitung berdasarkan kecepatan rata-rata UAV. Selain itu, total waktu komputasi (*runtime*) selama proses optiamsi juga dicatat sebagai bahan pertimbangan performa algoritma. Dalam pengujian ini, dilakukan proses *grid search* pada beberapa variasi parameter algoritma algoritma *hybrid* SA untuk memperoleh hasil variasi konfigurasi terbaik. Masing-masing kombinasi konfigurasi diuji dengan sejumlah *run* independen guna mengevaluasi rata-rata performa serta kestabilan solusi yang diperoleh.

Sebagai tolok ukur, dilakukan pengujian menggunakan Google OR-Tools, sebuah pustaka *open source* yang umum digunakan pada industri untuk menyelesaikan permasalahan *vehicle routing problem* (VRP). Hasil lintasan UAV yang dihasilkan oleh algoritma *hybrid* SA kemudian dibandingkan secara langsung dengan lintasan yang diperoleh dari optimasi menggunakan OR-Tools sehingga dapat memberikan gambaran objektif mengenai kualitas solusi yang dihasilkan (Muriyatmoko et al., 2024).

3.2.6 Analisis dan Pembahasan

Tahap ini bertujuan untuk melakukan analisis terhadap hasil yang diperoleh dari proses pengujian dan evaluasi. Analisis dilakukan dengan mengevaluasi sejauh mana metode algoritma *hybrid* SA yang diterapkan efektif dalam menghasilkan lintasan UAV, baik dalam konteks total jarak tempuh, estimasi waktu penerbangan, maupun total waktu komputasi yang dihasilkan. Pada bagian ini, akan dibahas pula variasi hasil yang diperoleh antar parameter algoritma yang diuji, serta identifikasi kelebihan dan keterbatasan pendekatan yang digunakan. Hasil analisis ini diharapkan dapat memberikan dasar yang kuat bagi pengembangan penelitian serupa di masa mendatang.

3.2.7 Penyusunan Buku Tugas Akhir

Setelah seluruh tahapan telah selesai dilakukan, tahapan selanjutnya dan yang terakhir adalah penyusunan buku tugas akhir yang mendokumentasikan keseluruhan pengerjaan, mulai dari proses hingga hasil yang didapat dalam bentuk laporan secara sistematis. Dalam buku ini, mahasiswa juga menjelaskan seluruh hasil temuan dan analisis yang diperoleh di semua

tahapannya. Selain menjadi catatan lengkap keseluruhan penelitian, dokumentasi ini juga memberikan gambaran besar tentang perjalanan dari tugas akhir tersebut dan diharapkan akan menjadi sebuah referensi yang menunjukkan kontribusi ilmiah dari penelitian-penelitian yang telah dilakukan pada pengembangan teknologi optimasi rute UAV dalam *urban computing*.

BAB 4 HASIL DAN PEMBAHASAN

Dalam bab ini, terdapat uraian hasil dari penerapan metode yang telah dijelaskan pada bab sebelumnya serta pembahasan terhadap penemuan yang telah diperoleh selama proses implementasi. Setiap tahap yang telah dilaksanakan diuraikan secara bertahap untuk menunjukkan bagaimana metode yang digunakan dapat menghasilkan solusi dalam permasalahan optimasi pada area target UAV.

4.1 Deskripsi Data

Data yang digunakan dalam tugas akhir terdiri dari kumpulan titik koordinat yang merepresentasikan lokasi pada suatu lahan pertanian, termasuk area yang diklasifikasikan sebagai *stressed region*. Dataset diperoleh melalui integrasi dari dua referensi utama. Data koordinat lahan sebagai data primer didapatkan dari penelitian (Utamima et al., 2019). Sementara itu, informasi terkait klasifikasi *stressed region* didapatkan melalui data visualisasi dari penelitian (Srivastava et al., 2019) sebagai data sekunder. Kedua data tersebut diproses ulang menyesuaikan dengan kebutuhan pada proses pengerjaan tugas akhir.

Dataset ini memiliki 73 entri dengan empat atribut, yaitu Label, X, Y, dan Region. Label berfungsi sebagai identifikasi unik untuk setiap titik koordinat dalam dataset yang diberi nama secara berurutan, mulai dari a1, a2, dan seterusnya, guna memudahkan proses pelacakan dan analisis pada tahap klasterisasi serta optimasi rute. Atribut X dan Y merepresentasikan nilai koordinat spasial dua dimensi setiap titik, dimana X menunjukkan *longitude* atau posisi horizontal pada lahan pertanian dan Y menunjukkan *latitude* atau posisi vertikal pada lahan pertanian. Kedua koordinat tersebut menggambarkan letak geografis setiap titik pada ruang lingkup UAV.

Adapun atribut Region merupakan label kategorikal yang mengidentifikasi posisi relatif setiap titik pada struktur spasial lahan. Atribut ini memiliki empat kategori nilai, yaitu outer, inner_1, inner_2, dan inner_3. Nilai outer menunjukkan titik-titik yang membentuk batas kontur lahan secara keseluruhan dan merepresentasikan kerangka utama lahan pertanian. Sementara itu tiga label inner digunakan untuk mengelompokkan titik-titik yang berada di dalam lahan dan merepresentasikan tiga area *stressed region* yang berbeda. Dengan adanya klasifikasi ini, setiap titik dapat dikenali berdasarkan letak dan fungsi dalam struktur spasial lahan.

Untuk memberikan Gambaran yang lebih jelas mengenai struktur dan isi data yang digunakan, berikut disajikan keseluruhan dataset yang memuat informasi koordinasi titik pada lahan pertanian, sebagaimana ditampilkan pada tabel 4.3.

Tabel 4.1 Data Koordinat Titik pada Lahan Pertanian

Label	X	Y	Region
a1	-539	22238	outer
a2	25476	2225	outer
a3	34279	5979	outer
a4	34279	349	outer
a5	24764	-1346	outer
a6	10375	-1448	outer

a7	-556	-1327	outer
a8	804	195	inner_1
a9	938	1945	inner_1
a10	94	173	inner_1
a11	824	1729	inner_1
a12	826	164	inner_1
a13	936	164	inner_1
a14	94	104	inner_1
a15	83	1037	inner_1
a16	82	452	inner_1
a17	722	45	inner_1
a18	718	243	inner_1
a19	553	238	inner_1
a20	553	33	inner_1
a21	442	334	inner_1
a22	4434	1413	inner_1
a23	34	1415	inner_1
a24	338	1559	inner_1
a25	44	156	inner_1
a26	44	1668	inner_1
a27	555	1668	inner_1
a28	557	1863	inner_1
a29	798	1865	inner_1
a30	1626	61	inner_2
a31	1728	608	inner_2
a32	1726	262	inner_2
a33	161	262	inner_2
a34	1609	185	inner_2
a35	1449	184	inner_2
a36	145	258	inner_2
a37	1332	262	inner_2
a38	133	348	inner_2
a39	1227	35	inner_2
a40	1229	611	inner_2
a41	133	611	inner_2
a42	1332	672	inner_2
a43	1229	674	inner_2
a44	1228	1394	inner_2
a45	133	1394	inner_2
a46	1333	1864	inner_2
a47	1475	1864	inner_2
a48	1475	1704	inner_2
a49	1625	1704	inner_2
a50	1626	1484	inner_2
a51	1728	1482	inner_2
a52	173	1273	inner_2

a53	1828	1273	inner_2
a54	1827	1142	inner_2
a55	1728	114	inner_2
a56	1727	744	inner_2
a57	1626	742	inner_2
a58	2037	789	inner_3
a59	2038	1119	inner_3
a60	2109	1119	inner_3
a61	2111	1356	inner_3
a62	2192	1354	inner_3
a63	220	158	inner_3
a64	2296	158	inner_3
a65	2296	1357	inner_3
a66	2404	1355	inner_3
a67	2402	792	inner_3
a68	2296	792	inner_3
a69	2298	546	inner_3
a70	2221	546	inner_3
a71	2218	458	inner_3
a72	211	462	inner_3
a73	2109	787	inner_3

4.2 Environment

Bagian ini menguraikan *environment* yang digunakan dalam proses pengerjaan tugas akhir, meliputi spesifikasi perangkat keras serta perangkat lunak yang mendukung kegiatan pengembangan sistem. Rincian spesifikasi perangkat keras yang digunakan dapat dilihat pada Tabel 4.2.

Tabel 4.2 Perangkat Keras Yang Digunakan

Perangkat	VivoBook ASUS Laptop
Processor	11th Gen Intel® Core™ i5-1135G7
Memori	8 GB
Sistem Operasi	Microsoft Windows 11 Home Single Language

Adapun perangkat lunak yang digunakan dalam proses pembangunan model dijelaskan pada Tabel 4.3.

Tabel 4.3 Perangkat Lunak Dan Pustaka Python yang Digunakan

Bahasa Pemrograman	Python
Tools	Cursor
Pustaka	pandas, numpy, matplotlib, shapely, sklearn, scipy, ortools, itertools, tqdm, csv, collections, time, random

4.3 Implementasi

Bagian ini menjelaskan tahapan implementasi yang telah dilakukan, mulai dari proses pengolahan data hingga proses optimasi UAV pada lahan. Seluruh langkah dilakukan berdasarkan diagram alir metode yang telah dirancang pada bab sebelumnya.

4.3.1 Pengolahan Data

Tahap Implementasi diawali dengan pengolahan data yang dibagi menjadi beberapa tahapan yang mencakup pengumpulan data dan praproses data. Tahapan ini bertujuan untuk menyiapkan struktur data spasial yang digunakan untuk optimasi rute UAV. Berikut penjelasan masing-masing tahapan sebagai berikut.

4.3.1.1 Pengumpulan Data

Tahapan pertama adalah pengambilan data koordinat lahan pertanian yang telah diklasifikasikan berdasarkan wilayahnya. Data tersebut tersedia dalam format *comma-separated values* (CSV) dan diakses secara daring dari repositori GitHub menggunakan pustaka pandas. Pemanggilan dilakukan dengan membaca URL sumber data dan memuatnya ke dalam struktur *data frame*. Proses pemanggilan data dilakukan dengan menggunakan sintaks read_csv. Kode program 4.1 berikut menunjukkan proses pemanggilan data dari GitHub dan penampungannya dalam bentuk *data frame*.

```
import pandas as pd
url =
"https://raw.githubusercontent.com/naufalaschafitz/TugasAkhir/refs/heads/main/coordinates_with_region.csv"

df = pd.read_csv(url)
```

Kode 4.1 *Import* Data dari CSV

4.3.1.2 Segmentasi Data dan Pembentukan Poligon Tertutup

Pada bagian ini, data yang telah dimuat akan dikelompokkan berdasarkan atribut Region. Segmentasi ini dilakukan dengan memanfaatkan metode *indexing* pada pandas untuk memfilter data sesuai dengan nilai pada kolom Region.

Setelah segmentasi selesai, dilakukan pembentukan poligon tertutup pada masing-masing wilayah. Proses ini dilakukan dengan memastikan titik awal pada setiap kumpulan koordinat ditambahkan kembali di posisi terakhir, dengan begitu terbentuklah sebuah lintasan tertutup yang dapat di representasikan sebagai objek poligon menggunakan pustaka shapely. Berikut implementasi kode ditampilkan pada kode berikut ini.

```
inner_regions = ["inner_1", "inner_2", "inner_3"]
inner_points_dict = {}
inner_polygons = {}

for region in inner_regions:
    points = df[df["Region"] == region][["X", "Y"]].values
    if len(points) > 0:
        inner_points_dict[region] = points
        # pastikan polygon tertutup
        if not np.array_equal(points[0], points[-1]):
            points = np.vstack([points, points[0]])
```

```
        inner_polygons[region] = Polygon(points)

outer_points = df[df["Region"] == "outer"][["X", "Y"]].values
if len(outer_points) > 0:
    if not np.array_equal(outer_points[0], outer_points[-1]):
        outer_points = np.vstack([outer_points, outer_points[0]])
    outer_polygon = Polygon(outer_points)
else:
    outer_polygon = None
```

Kode 4.2 Segmentasi Data dan Pembentukan Poligon Tertutup

4.3.2 Klasterisasi Data

Klasterisasi dilakukan untuk membentuk representasi spasial yang terstruktur pada setiap wilayah *stressed region*. Tujuan dari proses ini adalah untuk membagi area target menjadi beberapa kelompok klaster sehingga memudahkan UAV dalam menentukkan urutan lintasan penyemprotan agrokimia secara efektif. Proses tersebut dilakukan menggunakan pendekatan dengan algoritma lloyd (K-Means), yang diawali dengan pembuatan titik secara acak dalam area wilayah (*sampling*), kemudian dilanjutkan dengan proses *grid search* dalam menentukan jumlah klaster, pembentukan klaster, dan visualisasi hasil klaster yang terbentuk. Berikut rangkaian keseluruhan proses sebagai berikut.

4.3.2.1 Pembuatan Titik Acak dalam Wilayah (Region-based *Sampling)*

Tahap awal klasterisasi diawali dengan pembuatan sejumlah titik sampel secara acak di dalam setiap *stressed region*, yaitu area inner_1, inner_2, dan inner_3. Secara teknis, titik-titik tersebut dihasilkan melalui proses *random sampling* sebanyak nilai n_samples yang ditentukan. Namun, titik-titik hanya akan diterima dan disimpan jika titik berada dalam batas poligon dari area target. Berikut adalah fungsi generate_points_within_boundary() yang digunakan untuk menjalankan proses berikut.

```
def generate_random_points_in_polygon(polygon, n_samples):
    minx, miny, maxx, maxy = polygon.bounds
    points = []
    while len(points) < n_samples:
        x = np.random.uniform(minx, maxx)
        y = np.random.uniform(miny, maxy)
        p = Point(x, y)
        if polygon.contains(p):
            points.append((x, y))
    return np.array(points)
```

Kode 4.3 Fungsi Untuk Menghasilkan Titik Acak Di Dalam Batas Poligon

Setelah fungsi didefinisikan, langkah berikutnya adalah mengsekusi fungsi tersebut. Untuk menjamin representasi spasial yang padat dan merata, masing-masing *stressed region* akan dilakukan proses *sampling* sebanyak 20.000 titik.

```
points_inner_1 = generate_random_points_in_polygon(inner_polygons["inner_1"],
n_samples=20000)
points_inner_2 = generate_random_points_in_polygon(inner_polygons["inner_2"],
n_samples=20000)
```

```
points_inner_3 = generate_random_points_in_polygon(inner_polygons["inner_3"],
n_samples=20000)
```

Kode 4.4 Menghasilkan Titik Acak dalam Masing-masing Inner Region

4.3.2.2 Proses *Grid Search* Evaluasi *Overlap-Coverage*

Pada tahap ini, dilakukan proses grid search untuk mecari konfigurasi jumlah klaster yang paling efektif dimana hasil tersebut akan digunakan dalam perencanaan rute penyemprotan rute UAV. Terdapat dua bentuk evaluasi yang digunakan dalam menentukan jumlah klaster tersebut, yaitu persentase cakupan area (*coverage ratio*) dan persentase area tumpang tindih (*overlap ratio*). Fungsi compute_uav_coverage_ratio() digunakan untuk menghitung *coverage ratio*, yaitu persentase area *stressed region* yang tercakup oleh kumpulan buffer lingkaran radius semprot UAV pada *centroid*. Selanjutnya, fungsi compute_overlap_between_centroids() digunakan untuk menghitung total area *overlap* antar buffer radius semprot dari setiap *centroid*. Berikut kode fungsi tersebut sekaligus kode fungsi yang digunakan untuk klasterisasi dan juga visualisasinya ditunjukkan di bawah ini.

```
def perform_clustering_single(points, n_clusters, random_state):
    kmeans = KMeans(n_clusters=n_clusters, random_state=random_state, n_init=10)
    kmeans.fit(points)
    cluster_labels = kmeans.labels_
    centroids = kmeans.cluster_centers_
    return points, cluster_labels, centroids

def plot_clusters_single(ax, points, labels, centroids, label_prefix,
spray_radius=9):
    colors = plt.cm.get_cmap("tab10", len(set(labels)))
    unique_clusters = sorted(set(labels))
    cluster_map = {global_id: local_id for local_id, global_id in
enumerate(unique_clusters)}

    for i in unique_clusters:
        cluster_points = points[labels == i]
        ax.scatter(cluster_points[:,0], cluster_points[:,1],
                   color=colors(cluster_map[i]), alpha=0.6, edgecolor='w', s=10)
    for idx, (cx, cy) in enumerate(centroids):
        ax.scatter(cx, cy, marker='*', s=150, c='black', edgecolor='w')
        circle = plt.Circle((cx, cy), radius=spray_radius, color='black',
alpha=0.2, linewidth=1, fill=True)
        ax.add_patch(circle)
        ax.text(cx + 1.5, cy + 1.5, f"{label_prefix}-C{idx}", fontsize=8,
color='black')

def compute_uav_coverage_ratio(centroids, boundary_polygon, radius):
    coverage_circles = [Point(cx, cy).buffer(radius) for (cx, cy) in centroids]
    total_coverage_union = unary_union(coverage_circles)
    effective_coverage = total_coverage_union.intersection(boundary_polygon)
    ratio = (effective_coverage.area / boundary_polygon.area) * 100
    return effective_coverage.area, boundary_polygon.area, ratio
```

```
def compute_overlap_between_centroids(centroids, radius):
    coverage_circles = [Point(cx, cy).buffer(radius) for (cx, cy) in centroids]
    total_overlap_area = 0.0
    for i, j in combinations(range(len(coverage_circles)), 2):
        circle_i = coverage_circles[i]
        circle_j = coverage_circles[j]
        if circle_i.intersects(circle_j):
            intersection = circle_i.intersection(circle_j)
            total_overlap_area += intersection.area
    return total_overlap_area
```

Kode 4.5 Fungsi Utama untuk Klasterisasi, Perhitungan *Coverage*, dan *Overlap* UAV

Proses *grid search* dilakukan pada masing-masing *stressed region* dengan mencoba berbagai jumlah klaster pada rentang 5 hingga 50. Pada setiap iterasi, titik *sampling* diklasterisasi dengan perform_clustering_single(), hasil *centroid* yang terbentuk akan dievaluasi *coverage ratio*-nya dengan compute_uav_coverage_ratio(), dan dihitung *overlap ratio* antar radius *centroid* dengan compute_overlap_between_centroids(). Hasil kedua metrik tersebut akan dievaluasi dengan hasil *coverage ratio* tidak kurang dari 97% dan hasil *overlap ratio* kurang dari 20%. Hasil terbaik dari kedua syarat metrik akan dipilih sebagai jumlah klaster yang akan digunakan dalam pengerjaan tugas akhir. Berikut kode dalam proses *grid search* ditampilkan dalam kode berikut.

```
cluster_range = range(5, 50, 1)

results_summary = []
optimal_centroids_per_inner = {}
all_results_list = []

for region in ["inner_1", "inner_2", "inner_3"]:
    print(f"\n=== OPTIMASI UNTUK {region.upper()} ===")
    points_inner = generate_random_points_in_polygon(inner_polygons[region],
n_samples=20000)
    best_sol = None
    best_loss = float('inf')

    for n_clusters in cluster_range:
        points, labels, centroids = perform_clustering_single(points_inner,
n_clusters, random_state)
        covered_area, area_inner, coverage_ratio =
compute_uav_coverage_ratio(centroids, inner_polygons[region], radius)
        overlap_area = compute_overlap_between_centroids(centroids, radius)
        overlap_ratio = (overlap_area / covered_area) * 100 if covered_area > 0
else 0
        loss = (3 * ((100 - coverage_ratio) ** 2)) + ((overlap_ratio - 20) ** 2)

        all_results_list.append({
            "Inner": region,
            "Clusters": n_clusters,
            "Coverage (%)": coverage_ratio,
```

```
            "Overlap (%)": overlap_ratio,
            "Loss": loss
        })

        print(f"Clusters: {n_clusters:2d} | Coverage: {coverage_ratio:6.2f}% |
Overlap: {overlap_ratio:6.2f}% | Loss: {loss:8.2f}")

        if coverage_ratio >= 97 and abs(overlap_ratio - 20) < 10:
            if loss < best_loss:
                best_sol = {
                    "region": region,
                    "clusters": n_clusters,
                    "coverage": coverage_ratio,
                    "overlap": overlap_ratio,
                    "centroids": centroids
                }
                best_loss = loss

    if best_sol:
        optimal_centroids_per_inner[region] = best_sol["centroids"]
        results_summary.append({
            "Inner": region,
            "Optimal Clusters": best_sol["clusters"],
            "Coverage (%)": f"{best_sol['coverage']:.2f}",
            "Overlap (%)": f"{best_sol['overlap']:.2f}"
        })
    else:
        results_summary.append({
            "Inner": region,
            "Optimal Clusters": "Not Found",
            "Coverage (%)": "N/A",
            "Overlap (%)": "N/A"
        })
```

Kode 4.6 *Grid Search* Evaluasi Klasterisasi *Stressed Region*

Melalui proses ini, hasil konfigurasi klasterisasi pada masing-masing wilayah *stressed region* dapat digunakan pada tahap klasterisasi akhir dan tahap optimasi rute UAV selanjutnya.

4.3.2.3 Proses Klasterisasi Hasil *Grid Search*

Setelah diperoleh jumlah klaster masing-masing area dari proses *grid search*, tahap selanjutnya adalah melakukan proses klasterisasi akhir. Klasterisasi dilakukan dengan menggunakan jumlah klaster yang diperoleh. Proses klasterisasi dijalankan dengan fungsi perform_clustering_single() pada titik *sampling* masing-masing wilayah ddengan parameter jumlah klaster (n_clusters) yang telah didapatkan sebelumnya. Berikut potongan kode dari proses tersebut.

```
points_1, labels_1, centroids_1 = perform_clustering_single(points_inner_1, 33,
random_state)
```

```
points_2, labels_2, centroids_2 = perform_clustering_single(points_inner_2, 34,
random_state)
points_3, labels_3, centroids_3 = perform_clustering_single(points_inner_3, 14,
random_state)
```

Kode 4.7 Kode Menjalankan Klasterisasi

Hasil *centroid* yang terbentuk pada masing-masing area *stressed region* akan digunakan sebagai representasi titik semprot UAV. Dengan demikian, klasterisasi ini akan menjadi dasar utama dalam membentuk dataset yang akan diproses pada tahap optimasi rute UAV/

4.3.2.4 Visualisasi Hasil Klasterisasi

Tahap terakhir dari proses klasterisasi adalah melakukan visualisasi hasil klasterisasi yang dilakukan, sekaligus menyimpan koordinat *centroid* yang telah diperoleh ke dalam sebuah file CSV. Visualisasi dilakukan untuk memberikan gambaran spasial mengenai distribusi klaster pada masing-masing area *stressed region*. Visualisasi ini memetakan titik-titik *sampling*, klaster, *centroid*, dan buffer lingkaran di setiap *centroid* yang merepresentasikan radius smprot UAV pada masing-masing *centroid*.

Visualisasi dilakukan dengan menjalankan fungsi plot_clusters_single() yang telah dibuat sebelumnya. Selain divisualisasikan, seluruh koordinat *centroid* akan dikumpulkan ke dalam sebuah tabel dan disimpan dalam *file* CSV untuk memudahkan proses lanjutan pada tahap optimasi rute UAV. Berikut adalah kode yang digunakan dalam proses tersebut.

```
fig, ax = plt.subplots(figsize=(14, 10))
for region, points in inner_points_dict.items():
    closed_points = np.vstack([points, points[0]]) if not
np.array_equal(points[0], points[-1]) else points
    ax.plot(*zip(*closed_points), linestyle='-', color='black', linewidth=1.5)

plot_clusters_single(ax, points_1, labels_1, centroids_1, "Inner1")
plot_clusters_single(ax, points_2, labels_2, centroids_2, "Inner2")
plot_clusters_single(ax, points_3, labels_3, centroids_3, "Inner3")

ax.set_title("Single-level KMeans Clustering + UAV Coverage + Overlap Analysis",
fontsize=14)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.axis("equal")
ax.grid(True)
plt.show()

# Buat dataframe centroid untuk tiap inner
data_inner_1 = pd.DataFrame(centroids_1, columns=["X", "Y"])
data_inner_1["Region"] = "inner_1"

data_inner_2 = pd.DataFrame(centroids_2, columns=["X", "Y"])
data_inner_2["Region"] = "inner_2"

data_inner_3 = pd.DataFrame(centroids_3, columns=["X", "Y"])
data_inner_3["Region"] = "inner_3"
```

```
# Gabungkan semua jadi satu tabel
centroids_df = pd.concat([data_inner_1, data_inner_2, data_inner_3],
ignore_index=True)

# Simpan ke CSV
centroids_df.to_csv("centroids_clusters_with_region.csv", index=False)

print("File CSV berhasil dibuat: centroids_clusters_with_region.csv")
```

Kode 4.8 Visualisasi Klaster dan Penympanan Hasil *Centroid*

4.3.3 Desain Algoritma

Bagian ini menjelaskan proses perancangan algoritma yang digunakan pada optimasi rute UAV dalam distribusi algrokimia pada area *stressed region*. Dalam tugas akhir ini, dilakukan perancangan algoritma melalui pendekatan berbasis tiga algoritma utama, yaitu *Simulated Annealing* (SA) sebagai algoritma inti, *Local Search* sebagai strategi peningkatan solusi, dan pendekatan *greedy* sebagai metode pembentukan solusi awal. Ketiga pendekatan ini digunakan secara terstruktur untuk menyusun rute UAV yang efisien dari segi jarak tempuh dan waktu *runtime*.

Langkah awal dimulai dengan memuat dua sumber data yang telah disimpan sebelumnya, yaitu data koordinat lahan pertanian (coordinates_withregion.csv) dan hasil centroid klasterisasi (Centroids.csv). Kedua data tersebut diakses dari repositori Github mahasiswa melalui link URL *file* tersebut. Untuk melengkapi data tersebut ditambahkan titik depot UAV pada koordinat (0,0) sebagai titik awal dan akhir rute.

4.3.3.1 Pembentukan *Distance Matrix* dan Fungsi Evaluasi

Langkah awal dalam proses optimasi adalah membentuk *distance matrix*, yaitu sebuah metrics dua dimensi berisi jarak antar semua titik yang ada. Perhitungan jarak antar titik dilakukan dengan menggunakan metode *euclidean distance* melalui fungsi cdist. Fungsi ini bersifat simetris dan digunakan sebagai basis evaluasi dalam seluruh proses optimasi. Setelah *distance matrix* terbentuk, dibuat *fitness function* untuk mengukur kualitas rute UAV berdasarkan Panjang total lintasan yang harus ditepuh. Fngsi tersebut akan terus digunakan selama proses algoritma berjalan untuk menentukan apakah Solusi baru lebih baik daripada Solusi sebelumnya. *Fitness function* menghitung akumulasi jarak antar titik secara berurutan sesuai urutan dalam rute kemudian menambahkan jarak dari titik terakhir Kembali ke titik depot awal sehingga membentuk rute tertutup. Berikut merupakan kode yang digunakan dalam proses ini

```
distance_matrix = cdist(coords_with_depot, coords_with_depot)

def fitness(tour, distance_matrix):
    return sum(distance_matrix[tour[i], tour[i+1]] for i in range(len(tour)-1))
+ distance_matrix[tour[-1], tour[0]]
```

Kode 4.9 Pembuatan Jarak Matriks *Euclidean* dan fungsi *fitness* untuk TSP

Komponen ini membentuk kerangka daasar bagi seluruh mekanisme pengambilan Keputusan dalam model algoritma yang dibangun, karena setiap iterasi yang dijalankan akan mengevaluasi dan membandingkan Solusi berdasarkan nilai *fitness* yang diperoleh dari hasil perhitungan *distance matrix* tersebut.

4.3.3.2 Greedy Initial Route

Tahap ini bertujuan untuk membangun solusi awal yang digunakan sebagai titik awal eksplorasi pada pembangunan model algoritma. Dalam tugas akhir ini, solusi awal dibentuk menggunakan metode *greedy nearest neighbor*, dimana algoritma ini memilih titik terdekat dari posisi saat ini secara iteratif hingga semua titik dikunjungi. *Greedy search* merupakan pendekatan heuristik yang efisien dimana di setiap langkah, algoritma selalu memilih opsi terbaik berdasarkan penilaian local saat itu. Meskipun hasilnya tidak menjamin solusi optimal secara global, pendekatan *greedy* mampu menghasilkan Solusi awal berkualitas yang dapat mempercepat proses pencarian solusi akhir. Solusi awal yang baik juga dapat membantu mengecualikan area pencarian yang hanya mengarah pada hasil yang lebih buruk (Löffler et al., 2024).

Dalam implementasinya, Solusi hasil *greedy* menjadi dasar dalam algoritma *simulated annealing* untuk dieksplor lebih lanjut secara global. Berikut merupakan kode fungsi generate_greedy_initial() yang digunakan dalam proses ini.

```
def generate_greedy_initial_route(distance_matrix):
    n = len(distance_matrix)
    unvisited = set(range(1, n))
    route = [0]

    current = 0
    while unvisited:
        next_node = min(unvisited, key=lambda x: distance_matrix[current][x])
        route.append(next_node)
        unvisited.remove(next_node)
        current = next_node

    return route
```

Kode 4.10 Pembuatan Rute Awal dengan *Greedy Nearest Neighbor*

4.3.3.3 Two-Opt Method

Two-Opt adalah metode *local search* yang digunakan dalam tugas akhir ini untuk mengurangi total jarak tempuh UAV secara iteratif. Meode ini bekerja dengan mengevaluasi semua pasangan sisi pada rute dan memodifikasi urutan antar simpul apabila ditemukan sebuah kombinasi yang lebih baik. Algoritma *two-opt* secara bertahap memperbaiki total jarak hingga mencapai nilai minimum, dengan cara mengganti pasangan sisi (i, i+1) dan (j, j+1) menjadi (i, j) dan (i+1, j+1) sehingga menghilangkan perpotongan lintasan dan meminimalkan nilai perjalanan secara keseluruhan (Chandomi-Castellanos et al., 2022). Proses ini diulang hingga tidak ditemukan lagi kombinasi sisi yang memberikan perbaikan berarti terhadap solusi saat ini.

Dalam implementasinya, metode *two-opt* diintegrasikan ke dalam algoritma *simulated annealing* sebagai strategi *local refinement* secara periodik setiap sejumlah iterasi tertentu. Fungsi two_opt() digunakan untuk memoles solusi yang sebelumnya dihasilkan oleh proses

eksplorasi acak dari fungsi get_neighbor(). Dengan demikian, *two-opt* berperan sebagai komponen eksploitasi dalam algoritma yang berfungsi untuk memperkuat proses pencarian global dari *simulated annealing*. Berikut merupakan kode fungsi yang digunakan dalam proses tersebut.

```
def two_opt(route, distance_matrix, max_iter=1000):
    best = route
    improved = True
    while improved and max_iter > 0:
        improved = False
        for i in range(1, len(route) - 2):
            for j in range(i + 1, len(route) - 1):
                if j - i == 1:
                    continue
                new_route = best[:i] + best[i:j][::-1] + best[j:]
                if fitness(new_route, distance_matrix) < fitness(best,
distance_matrix):
                    best = new_route
                    improved = True
        max_iter -= 1
    return best
```

Kode 4.11 Fungsi Algoritma *Two-opt*

4.3.3.4 Simulated Annealing

Simulated Annealing (SA) merupakan algoritma metaheuristik yang diadopsi dari prinsip pendinginan logam. Dalam tugas akhir ini, SA digunakan untuk mengoptimalkan urutan kunjungan UAV terhadap titik-titik *centroid* dengan meminimalkan fungsi jarak total. Setelah proses awal dilakukan menggunakan *greedy*, algoritma SA mulai berjalan secara iteratif. Di setiap iterasi, SA membangkitkan solusi tetangga menggunakan fungsi get_neighbor(), dimana fungsi tersebut melakukan modifikasi rute dengan salah satu dari tiga skema perubahan yang dibuat, yaitu *reverse*, *swap*, dan *insert*. Modifikasi ini menciptakan variasi struktur struktur Solusi yang variatif dan memungkinkan eksplorasi menuju wilayah Solusi baru. Selain itu, di setiap beberapa iterasi tertentu selama algoritma berjalan, algoritma mengaktifkan proses *local search* dengan menggunakan metode *two-opt*. Integrasi *local search* ini berperan sebagai startegi *refinement* untuk memperbaiki Solusi yang sedang dievaluasi. Proses ini berlangsung hingga suhu T turun di bawah nilai ambang batas (*stopping temperature*) atau jumlah iterasi maksimum tercapai.

Berikut merupakan kode fungsi yang digunakan dalam proses tersebut.

```
def get_neighbor(route):
    new_route = route[:]
    i, j = sorted(np.random.choice(range(1, len(route)), size=2, replace=False))
    move_type = np.random.choice(["reverse", "swap", "insert"])

    if move_type == "reverse":
        new_route[i:j+1] = reversed(new_route[i:j+1])
    elif move_type == "swap":
        new_route[i], new_route[j] = new_route[j], new_route[i]
```

```
    elif move_type == "insert":
        node = new_route.pop(i)
        new_route.insert(j, node)

    return new_route

def simulated_annealing(
    distance_matrix,
    initial_route,
    initial_temp=100,
    cooling_rate=0.99,
    stopping_temp=1e-4,
    max_iter=2000,
    interval_log=50,
    n_iter_2opt=10,
    local_max_iter=30
):
    current_route = initial_route[:]
    current_cost = fitness(current_route, distance_matrix)
    best_route = current_route[:]
    best_cost = current_cost
    temperature = initial_temp
    cost_log = []

    iteration = 0
    while temperature > stopping_temp and iteration < max_iter:
        neighbor_route = get_neighbor(current_route)
        # Setiap n iterasi → refine dengan 2-Opt
        if iteration % n_iter_2opt == 0:
            neighbor_route = two_opt(neighbor_route, distance_matrix,
max_iter=local_max_iter)

        neighbor_cost = fitness(neighbor_route, distance_matrix)

        # Acceptance
        if neighbor_cost < current_cost or np.random.rand() <
np.exp((current_cost - neighbor_cost) / temperature):
            current_route = neighbor_route
            current_cost = neighbor_cost
            if current_cost < best_cost:
                best_route = current_route[:]
                best_cost = current_cost

        cost_log.append(current_cost)


        if iteration % interval_log == 0:
            print(f"Iterasi {iteration:4d} | Cost: {current_cost:.2f} | Best:
{best_cost:.2f} | Suhu: {temperature:.2f}")
```

```
        # Cooling
        temperature *= cooling_rate
        iteration += 1

    print(f"Final Iterasi {iteration} | Best Cost: {best_cost:.2f}")
    return best_route, best_cost, cost_log
```

Kode 4.12 Implementasi Algoritma Hybrid Simulated Annealing

4.3.4 Desain Algoritma Pembanding (Google OR-Tools)

Sebagai algoritma pembanding dalam tugas akhir ini, digunakan pendekatan berbasis Google OR-Tools, yaitu sebuah pustaka *open source* yang umum digunakan dalam permasalahan VRP. Dalam implementasinya, OR-Tools digunakan untuk menemukan lintasan UAV dengan total jarak minimal, dimana dalam tugas akhir ini menjadi sebuah *baseline* dalam mengevaluasi kualitas solusi algoritma *hybrid simulated annealing* yang diimplementasikan. Proses diawali dengan membentuk matriks jarak melalui antar *centroid* hasil klasterisasi yang kemudian dikonversi ke dalam format *list* untuk kompatibilitas dengan *callback* OR-Tools. Selanjutnya dibangun model optimasi menggunakan RoutingIndexManager dan RoutingModel dengan depot ditetapkan sebagai indeks 0 (titik awal).

Model optimasi ini diatur dengan menggunakan strategi PATH_CHEAPEST_ARC yang membentuk solusi awal secara cepat, lalu solusi akan diperbaiki melalui metaheuristik *local search* (GUIDED_LOCAL_SEARCH) yang bertujuan untuk meminimalkan total jarak lintasan. Batas waktu komputasi pada algoritma dapat diatur pada search_params.time_limit.seconds agar algoritma dapat menemukan solusi terbaik dalam waktu yang wajar dan setara dengan hasil eksperimen algoritma *hybrid* SA. Jika solusi ditemukan, lintasan optimal dan jarak rute akan dihitung berdasarkan fungsi *distance matrix* yang didefinisikan sebelumnya. Berikut kode fungsi yang dibuat dalam proses ini.

```
def solve_tsp_closed(coords):
    dist_mat = cdist(coords, coords).tolist()
    n = len(coords)
    depot = 0

    manager = pywrapcp.RoutingIndexManager(n, 1, depot)
    routing = pywrapcp.RoutingModel(manager)

    def distance_callback(from_idx, to_idx):
        return
int(dist_mat[manager.IndexToNode(from_idx)][manager.IndexToNode(to_idx)] *
1000)

    transit_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_index)

    search_params = pywrapcp.DefaultRoutingSearchParameters()
    search_params.first_solution_strategy =
routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC
```

```
    search_params.local_search_metaheuristic =
routing_enums_pb2.LocalSearchMetaheuristic.GUIDED_LOCAL_SEARCH
    search_params.time_limit.seconds = 190

    solution = routing.SolveWithParameters(search_params)

    if solution:
        index = routing.Start(0)
        route = []
        while not routing.IsEnd(index):
            route.append(manager.IndexToNode(index))
            index = solution.Value(routing.NextVar(index))
        route.append(route[0])
        total_cost = sum(dist_mat[route[i]][route[i+1]] for i in
range(len(route)-1))
        return route, total_cost
    else:
        return None, None
```

Kode 4.13 Fungsi Algoritma OR-Tools

4.4 Hasil Uji Coba dan Pembahasan

Pada Subbab ini akan dijelaskan hasil uji coba sekaligus analisis hasil dari setiap tahapan yang telah dilakukan, baik itu dari tahapan klasterisasi hingga tahap implementasi algoritma *simulated annealing* pada dataset.

4.4.1 Klasterisasi Data

Bagian ini membahas hasil dari proses klasterisasi yang dilakukan terhadap area *stressed region* pada lahan pertanian. Klasterisasi dilakukan dengan menggunakan metode *lloyd algorithm* (K-Means) dengan pendekatan satu tingkat, dimana jumlah klaster optimal pada masing-masing *stressed region* diperoleh melalui proses *grid search*. *Grid search* dilakukan dengan mengevaluasi rasio cakupan area (*coverage ratio*) dan rasio area tumpang tindih (*overlap ratio*) yang mempu memenuhi target *coverage* minimal 97% dan *overlap* tidak melebihi 20%.

Berikut tabel yang menyajikan hasil jumlah klaster yang diperoleh melalui *grid search* pada setiap area *stressed region*.

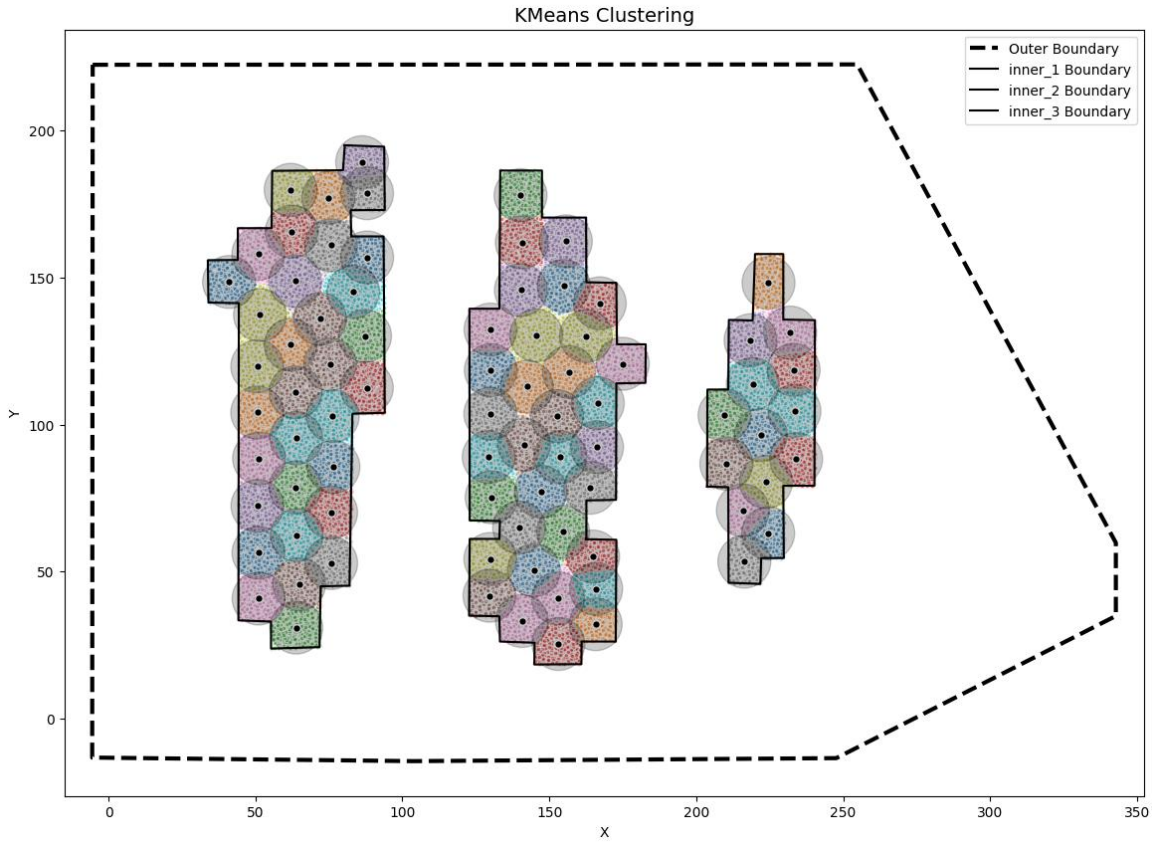
Tabel 4.4 Jumlah Klaster dan Subklaster per-*Stressed Region*

No	Area	Jumlah Klaster
1	Inner_1	33
2	Inner_2	34
3	Inner_3	14

Pada hasil tabel 4.4, terlihat bahwa jumlah klaster yang terbentuk pada setiap area *stressed region* berbeda dimana inner_3 hanya memiliki 14 klaster, sedangkan inner_1 dan inner_2 masing-masing memiliki 33 dan 34 klaster. Dari hasil klasterisasi 3 area *inner*, diperoleh ditribusi jumlah titik *sampling* yang tidak merata, dimana *inner* 1 dan 2 memiliki titik lebih banyak dibanding *inner* 3. Hal ini terjadi karena area *stressed region* pada ladang memang lebih

dominan terdistribusi pada dua wilayah utama, sedangkan *inner* 3 hanya mencakup area kecil di bagian kanan lahan. Hal ini sekaligus menunjukkan distribusi stres tanaman pada lahan yang tidak merata, serta dapat dijelaskan dengan distribusi luas area masing-masing *stressed region*, dimana inner_3 memang memiliki luas area lebih kecil sehingga jumlah kebutuhan *centroid* yang diperlukan lebih sedikit untuk mencapai target *coverage* dan *overlap* yang diperlukan. Dengan demikian, distribusi jumlah titik *centroid* pada setiap klaster telah sesuai dengan proporsi luas wilayah masing-masing *stressed region*.

Setelah klasterisasi dilakukan, hasilnya divisualisasikan dalam bentuk pemetaan spasial menggunakan diagram titik sebar dan anotasi *centroid* pada masing-masing subklaster. Visualisasi ini memberikan gambaran menyeluruh mengenai persebaran spasial klaster yang terbentuk berdasarkan jumlah yang telah diperoleh dari hasil *grid search* sebelumnya. Berikut hasil klasterisasi yang terbentuk dari ketiga *inner region* ditampilkan di bawah ini.



Gambar 4.1 Hasil Visualisasi Klasterisasi Menggunakan *Lloyd Algorithm*

Pada Gambar 4.1 terlihat bahwa hasil klasterisasi divisualisasikan menggunakan diagram Voronoi untuk memetakan wilayah cakupan masing-masing klaster pada area pertanian. Setiap poligon Voronoi memiliki warna berbeda yang merepresentasikan satu wilayah klaster sehingga memudahkan identifikasi spasial antar wilayah semprot UAV. Diagram Voronoi digunakan untuk menunjukkan kedekatan spasial antara titik semprot (*centroid*) dan area di sekitarnya, di mana setiap wilayah hanya diasosiasikan dengan satu *centroid* terdekat. Titik-titik hitam di tengah poligon menunjukkan posisi *centroid*, sementara lingkaran abu-abu dengan radius 9 meter menggambarkan jangkauan semprotan UAV yang digunakan untuk perhitungan

area cakupan. Representasi ini memberikan gambaran intuitif mengenai distribusi spasial UAV dan membantu dalam mengidentifikasi area yang mengalami tumpang tindih. Jumlah klaster dan sebarannya pada Gambar 4.1 sesuai dengan konfigurasi pada Tabel 4.4, sedangkan garis putus-putus dan padat menandai batas fisik lahan pertanian yang digunakan dalam analisis.

Hasil klasterisasi ini tidak hanya memperlihatkan sebaran spasial secara visual, tetapi juga menjadi dasar evaluasi efektivitas distribusi. Evaluasi dilakukan terhadap dua metrik utama, yaitu *coverage* dan *overlap*. *Coverage* mengukur persentase area *stressed region* yang tercakup oleh semprotan UAV, sedangkan *overlap* menunjukkan area yang mengalami tumpang tindih semprotan dari dua atau lebih klaster. Nilai kedua metrik dihitung menggunakan *overlay* spasial antara lingkaran semprotan dan luas stressed region. Dengan demikian, penggunaan diagram Voronoi dalam visualisasi ini tidak hanya menyajikan peta spasial distribusi UAV, tetapi juga mendukung penilaian kuantitatif terhadap efisiensi klasterisasi. Hasil lengkap perhitungan coverage dan overlap disajikan pada Tabel 4.5.

Tabel 4.5 Perhitungan Luas Persentase *Coverage* dan *Overlap Area* Tiap *Stressed Region*

Area	Luas area (m^2)	Luas area yang tercakup (m^2)	Persentase *coverage* (%)	Luas *overlap* (m^2)	Persentase *overlap* (%)
inner_1	6764.79	6587.93	97.39	1233.2	18.23
inner_2	6852.79	6663	97.23	1470.42	21.46
inner_3	2671.47	2595.39	97.15	547.97	20.51

Berdasarkan hasil pada tabel 4.5, terlihat bahwa tingkat *coverage* UAV pada area *stressed region* telah mencapai lebih dari 97% dengan persentase *coverage* masing-masing daerah sebesar 97.39% untuk inner_1, 97.23% untuk inner_2, dan 97.15% untuk inner_3. Hal ini menunjukkan bahwa radius semprot UAV sebesar 9 meter yang diberikan pada setiap *centroid* mampu menjangkau hampir seluruh area target pada masing-masing wilayah. Selain cakupan, nilai *overlap* yang diperoleh di setiap *stressed region* secara berturut-turut adalah 18.23%, 21.46%, dan 20.51%. Nilai ini mengindikasikan bahwa sebagian wilayah, terdapat beberapa bagian area yang terkena semprotan UAV sebanyak lebih dari satu kali akibat irisan radius semprot antar *centroid*.

Untuk memperoleh gambaran yang lebih menyeluruh mengenai distribusi kedua metrik secara global, dilakukan perhitungan total nilai *coverage ratio* dan *overlap ratio* terhadap luas total wilayah yang dianalisis. Berikut hasilnya disajikan dalam tabel sebagai berikut.

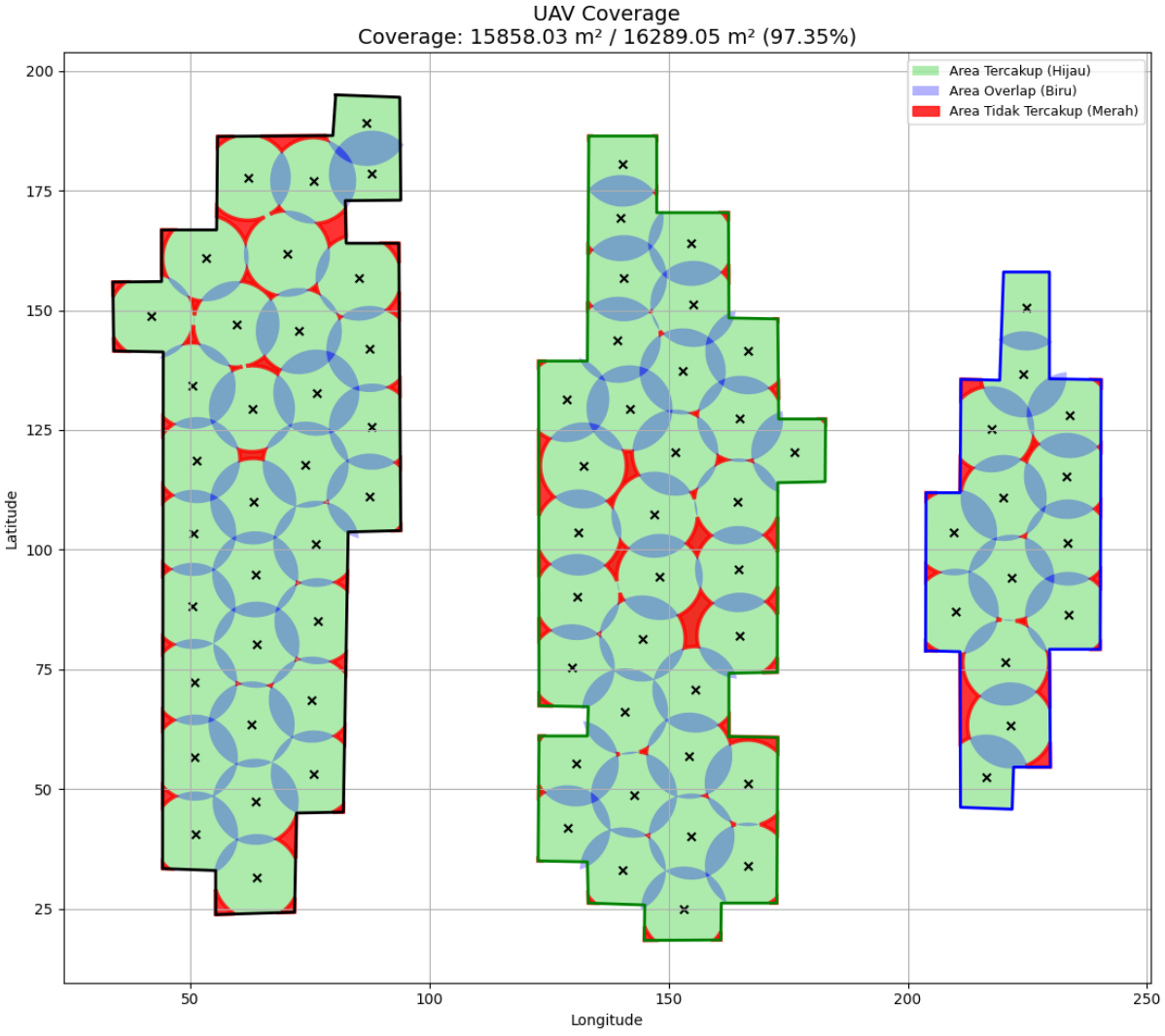
Tabel 4.6 Nilai Persentase *Coverage* dan *Overlap Ratio* Secara Global

Metrik	Total nilai (m^2)	Luas area (m^2)	Persentase (%)
Coverage area	15858.32	16289.05	97.35
Overlap area	3251.59	16289.05	19.96

Dari tabel berikut, dapat diketahui bahwa secara keseluruhan, cakupan semprot UAV telah menjangkau 97.28% dari total area *stressed region* dengan area *overlap ratio* mencapai 19.96% terhadap total area, atau setara dengan 20.52% terhadap total area yang berhasil tercakup. Dari hasil nilai metrik pada kedua tabel (tabel 4.5 dan 4.6), terlihat bahwa peningkatan *coverage* yang tinggi pada setiap *stressed region* (>97%) memang diikuti dengan nilai *overlap* yang

relatif meningkat, berada pada kisaran 18% hingga 21%. Hal ini wajar terjadi karena semakin luas area yang ingin dijangkau untuk memastikan *coverage* optimal, maka radius semprot antar *centroid* akan semakin saling beririsan, sehingga nilai *overlap* turut naik. Fenomena ini menunjukkan adanya *trade-off* langsung antara *coverage* dan *overlap*, dimana menaikkan *coverage* pasti akan menaikkan *overlap*.

Untuk memberikan ilustrasi visual yang lebih komprehensif, dibuat visualisasi distribusi *centroid* dengan area *coverage* dan *overlap* yang diperlihatkan secara keseluruhan area *stressed region*. Visualisasi ini dapat digunakan untuk mengidentifikasi, baik area yang berhasil tercakup dan area yang terkena penyemprotan lebih dari sekali. Visualisasi ditampilkan dalam gambar berikut.



Gambar 4.2 Visualisasi Distribusi *Coverage* dan *Overlap* pada *Stressed Region*

Pada Gambar 4.2 terlihat bahwa hasil klasterisasi *centroid* divisualisasikan menggunakan diagram Voronoi yang membagi area pertanian ke dalam wilayah cakupan masing-masing titik semprot. Setiap wilayah klaster ditandai dengan warna yang berbeda untuk mempermudah identifikasi spasial antar *centroid*. Selain itu, visualisasi ini dilengkapi dengan keterangan warna untuk area hasil semprotan, antara lain warna hijau merepresentasikan area yang

tercakup (*coverage*), warna merah menunjukkan area yang tidak tercakup (*uncovered*), dan warna biru menunjukkan area yang mengalami tumpang tindih semprotan (*overlap*) antar klaster. Pewarnaan ini dimaksudkan untuk memberikan gambaran intuitif terhadap distribusi efektivitas penyemprotan agrokimia pada masing-masing wilayah hasil klasterisasi, sehingga dapat dianalisis sejauh mana efisiensi spasial telah tercapai.

Jika dibandingkan dengan studi terdahulu, seperti yang dilakukan oleh Srivastava er al (2020), hasil yang diperoleh pada tugas akhir ini masih dapat dianggap sangat wajar. Dalam penelitian tersebut, metode klasterisasi menggunakan pendekatan voronoi dan algoritma juga bertujuan dalam memastikan nilai *stressed region* yang berhasil tercakup sebanyak lebih dari 97%, meskipun pada akhirnya, tetap terdapat area *overlap* yang bervariasi akibat radius semprot yang berbentuk lingkaran. Artinya pencapaian nilai *coverage ratio* sekitar 97% dengan *overlap* sekitar 20% pada tugas akhir ini bisa dibilang konsisten dengan temuan empiris sebelumnya dalam implementasi UAV untuk distribusi agrokimia pada wilayah pertanian berbentuk tidak beraturan (Srivastava et al., 2020).

Dengan demikian, hasil evaluasi ini menunjukkan bahwa proses klasterisasi yang telah dilakukan telah mampu menghasilkan distribusi *centroid* yang cukup baik dalam hal cakupan, sekaligus menjaga tingkat *overlap* dalam angka yang wajar atau tidak berlebihan. Temuan ini menjadi dasar penting yang akan digunakan pada tahap selanjutnya, yaitu optimasi lintasan UAV, agar penyemprotan dapat dilakukan dengan jalur yang lebih efisien, baik dari sisi jarak tempuh maupun waktu pengoperasiannya.

4.4.2 Optimasi Rute UAV

Pada tahap ini dilakukan proses optimasi rute UAV menggunakan algoritma *hybrid simulated annealing* (SA) yang telah dirancang. Pengujian diawali dengan mengevaluasi performa algoritma pada 7 variasi parameter utama, meliputi kombinasi nilai *initial temperature*, *cooling rate*, *max iteration*, dan *two-opt interval*. Variasi ini dirancang untuk mengobservasi pengaruh perubahan parameter terhadap kualitas solusi yang dihasilkan, baik dari sisi total jarak lintasan UAV maupun kestablian hasil pada setiap *run*.

Setelah diperoleh gambaran awal performa algoritma pada masing-masing variasi, dilakukan proses *grid search* lanjutan di dalam rentang parameter tersebut. *Grid search* ini bertujuan untuk mencari kombinasi parameter yang dapat menghasilkan total jarak lintasan terbaik. Dari proses ini, diamnil 3 konfigurasi parameter terbaik, dimana masing-masing menunjukkan potensi dalam menghasilkan lintasan dengan total jarak terendah. Dari tiga variasi parameter yang didapatkan, untuk menguji kestabilan performanya, dilakukan pengujian lanjutan dengan menjalankan algoritma sebanyak 10 kali *run* pada ketiga variasi parameter terbaik tersebut. Hal ini bertujuan untuk mengukur konsistensi algoritma *hybrid simulated annealing* dalam menghasilkan solusi lintasan UAV yang efisien, meskipun terdapat sifat *stohastic* pada proses inisalisasi dan iterasi algoritma metaheuristik.

Dengan pendekatan ini, diperoleh hasil total jarak lintasan terbaik yang dapat digunakan sebagai basis perbanidngan dengan algoritma OR-Tools pada tahap analisis komparatif berikutnya. Hasil detail masing-masing variasi, proses *grid search*, dan hasil *run* pada konfigurasi terbaik akan diuraikan pada subbab-subbab berikutnya..

4.4.2.1 Uji Coba Awal Algoritma *Hybrid Simulated Annealing*

Pada tahap ini dilakukan pengujian algoritma *hybrid simulated annealing* (SA), dimana algoritma ini tidak hanya menggunakan mekanisme standar SA, tetapi dilengkapi dengan proses *greedy initialization* serta integrasi metode *two-opt* sebagai strategi *local search*. Selain itu fungsi get_neighbor dengan variasi mutasi (*reverse*, *swap*, *insert*) digunakan untuk memperkaya proses eksplorasi solusi. Pengujian dilakukan pada tujuh jenis variasi konfigurasi parameter, meliputi pengaturan nilai suhu awal (*initial temperature*), laju pendinginan (*cooling rate*), jumlah iterasi maksimum, dan interval eksekusi *two-opt* (n_iter_2opt). Sementara itu, parameter lain, seperti *stopping temperature* dan jumlah iterasi *two-opt* diset sebesar 1e-4 dan 30 iterasi pada semua variasi parameter. Setiap variasi diuji sebanyak 10 kali *run* untuk mengevaluasi kestabilan performa algoritma terhadap perubahan parameter, serta menilai solusi dalam konteks jarak dan waktu komputasi (*runtime*). Berikut rincian parameter yang digunakan untuk setiap variasi ditampilkan pada tabel 4.7 berikut.

Tabel 4.7 Variasi Parameter Algoritma *Hybrid SImulated Annealing*

Variasi	*Initial temp*	*Cooling rate*	*Max iteration*	*Two-opt interval*
1	100	0.995	1000	10
2	200	0.99	1000	10
3	100	0.97	1000	10
4	100	0.995	2000	10
5	100	0.995	1000	5
6	500	0.999	3000	20
7	500	0.995	3000	20

Setelah itu, dilakukan uji performa algoritma berdasarkan variasi parameter yang telah ditentukan sebanyak 10 kali *run* pada setiap variasinya. Hasil uji yang diperoleh kemudian diolah ke dalam bentuk nilai rata-rata, minimum, dan standar deviasi jarak tempuh UAV, serta nilai rata-rata durasi komputasi (*runtime*) yang dihasilkan. Hasil tersebut dirangkum ke dalam satu tabel 4.8 berikut ini.

Tabel 4.8 Rangkuman Hasil Uji Coba *Hybrid* SA Pervariasi Parameter

Variasi	Jarak tempuh			Rata-rata *runtime* (detik)
	Rata-rata (m)	Minimum (m)	Standar deviasi (m)	
1	1470.53	1445.19	20.4	33.99
2	1464.77	1439.53	22.17	33.8
3	1492.04	1453.17	26.37	14.03
4	1453.91	1428.77	23.71	75.86
5	1476.39	1445.51	21.51	66.9
6	1475.52	1453.44	14.31	64.31
7	1451.36	1430.01	17.67	59.93

Berdasarkan hasil tabel 4.8, terlihat bahwa nilai rata-rata jarak tempuh yang dicapai algoritma berada pada kisaran 1451-1492 meter, dengan nilai rata-rata terendah pada variasi ke-7 sebesar 1451.36 meter, dengan nilai minimum jarak sebesar 1430.01 meter. Hal ini menunjukkan bahwa kombinasi parameter dengan nilai *initial temperature* dan jumlah iterasi yang tinggi serta *cooling rate* yang relatif lambat dapat memberikan kualitas solusi yang lebih

baik dalam meminimalkan total jarak rute UAV. Sementara itu, hasil standar deviasi yang dihasilkan memperlihatkan tingkat kestabilan bisa dianggap homogen di rentang 14 hingga 26 meter, dengan variasi ke-6 menunjukkan nilai terendah sebesar 14.31 meter, sedangkan variasi ke-3 memiliki nilai terbesar sebesar 26.37 meter dan memiliki rata-rata jarak tempuh tertinggi, yaitu 1492.04 meter. Hal ini mengindikasikan bahwa meskipun terdapat variasi kestabilan hasil antar *run*, algoritma *hybrid simulated annealing* cenderung memberikan performa yang cukup stabil pada sebagian besar konfigurasi parameter yang diuji. Nilai standar deviasi yang masih dalam rentang wajar ini mencerminkan karakteristik algoritma metaheuristik yang selalu memiliki fluktuasi hasil akibat sifat randomisasi pada proses insialisasi dan transisi solusi antar iterasi.

Dari hasil pembahasan di atas, hasil dari variasi parameter yang diperoleh, mengedepankan keseimbangan antara nilai suhu awal yang tinggi, *cooling rate* yang cukup lambat, jumlah iterasi yang besar, dan frekuensi interval *two-opt* yang lebih sering terbukti memberikan kualitas lintasan yang lebih baik. Hal ini mendukung prinsip dasar algoritma *simulated annealing* yang memerlukan harmonisasi ketiga komponen utama tersebut agar proses eksplorasi dan eksploitasi dapat berjalan dengan efektif. Temuan ini menjadi pijakan untuk tahap berikutnya, yaitu proses *grid search* yang lebih fokus pada parameter yang menghasilkan nilai terbaik, serta uji coba *multi run* (10 kali) pada varaisi tersebut agar memperoleh evaluasi performa yang lebih komprehensif.

4.4.2.2 *Grid Search* dan Multi-Run pada Parameter Terbaik

Pada tahap ini, dilakukan proses *grid search* yang secara sistematis mengombinasikan kombinasi nilai empat parameter yang digunakan pada saat uji coba awal algoritma *hybrid simulated annealing*. Hal ini dilakukan untuk memperoleh konfigurasi parameter yang mampu meminimalkan total jarak rute UAV. Secara total, terdapat 108 kombinasi parameter yang diuji, dengan variasi yang terdiri dari 3 nilai *initial temperature* (100, 200, 500), 4 nilai *cooling rate* (0.97, 0.99, 0.995, 0.999), 3 nilai *max iteration* (1000,2000,3000), dan 3 nilai *two-opt interval* (5,10,20).

Hasil dari *grid search* menunjukkan terdapat beberapa pola yang menarik. Untuk mendapatkan konteks akan hal tersebut, berikut merupakan ringkasan rata-rata *cost* hasil *grid search* berdasarkan setiap nilai parameter yang digunakan.

Tabel 4.9 Rata-rata Jarak Tempuh Berdasarkan Setiap Nilai Parameter

Parameter	Nilai	Rata-rata nilai jarak (m)
Initial temp	100	1475.43
	200	1469.3
	500	1479.07
Cooling rate	0.97	1495.46
	0.99	1465.53
	0.995	1462.99
	0.999	1470.04
Max iteration	1000	1487.79
	2000	1472.17
	3000	1463.85
Two-opt interval	5	1466.85
	10	1472

	20	1484.65

Berdasarkan tabel 4.9, terlihat pola konsisten pada hasil *grid search* dengan *hybrid simulated annealling*, dimana konfigurasi parameter dengan nilai *cooling rate* lebih lambat (0.995) dan jumlah iterasi lebih tinggi (3000) cenderung memberikan rata-rata total jarak tempuh yang lebih rendah. Parameter *cooling rate* 0.995 menghasilkan rata-rata *cost* terendah sebesar 1462.99 meter, sedangkan *cooling rate* 0.97 yang realtif cepat justru menunjukkan *cost* tertinggi sebesar 1495.46 meter. Begitu juga pada parameter *max iteration*, rata-rata *cost* menurun dari iterasi 1000 sebesar 1472.17 meter hingga iterasi 3000 sebesar 1463.85 meter. Sementara itu, pada parameter *initial temperature*, nilai 200 memberikan rata-rata nilai yang lebih rendah dibanding *initial temperature* yang lebih kecil maupun lebih besar. Hal ini menunjukkan bahwa nilai *initial temperature* yang terlalu tinggi bukan berarti secara otomatis akan mendapatkan hasil yang lebih baik. Hal tersebut perlu diimbangi dengan *cooling rate* dan jumlah iterasi yang sesuai. Untuk parameter *two-opt interval*, terlihat bahwa semakin sering dilakukan *local search*, hasil *cost* cenderung lebih rendah, dimana interval 5 memiliki rata-rata sebesar 1466.85 meter. Temuan ini memperkuat pentingnya keseimbangan parameter algoritma agar proses pencarian solusi dapat berjalan optimal, dari segi eksplorasi dan eksploitasi.

Dari keseluruhan hasil *grid search*, diperoleh tiga konfigurasi parameter terbaik yang akan difokuskan pada tahap uji coba *multi-run* selanjutnya. Variasi ini dipilih berdasarkan performa total jarak lintasan Uav terendah pada hasil *grid search* sebelumnya. Berikut tabel yang menyajikan informasi ketiga konfigurasi tersebut.

Tabel 4.10 Tiga Konfigurasi Parameter Terbaik Hasil Grid Search untuk Uji Multi-run

No	*Initial temp*	Cooling rate	Max iter	2-opt interval
1	500	0.995	3000	5
2	200	0.995	3000	5
3	500	0.99	3000	10

Hasil pengujian *multi-run* pada ketiga knfigurasi parameter terbaik yang diperoleh dari *grid search* dirangkum pada tabel 4.11 berikut. Tabel ini memperlihatkan nilai rata-rata, minimum, dan standar total jarak rute UAV, serta rata-rata waktu *runtime* komputasi yang diperlukan untuk menyelesaikan proses optimasi pada masing-masing konfigurasi.

Tabel 4.11 Hasil Rangkuman Hasil Uji Coba pada Tiga Konfigurasi Parameter Terbaik

Variasi	Jarak tempuh			Rata-rata *runtime* (detik)
	Rata-rata (m)	Minimum (m)	Standar deviasi (m)	
1	1448.66	1428.51	19.03	208.26
2	1439.16	1425.22	10.32	175.05
3	1456.33	1429.17	21.94	55.63

Berdasarkan hasil pada tabel 4.11, terlhat bahwa variasi ke-2 tidak hanya menghasilkan rata-rata jarak tempuh paling rendah sebesar 1439.16, tetapi juga menunjukkan kestabilan performa terbaik dengan standar deviasi hanya 10.32 meter. Sementara itu, hasil rute terpendek juga berhasil diperoleh dari konfigurasi yang sama, yaitu sebesar 1425.22 meter. Selain itu,

hasil nilai minimum dari ketiga variasi yang dihasilkan berada di kisaran 1425 hingga 1429 meter, dimana nilai tersebut secara signifikan lebih baik dibandingkan dengan hasil minimum pada uji coba sebelumnya. Hal ini menunjukkan bahwa proses *grid search* yang dilakukan terbukti mampu mengarahkan pencarian ke konfigurasi parameter yang lebih baik dalam menghasilkan solusi rute terbaik yang lebih pendek dan konsisten antar *run*.

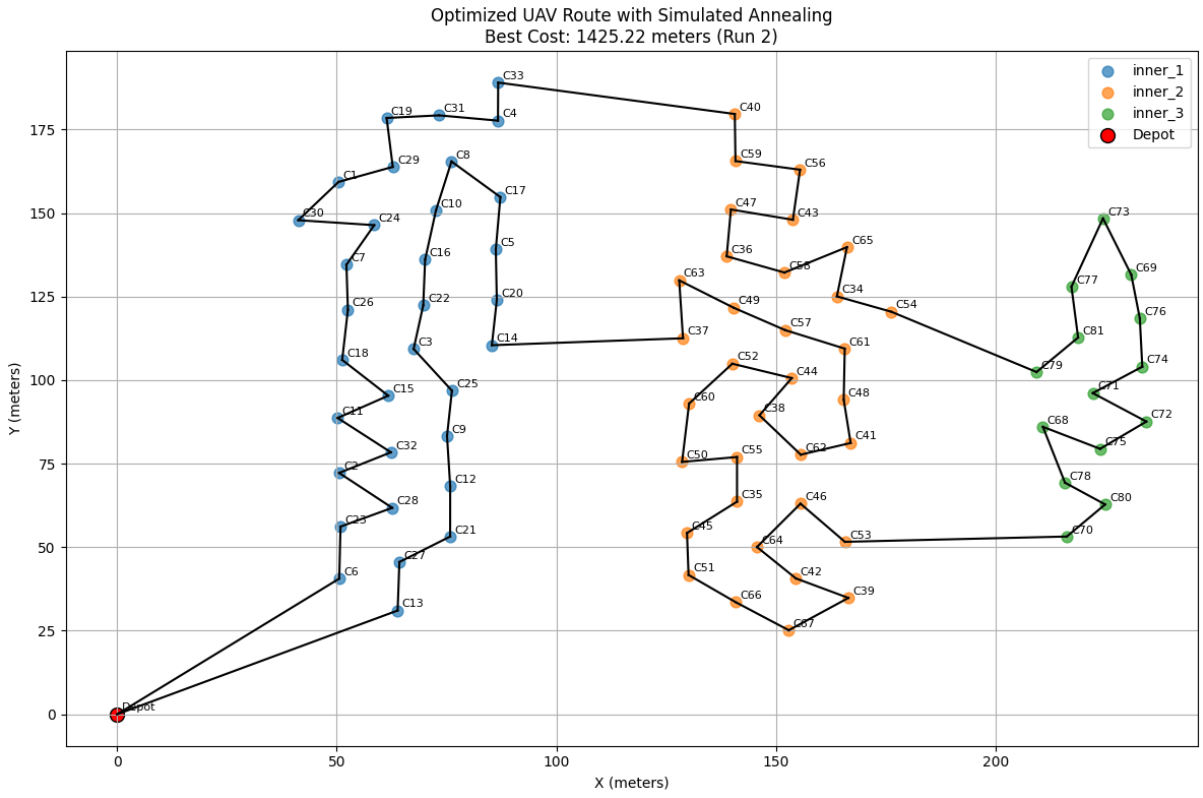
4.4.2.3 Visualisasi Rute Terbaik dari Seluruh Percobaan

Tahap ini akan menampilkan hasil rute terbaik yang diperoleh dari seluruh proses ujicoba optimasi lintasan UAV dengan algoritma *hubrid simulated annealing*. Berdasarkan seluruh hasli yang diperoleh, besar rute terpendek yang didapat adalah 1425.22 meter dengan waktu komputasi 186.08 detik. Berdasarkan kecepatan rata-rata UAV sebesar 7 m/s, diperoleh estimasi waktu penerbangan *drone* untuk menyelesaikan rute selama 203.6 detik atau 3 menit 24 detik. Berikut tabel yang merangkum konfigurasi parameter yang menghasilkan lintasan tersebut.

Tabel 4.12 Hasil Optimasi Terbaik Algoritma *Hybrid Simulated Annealing*

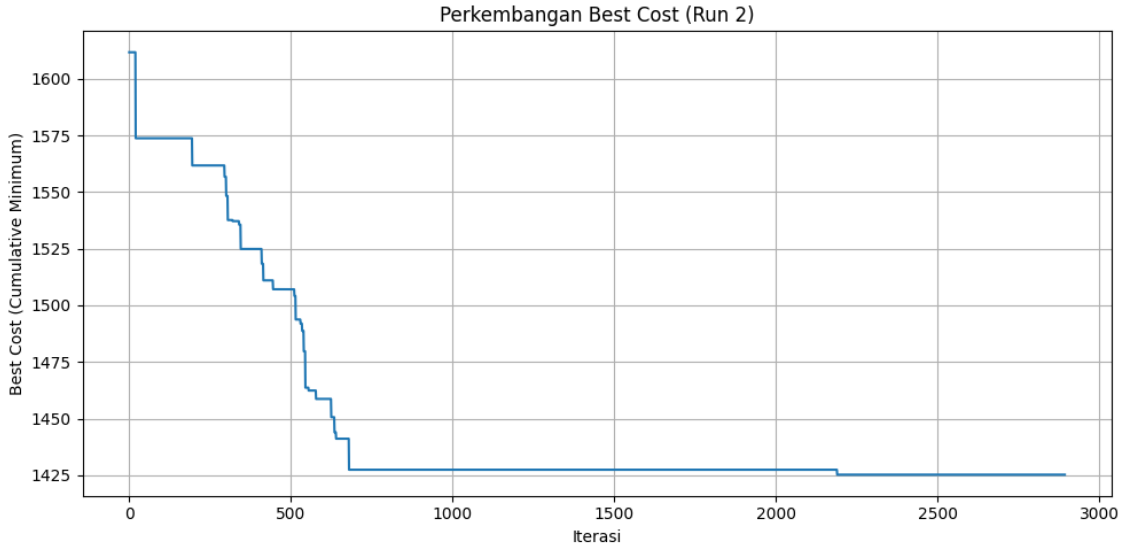
Initial temp	*Cooling rate*	*Max iter*	*2-opt interval*	Total jarak	Waktu *runtime*	Waktu penerbangan
200	0.995	3000	5	1425.22 m	186.08 detik	203.6 detik

Untuk memberikan gambaran spasial terhadap hasil tersebut, ditampilkan gambar dibawah ini hasil visualisasi rute UAV yang memperlihatkan rute yang terbentuk dari hasil optimasi yang diperoleh oleh algoritma.



Gambar 4.3 Visualisasi Jalur UAV pada Rute Terpendek oleh *Hybrid* SA

Selain visualisasi jalur rute, berikut ditampilkan grafik konvergensi nilai *best cost* yang dihasilkan selama iterasi berlangsung pada hasil *run* terbaik tersebut. Hal ini menegaskan bahwa algoritma berhasil menjalankan proses pendinginan dan transisi solusi secara efektif dalam menjalankan algoritma.



Gambar 4.4 Grafik Konvergensi Nilai *Cost* pada Run Terbaik

4.4.2.4 Uji Coba Algoritma dengan OR-Tools

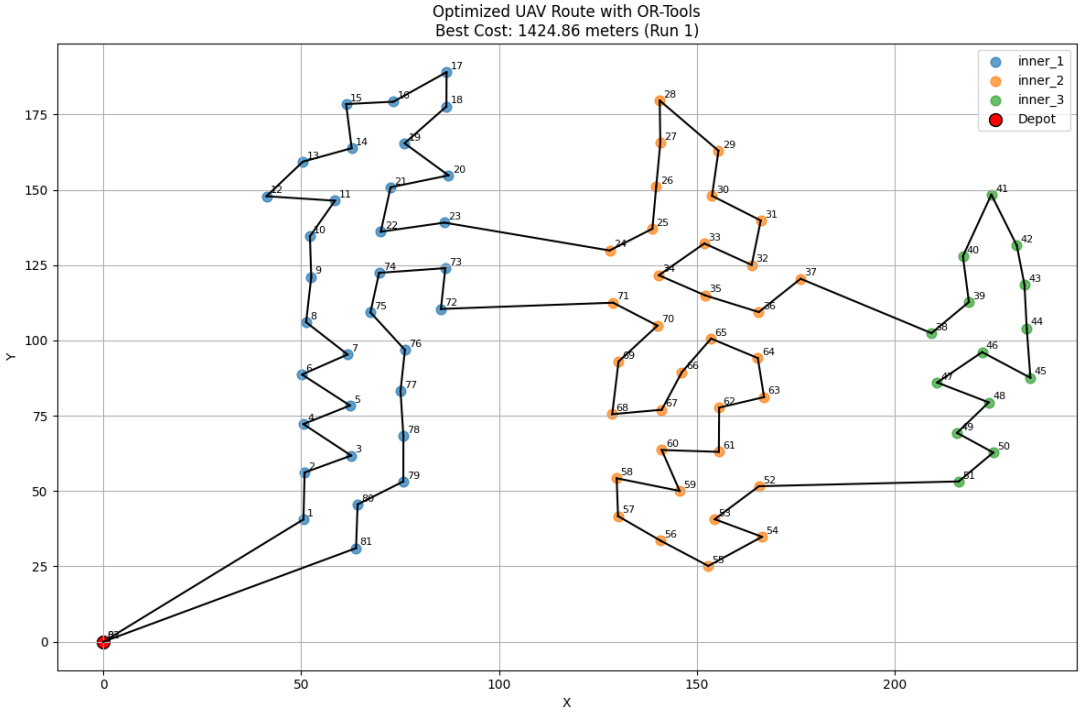
Dalam tahap ini, dilakukan optimasi rute UAV menggunakan Google OR-Tools sebagai pustaka *open source* yang umum digunakan untuk menyelesaikan masalah *Vehicle Routing Problem* (VRP). OR-Tools dijalankan pada data *centroid* hasil klasterisasi yang sama, sehingga dapat berfungsi sebagai *baseline* dalam mengevaluasi kualitas rute yang dihasilkan oleh algoritma *hybrid* SA maupun SA biasa. Untuk menjaga kesetaraan kondisi pengujian dan memastikan perbandingan yang adil, total waktu *runtime* pada algoritma OR-Tools ini diseragamkan dengan waktu rata-rata *runtime* yang digunakan pada konfigurasi saat mendapatkan hasil jarak terpendek dari algoritma *hybrid* SA, yaitu selama 186 detik. Sama seperti uji coba sebelumnya, optimasi rute juga akan dijalankan sebanyak 10 kali *run* independen pada dataset *centroid* yang sama. Berikut hasil yang diperoleh dirangkum dalam tabel di bawah ini sebagai berikut.

Tabel 4.13 Rekap Statistik Hasil Optimasi OR-Tools

Jarak tempuh			Rata-rata waktu *runtime* (detik)
Rata-rata (m)	Minimum (m)	Standar deviasi (m)	
1424.86	1424.86	0	186.01

Berdasarkan hasil rangkuman uji coba pada tabel 4.10, dapat dillihat bahwa algoritma OR-Tools menghasilkan total jarak tempuh rute UAV dengan nilai rata-rata sebesar 1677.17 meter yang konsisten di setiap *run*-nya sehingga menghasilkan nilai standar deviasi nol. Hasil ini menunjukkan bahwa algoritma ini memiliki sifat konsisten pada konfigurasi masalah pada parameter yang digunakan sehingga mampu memberikan solusi yang stabil di setiap eksekusinya, meskipun sudah dijalankan secara berulang dalam waktu komputasi yang telah

ditentukan selama 164 detik. Berikut visualisasi hasil lintasan rute yang terbentuk dari algoritma OR-Tools ditampilkan pada gambar 4.8 berikut ini.



Gambar 4.5 Hasil Visualisasi Rute UAV Terpendek Hasil OR-Tools

Untuk melengkapi hasil visualisasi rute, berikut disajikan statistik hasil terbaik yang diperoleh dari algoritma OR-Tools, meliputi total jarak tempuh rute UAV, waktu komputasi, serta estimasi waktu penerbangan UAV berdasarkan kecepatan rata-rata. Informasi ini ditampilkan pada tabel 4.11 sebagai tolok ukur langsung dalam mengevaluasi performa algoritma *hybrid* SA pada tahap perbandingan selanjutnya

Tabel 4.14 Hasil Optimasi Rute Terbaik OR-Tools

Metode	Jarak terpendek (m)	Waktu komputasi (detik)	Estimasi waktu penerbangan (detik)
OR-Tools	1424.86	186	203.55

4.4.2.5 Uji Coba Algoritma dengan *Ant Colony Optimization* (ACO)

Dalam tahap ini, dilakukan optimasi rute UAV menggunakan algoritma *Ant Colony Optimization* (ACO) yang diimplementasikan dalam bentuk *hybrid* hampir serupa dengan pendekatan *hybrid* SA yang dibentuk sebelumnya. Berbeda dengan uji coba algoritma utama dalam tugas akhir ini, dimana *hybrid* SA melalui proses *grid search* untuk menemukan konfigurasi parameter terbaik,pada ACO ini tidak dilakukan proses *tuning* parameter secara luas. Algoritma ini dijalankan dengan menggunakan satu set parameter tetap sehingga dapat berfungsi sebagai pembanding tambahan untuk mengevaluasi performa rute UAV yang dihasilkan. Parameter yang digunakan dalam ACO ditampilkan pada tabel berikut.

Tabel 4.15 Parameter pada Algoritma ACO

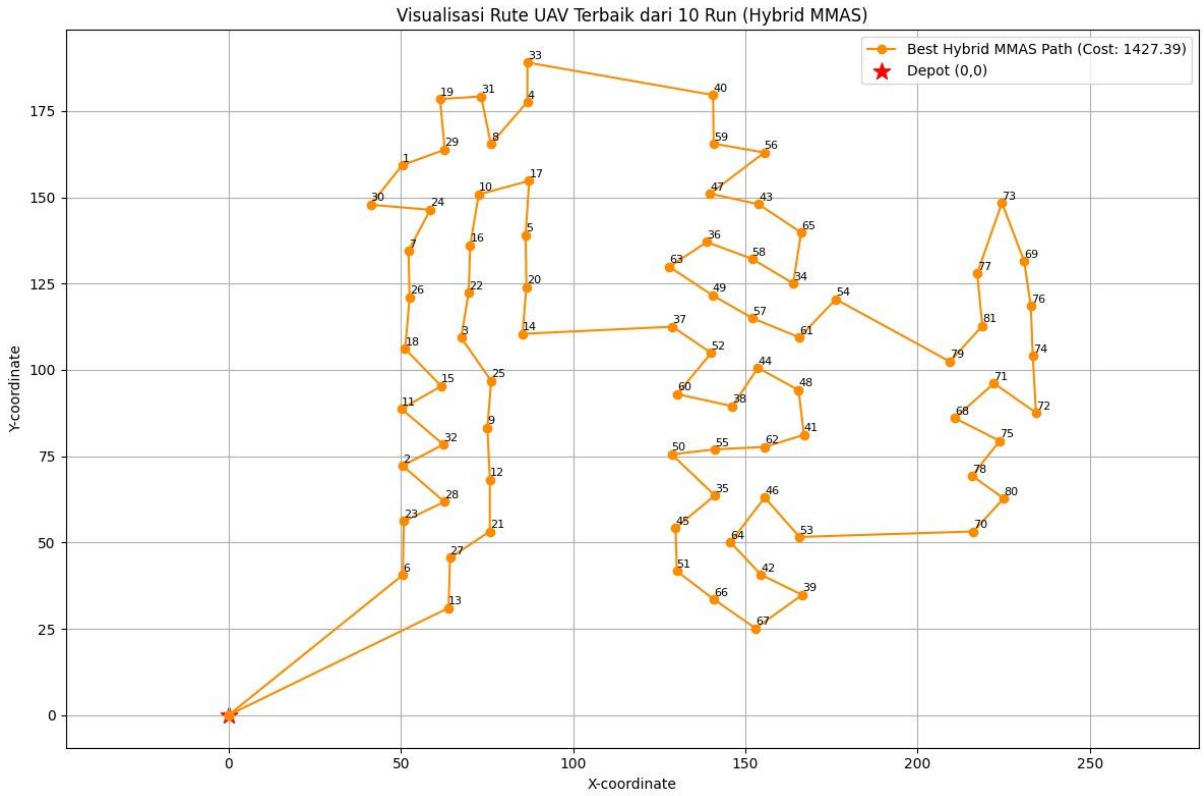
Parameter	Nilai
Alpha (α)	1
Beta (β)	7
Rho (ρ)	0.7
NUM_ANT	100
NUM_ITER	1000

Untuk menjaga kesetaraan kondisi pengujian dengan algoritma *hybrid* SA maupu OR-Tools, algoritma ACO ini dijalankan pada data *centroid* klasterisasi yang sama, serta dilakukan pengujian sebanyak 10 kali *run* secara independen. Hasil rangkuman dari 10 *run* yang diperoleh ditampilkan pada tabel berikut.

Tabel 4.16 Rekap Statistik Hasil Optimasi ACO

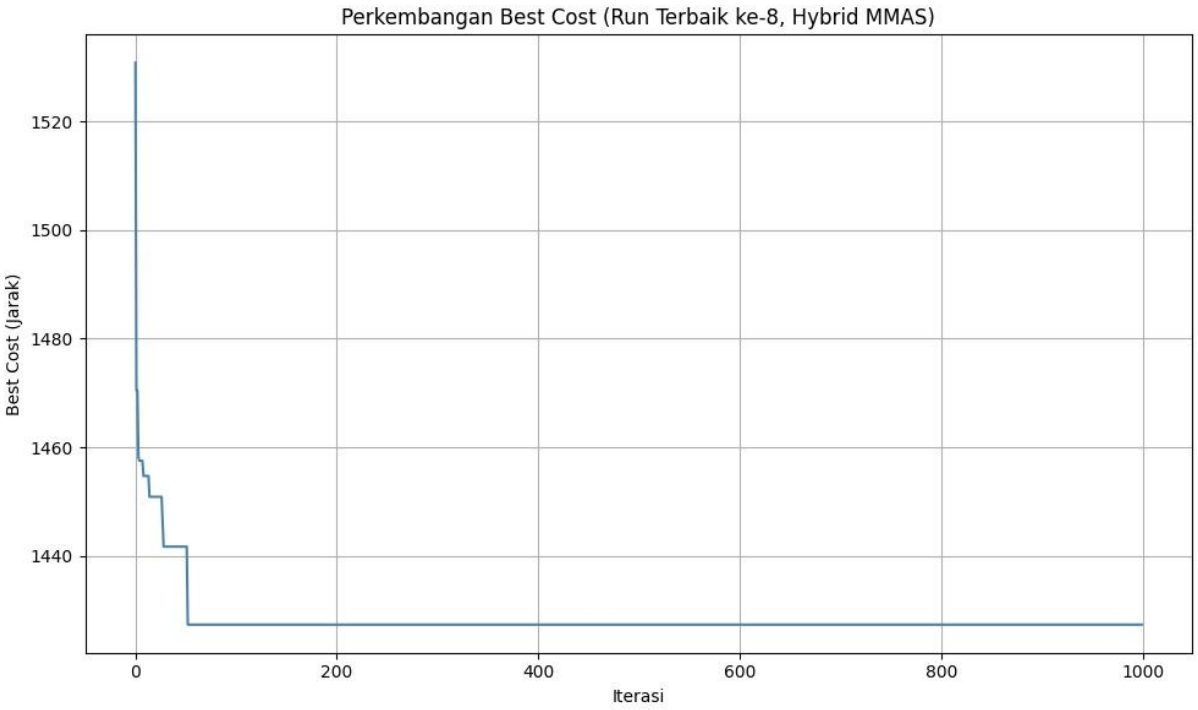
Jarak tempuh			Rata-rata waktu *runtime* (detik)
Rata-rata (m)	Minimum (m)	Standar deviasi (m)	
1449.47	1427.39	12.23	197.81

Berdasarkan tabel 4.16, dapat dilihat bahwa algoritma ACO menghasilkan rata-rata jarak rute sebesar 1449.47 meter, dengan standar deviasi 12.23 meter, dimana angka tersebut menunjukkan adanya variasi hasil antar *multi-run* yang dilakukan. Berikut visualisasi lintasan rute terbaik dan grafik konvergensi *best cost* yang dihasilkan di setiap iterasinya.



Gambar 4.6 Hasil Visualisasi Rute UAV Terpendek Hasil ACO

Berikut ditampilkan grafik konvergensi nilai *best cost* yang dihasilkan selama proses optimasi berlangsung pada hasil *run* terbaik yang diperoleh.



Gambar 4.7 Grafik Konvergensi *Best Cost* Pada ACO

Sebagai pelengkap visualisasi, berikut ditampilkan statistik hasil terbaik yang diperoleh dari algoritma ACO, meliputi total jarak tempuh terpendek, waktu komputasi, serta estimasi waktu penerbangannya. Informasi ini disajikan pada tabel 4.17, sebagai bahan perbandingan tambahan terhadap hasil algoritma *hybrid* SA dan OR-Tools pada tahap analisis selanjutnya.

Tabel 4.17 Hasil Optimasi Rute Terbaik ACO

Metode	Jarak terpendek (m)	Waktu komputasi (detik)	Estimasi waktu penerbangan (detik)
Ant Colony Optimization	1427.39	150.07	203.91

4.4.3 Perbandingan Kinerja Algoritma

Pada tahap ini dilakukan analisis komparatif terhadap kinerja pendekatan-pendekatan algoritma yang telah diuji sebelumnya untuk melihat sejauh mana efektivitas setiap metode dalam mengoptimasi rute UAV pada area *stressed region*, mulai dari *hybrid Simulated Annealing* (SA), Google OR-Tools, serta *Ant Colony Optimization* (ACO) sebagai pembanding tambahan. Perbandingan difokuskan pada tiga indikator utama, yaitu total jarak rute terpendek yang diperoleh, waktu komputasi, serta estimasi waktu penerbangan UAV berdasarkan hasil jarak tersebut.

Pada proses optimasi dengan algoritma *hybrid* SA, total jarak terpendek diperoleh pada konfigurasi variasi yang didapatkan dari hasil *grid search*. Berikut hasil pengujian konfigurasi ditampilkan pada tabel berikut ini.

Tabel 4.18 Parameter dan Hasil Terbaik Algoritma *Hybrid* SA

Parameter	Nilai
Initial temperature	200
Cooling rate	0.995
Max iteration	3000
Two-opt interval	5
Local max iteration	30
Total jarak terpendek	1425.22 meter
Rata-rata waktu *runtime*	186.08 detik
Estimasi waktu penerbangan	203.6 detik

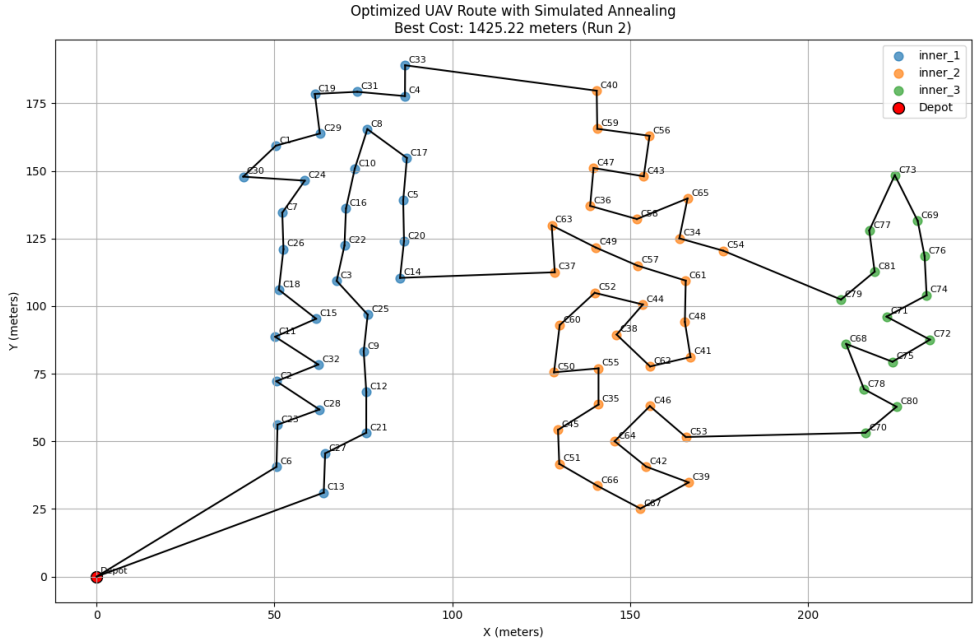
Dengan konfigurasi tersebut, *hybrid* SA dapat digunakan sebagai *baseline* utama untuk dibandingkan secara langsung terhadap performa OR-Tools maupun ACO.

4.4.3.1 Perbandingan *Hybrid* SA dengan OR-Tools

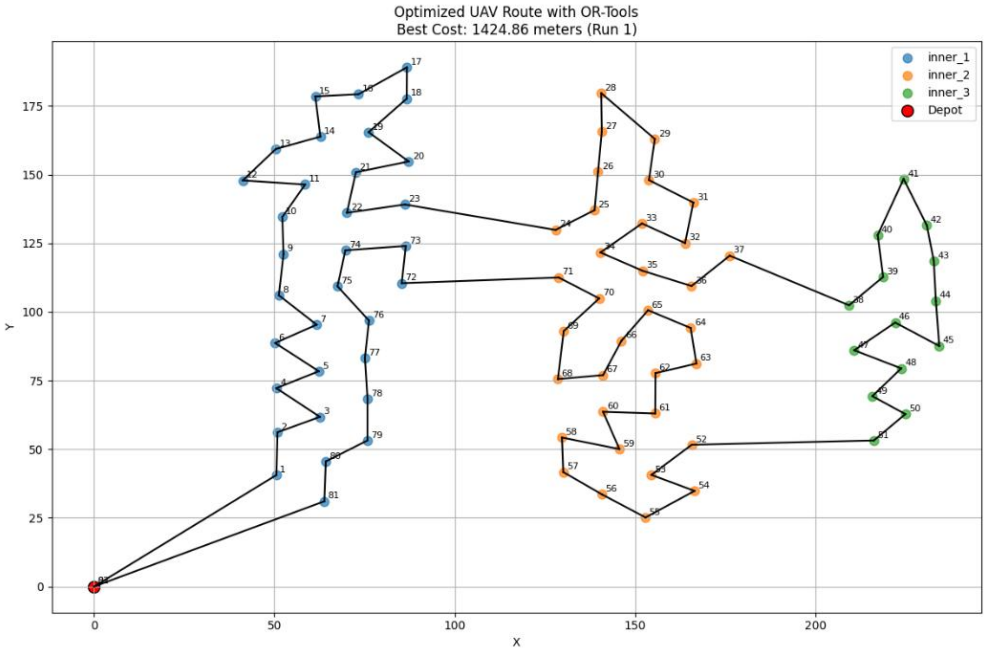
Pada subbab ini dilakukan analisis komparatif antara algoritma *hybrid* SA dan google OR-Tools dalam mengoptimasi rute UAV pada area *stressed region*. Kedua algoritma ini dijalankan pada dataset centroid hasil klasterisasi yang sama, sehingga evaluasi dapat dilakukan secara adil dengan kondisi input yang setara. Perbandingan difokuskan pada tiga aspek utama, yaitu total jarak rute terpendek yang diperoleh, waktu komputasi, serta estimasi waktu penerbangan UAV.

Pada algoritma *hybrid* SA, hasil terbaik diperoleh melalui konfigurasi parameter terbaik yang telah dirangkum sebelumnya pada Tabel 4.18. Sementara itu, untuk algoritma OR-Tools, tidak dilakukan variasi parameter karena algoritma ini menggunakan pendekatan deterministik berbasis heuristik industri. Proses optimasi dijalankan pada *dataset* yang sama dengan waktu *runtime* yang disetarakan dengan rata-rata waktu komputasi *hybrid* SA, yaitu 186 detik, sehingga menghasilkan total jarak tempuh terpendek sebesar 1424.86 meter.

Sebagai pelengkap hasil analisis komparatif, berikut ditampilkan visualisasi rute yang dihasilkan oleh kedua algoritma pada dataset *centroid* pada dua gambar berikut ini.



Gambar 4.8 Visualisasi Rute Terbaik Hasil *Hybrid* SA



Gambar 4.9 Visualisasi Rute Terbaik Hasil OR-Tools

Dari hasil visualisasi kedua lintasan, dapat diamati bahwa rute dari OR-Tools tampak lebih kompak dengan jalur yang cenderung langsung menjangkau titik-titik *centroid* tanpa banyak perpotongan lintasan ulang. Sementara itu, pada lintasan rute *hybrid* SA, meskipun rute yang dihasilkan juga berhasil menjangkau semua titik *centroid*, tetapi masih terdapat beberapa jalur yang sedikit memutar. Hal tersebut menjadi penyebab total jarak tempuh pada *hybrid* SA sedikit lebih panjang dibandingkan dengan hasil OR-Tools.

Untuk melengkapi pembahasan ini, berikut disajikan perbandingan hasil kinerja kedua algoritma secara kuantitatif, meliputi nilai jarak rute terpendek, waktu komputasi, dan estimasi

waktu penerbangan UAV yang dihitung berdasarkan total jarak tersebut. Hasil tersbut di tampilkan pada tabel di bawah ini.

Tabel 4.19 Perbandingan Hasil Kinerja Terbaik Algoritma dalam Optimasi Rute UAV

Algoritma	Jarak terpendek (m)	Waktu komputasi (detik)	Estimasi waktu penerbangan (detik)
Hybrid SA	1425.22	186.08	203.6
OR-Tools	1424.86	186	203.55

Pada tabel 4.12, terlihat bahwa selisih jarak rute UAV antara kedua algoritma sangatlah tipis, yaitu 0.36 meter atau sekitar 0.02% dari total rute OR-Tools. Hal ini menunjukkan bahwa algoritma *hybrid simulated annealing* masih berada pada rentang solusi yang kompetitif jika dibandingkan dengan solusi pada OR-Tools, meskipun algoritma ini memiliki sifat yang bergantung pada inisialisasi acak dan konfigurasi parameter. Sementara itu, OR-Tools yang bersifat deterministik dan sudah banyak digunakan dalam industri, memberikan hasil yang sangat stabil dan langsung menuju lintasan yang efektif dalam satu kali eksekusi.

Temuan ini juga mengindikasikan bahwa melalui proses *tuning* parameter yang teliti, terutama dengan pendekatan *grid search* yang telah dilakukan, *hybrid simulated annealing* dapat diarahkan untuk mencapai hasil lintasan yang tidak hanya mendekati kualitas OR-Tools, tetapi juga memiliki kelebihan fleksibilitas dalam menyesuaikan komponen *local search* maupun skema pendinginan sesuai karakteristik medan lahan. Dari perspektif kestabilan performa pada *multi-run*, algoritma *hybrid simulated annealing* memang belum dapat menyamai hasil konsistensi OR-Tools yang selalu memberikan hasil identik pada setiap eksekusi, tetapi melihat nilai standar deviasi algoritma ada dalam rentang 10 hingga 26 meter, menandakan bahwa adanya fluktuasi solusi yang masih cukup wajar untuk algoritma metaheuristik dengan proses eksplorasi acak. Variasi ini tetap dapat diterima dalam konteks distribusi agrokimia pada lahan pertanian, dimana perbedaan tersebut tidak memberikan dampak signifikan terhadap efektivitas operasi penyemprotan di lapangan.

Selain itu, dari hasil optimasi rute yang dihasilkan, menunjukkan bahwa hasil rute tersebut dapat dilalui dalam sekali penerbangan oleh UAV DJI Agras T-30 yang digunakan pada tugas akhir ini. Dengan kapasitas *drone*, dari segi kapasitas tangki, baterai, dan kecepatan rata-rata terbangnya, lintasan ini berada jauh di bawah ambang batas operasional *drone* sehingga seluruh area dapat diselesaikan dalam satu siklus penyemprotan. Dengan demikian, dapat disimpulkan bahwa penerapan algoritma *hybrid simulated annealing* dalam tugas akhir ini layak dijadikan alternatif dalam solusi optimasi rute UAV pada lahan pertanian. Dengan hasil rute yang mendekati solusi OR-Tools, hal tersebut sekaligus memperlihatkan potensi algoritma metaheuristik untuk digunakan pada skenario perencanaan distribusi agrokimia yang memerlukan fleksibilitas penyesuaian parameter sesuai kondisi nyata di lapangan.

4.4.3.2 Perbandingan *Hybrid* SA dengan ACO

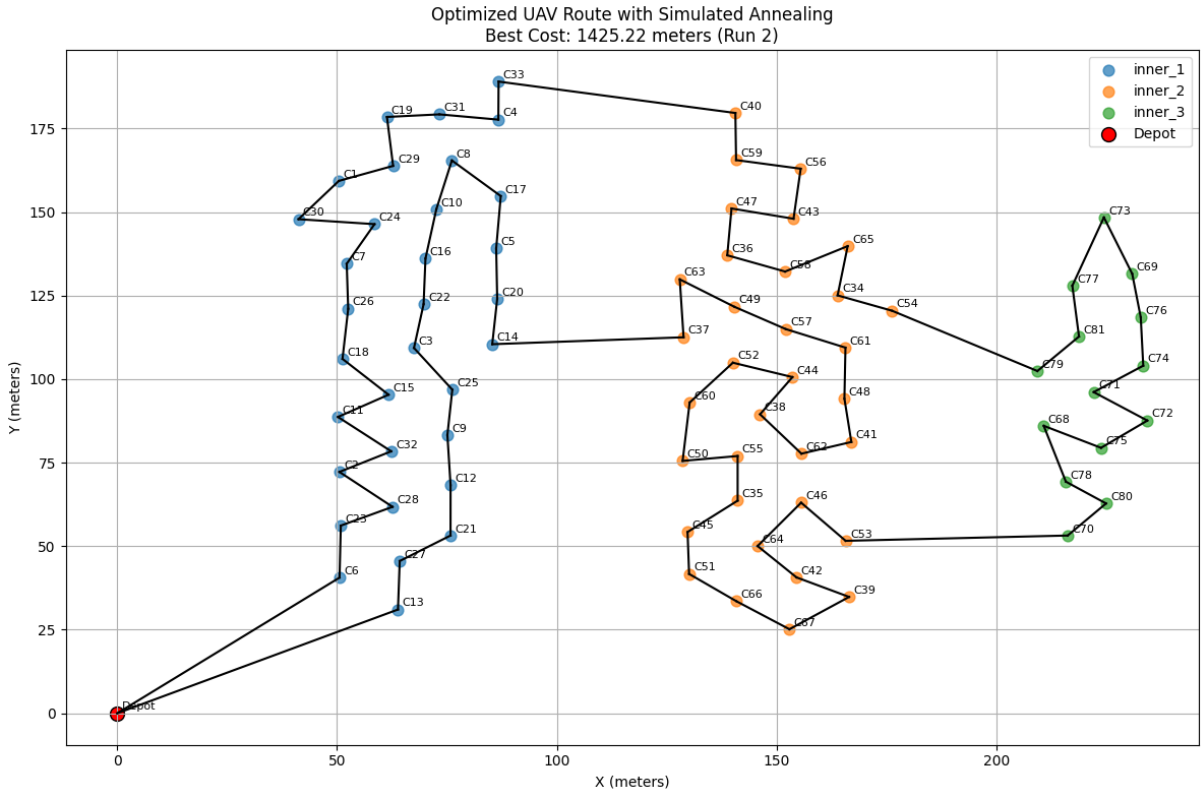
Pada subbab ini, dilakukan analisis perbandingan antara algoritma *hybrid* SA dan ACO yang digunakan sebagai pembanding tambahan dalam tugas akhir ini. Kedua algoritma diuji pada dataset yang identik sehingga evaluasi dapat dilakukan secara adil. Perbandingan difokuskan pada tiga aspek utama, yaitu total jarak tempuh terpendek yang diperoleh, waktu

komputasi, serta estimasi waktu penerbangan UAV berdasarkan hasil jarak tersebut. Berikut hasil uji coba komparatif dari kedua algoritma dirangkum pada tabel 4.20 di bawah ini.

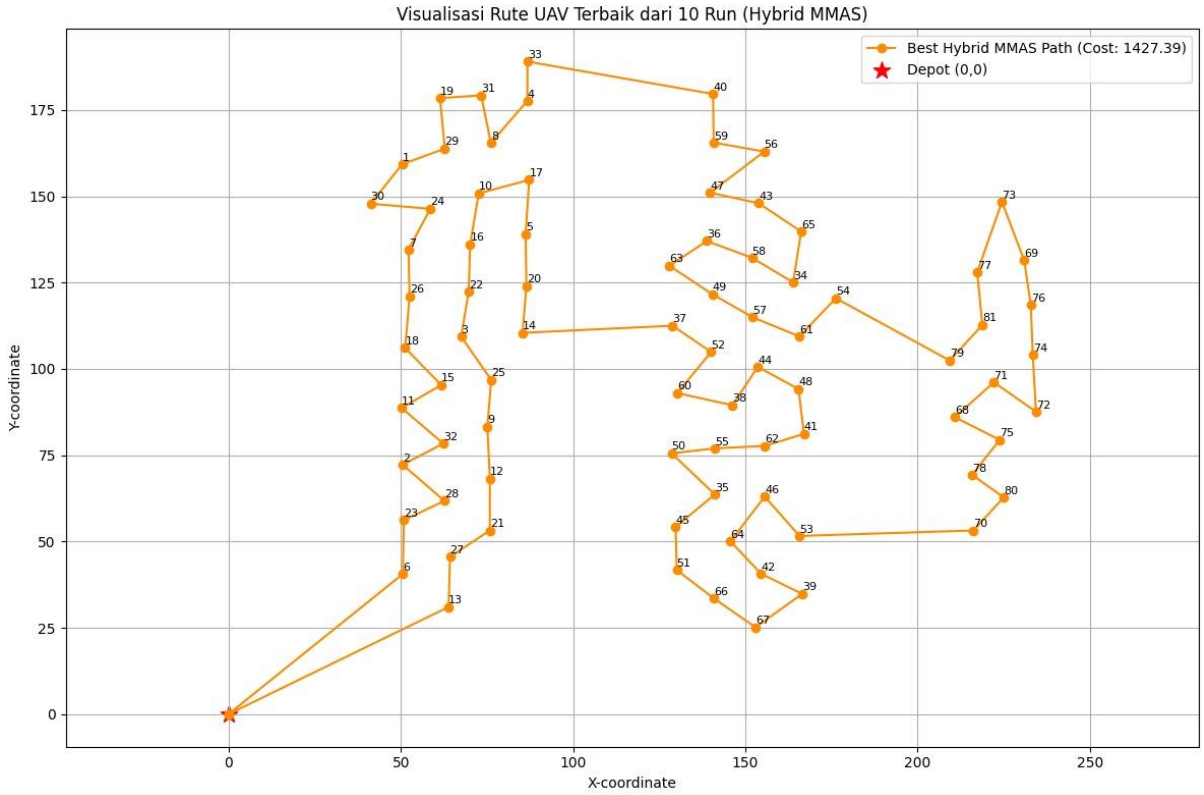
Tabel 4.20 Perbandingan Hasil Kinerja *Hybrid* SA dan ACO dalam Optimasi Rute

Algoritma	Jarak Terpendek (meter)	Jarak Rata-rata (meter)	Standar Deviasi (meter)	Rata-rata Waktu Komputasi (detik)	Estimasi Waktu Penerbangan (detik)
Hybrid SA	1425.22	1439.16	10.32	175.05	203.6
ACO	1427.39	1449.47	12.23	197.81	203.91

Selanjutnya, untuk memperjelas pola lintasan rute yang terbentuk dari kedua algoritma pada datase yang sama, ditampilkan visualisasi jalur UAV hasil optimasi pada dua gambar di bawah ini.



Gambar 4.10 Visualisasi Rute Terbaik Hasil *Hybrid* SA



Gambar 4.11 Visualisasi Rute Terbaik Hasil ACO

Berdasarkan hasil yang di dapatkan, pada tabel 4.20 dapat diketahui bahwa algoritma *hybrid* SA menghasilkan total jarak rute terpendek sebesar 1425.22 meter dengan rata-rata jarak sebesar 1439.16 meter, lebih pendek dibandingkan dengan ACO yang mencapai 1427.39 meter, dengan rata-rata jarak sebesar 1449.47 meter. Dari sisi kestabilan *multi-run*, standar deviasi *hybrid* SA sebesar 10.32 meter sedikit lebih rendah dibanding ACO yang berada pada 12.23 meter. Hasil ini menunjukkan variasi dari solusi *hybrid* SA relatif lebih kecil. Waktu komputasi rata-rata *hybrid* SA juga tercatat lebih cepat, yaitu 175.05 detik, sedangkan ACO memerlukan waktu rata-rata selama 197.81 detik.

Secara konsep, terdapat perbedaan mendasar antara kedua algoritma dalam hal pencarian solusi. *Hybrid* SA merupakan *single solution-based search* yang bekerja dengan memodifikasi satu solusi saat ini melalui mekanisme *neighborhood mutation* dan probabilitas penerimaan yang dikontrol *cooling schedule*, sedangkan ACO adalah *population-based search* yang menggunakan sejumlah agen, yaitu semut untuk mengeksplor jalur secara paralel dan memperkuat jalur terbaik melalui *pheromone update*. Secara umum, algoritma seperti ACO memiliki potensi dalam eksplorasi ruang solusi yang lebih luas dan mampu menghindari *local minima* lebih baik dibanding algoritma *single solution* seperti SA. Namun, pada kasus ini hasil menunjukkan hal sebaliknya, dimana *hybrid* SA dengan *grid search* mampu menghasilkan lintasan terbaik yang lebih pendek dan waktu komputasi lebih cepat dibanding ACO. Hal ini menunjukkan bahwa dengan konfigurasi parameter yang tepat, algoritma *single solution* seperti SA tetap dapat diarahkan untuk memberikan performa yang sangat kompetitif bahkan mengungguli ACO pada dataset dan skenario distribusi agrokimia yang diuji dalam tugas akhir.

Selain itu, dari hasil visualisasi lintasan yang dihasilkan ketiga algoritma, dapat diamati pola perpindahan UAV antara klaster 1, klaster 2, dan klaster 3 yang menunjukkan variasi arah

jalur menarik. Pada lintasan *hybrid* SA dan ACO, perpindahan dari klaster 1 ke klaster 2 umumnya melalui titik-titik pada bagian atas lahan, sedangkan saat kembali dari klaster 2 ke klaster 1, jalur cenderung melewati area tengah. Sementara itu, pada perpindahan dari klaster 2 menuju klaster 3, lintasan bergerak ke tengah lahan, sedikit naik ke atas untuk menjangkau centroid terdekat, sebelum akhirnya turun ke klaster 3 di area bawah lalu kembali ke klaster 2. Namun, pada hasil lintasan algoritma OR-Tools terlihat sedikit perbedaan, di mana baik perpindahan dari klaster 1 ke klaster 2 maupun sebaliknya sama-sama melalui jalur di bagian tengah lahan. Hal ini terjadi karena algoritma tidak diberikan rekomendasi atau aturan eksplisit jalur, melainkan sepenuhnya memilih secara otomatis berdasarkan posisi *centroid* terdekat dan proses minimasi total jarak tempuh yang dilakukan pada setiap algoritma.

Fenomena ini menegaskan bahwa jalur yang terbentuk, baik melewati atas, tengah, maupun bawah, sepenuhnya merupakan hasil optimasi algoritma terhadap distribusi *centroid* pada masing-masing klaster. Dengan demikian, pola jalur ini dapat menjadi pertimbangan operasional yang adaptif dalam penerapan di lapangan, untuk memastikan efisiensi distribusi agrokimia dalam satu siklus penerbangan UAV. Dengan demikian, dapat disimpulkan bahwa baik algoritma *hybrid* SA, OR-Tools, maupun ACO sama-sama mampu menghasilkan jalur UAV yang kompetitif dalam mengoptimasi rute distribusi agrokimia pada area *stressed region*. Perbedaan karakteristik algoritma, yakni *single solution-based* pada hybrid SA, *deterministic* pada OR-Tools, serta *population-based* pada ACO, menunjukkan bagaimana masing-masing pendekatan secara otomatis memilih jalur terbaik berdasarkan distribusi *centroid* tanpa adanya rekomendasi jalur manual. Hasil ini sekaligus menegaskan potensi adaptasi algoritma terhadap kondisi lahan yang berbeda

BAB 5 Kesimpulan dan Saran

Sebagai penutup, pada bab ini akan disampaikan kesimpulan yang diperoleh berdasarkan hasil pengujian dan analisis pada bab sebelumnya, serta disertakan beberapa poin saran yang dapat dijadikan acuan untuk pengembangan lebih lanjut sebagai penelitian atau pengerjaan tugas akhir pada bidang optimasi rute UAV dalam distribusi agrokimia.

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, berikut beberapa kesimpulan yang dapat diambil sebagai berikut:

1. Tugas akhir ini berusaha memodelkan permasalahan distribusi penyemprotan agrokimia pada lahan pertanian, khususnya pada area terdampak (*stressed region*) sebagai bentuk masalah *Vehicle Routing Problem* (VRP). Pemodelan dilakukan dengan merepresentasikan lahan pertanian terdampak ke dalam sejumlah titik *centroid* hasil klasterisasi yang kemudian dibentuk menjadi *graph* dimana setiap *node* harus dikunjungi oleh UAV untuk memastikan seluruh area mendapatkan intervensi penyemprotan. Pendekatan ini memungkinkan analisis rute optimal UAV dilakukan secara bertahap dan sistematis, mulai dari proses segmentasi data spasial, pembentukan *centroid*, hingga penyusunan jalur lintasan.
2. Dari sisi batasan kapasitas UAV, hasil perhitungan menunjukkan bahwa rute lintasan terpendek yang diperoleh, yaitu sebesar 1425.22 meter serta estimasi waktu penerbangan selama 203.66 detik, dimana angka tersebut masih dapat ditempuh oleh UAV DJI Agras T-30 dalam satu kali penerbangan. Hal ini mengingat *drone* yang digunakan memiliki kapasitas tangki 40 liter, kecepatan rata-rata 7 m/s, serta daya tahan beterai yang membuat waktu penerbangan efektif hingga 20 menit, sehingga seluruh titik pada lintasan dapat dilalui dalam satu siklus penyemprotan tanpa perlu pengisian ulang tangki atau penggantian beterai.
3. Permasalahan *covering set problem* pada tugas akhir ini ditangani melalui tahap klasterisasi dengan metode *lloyd algorithm* (K-Means) yang diperkuat menggunakan pendekatan *grid search*. Hasil menunjukkan bahwa area cakupan semprot UAV (*coverage ratio*) mencapai 97.28% dari total area terdampak dan tingkat tumpang tindih (*overlaap ratio*) sebesar 19.96%. Angka ini memperlihatkan bahwa distribusi *centroid* yang dihasilkan efektif dalam menjangkau hampir seluruh target, sekaligus menjaga *overlap* pada batas yang wajar sehingga dapat meminimalkan risiko penggunaan agrokimia yang berlebih.
4. Dari hasil pengujian variasi parameter pada algoritma *hybrid simulated annealing*, diperoleh bahwa keseimbangan antara nilai *initial temperature*, *cooling rate*, jumlah iterasi, serta interval *two-opt* berpengaruh signifikan terhadap kualitas rute yang dihasilkan. Nilai *cooling rate* yang gradual dan jumlah iterasi yang tinggi terbukti memberikan total jarak rute yang lebih rendah, meskipun berdampak pada waktu komputasi yang lebih tinggi. Hal ini memperlihatkan *trade-off* klasik pada algoritma metaheuristik dimana peningkatan kualitas solusi sering kali diiringi dengan kenaikan beban komputasi.
5. Melalui perbandingan dengan algoritma OR-Tools, algoritma *hybrid simulated annealing* mampu menghasilkan solusi rute UAV dengan kualitas kompetitif, hanya dengan perbedaan sebesar 0.36 atau 0.02%. Meskipun demikian, OR-Tools memiliki

keunggulan pada aspek kestabilan solusi, dimana algoritma *hybrid simulated annealing* masih memperlihatkan standar deviasi di rentang 10 hingga 26 meter antar *multi-run*. Angka variasi ini masih dapat diterima dalam konteks operasional distribusi agrokimia dalam lahan pertanian pada bentuk area tidak beraturan, sekaligus menunjukkan bahwa algoritma metaheuristik cukup fleksibel untuk disesuaikan dengan berbagai kondisi lahan

5.2 Saran

Adapun beberapa saran yang dapat dipertimbangkan untuk penelitian atau pengembangan lebih lanjut adalah sebagai berikut:

1. Penentuan parameter pada penelitian ini masih dilakukan berdasarkan pendekatan *trial and error*, dengan mempertimbangkan kompromi terhadap waktu komputasi. Oleh karena itu, disarankan agar penelitian selanjutnya dapat memperluas rentang variasi parameter yang diuji sehingga dapat mengeksplorasi ruang solusi lebih jauh dan berpotensi menemukan konfigurasi parameter yang menghasilkan kualitas lintasan yang lebih efisien.
2. Implementasi optimasi rute dalam UAV dapat diperluas dengan mempertimbangkan aspek-aspek realistis yang mencerminkan kondisi lapangan, seperti faktor medan atau topografi lahan pertanian, pengaruh cuaca terhadap kondisi penerbangan UAV, serta perbedaan kebutuhan pupuk yang berbeda di setiap daerah. Hal ini akan meningkatkan relevansi model dengan skenario distribusi agrokimia sebenarnya.
3. Untuk meningkatkan kualitas hasil optimasi, disarankan agar algoritma *hybrid simulated annealing* mempertimbangkan variasi model metaheuristik lain, seperti *genetic algorithm*, *ant colony optimization*, dan yang lainnya guna mengeksplorasi ruang solusi secara lebih luas sekaligus mengurangi risiko terjebak pada solusi lokal minimum.
4. Penelitian selanjutnya juga dapat mengembangkan model algoritma yang mampu menangani skenario distribusi *multi depot* yang melibatkan beberapa UAV secara bertahap dalam sistem ditribusi agrokimia. Dengan demikian, model yang dikembangkan akan semakin mendekati kondisi operasional di dunia nyata dan dapat memberikan kontribusi lebih praktis dalam pengelolaan lahan pertanian berskala besar.

DAFTAR PUSTAKA

Ahmed, F., Mohanta, J. C., Keshari, A., & Yadav, P. S. (2022). Recent Advances in Unmanned Aerial Vehicles: A Review. *Arabian Journal for Science and Engineering*, *47*(7), 7963–7984. https://doi.org/10.1007/s13369-022-06738-0

Arifuddin, A., Utamima, A., Mahananto, F., Vinarti, R. A., & Fernanda, N. (2024). Optimizing the Capacitated Vehicle Routing Problem at PQR Company: A Genetic Algorithm and Grey Wolf Optimizer Approach. *Procedia Computer Science*, *234*, 420–427. https://doi.org/10.1016/j.procs.2024.03.023

Ariyani, A. K., Mahmudy, W. F., & Anggodo, Y. P. (2018). Hybrid Genetic Algorithms and Simulated Annealing for Multi-trip Vehicle Routing Problem with Time Windows. *International Journal of Electrical and Computer Engineering (IJECE)*, *8*(6), 4713. https://doi.org/10.11591/ijece.v8i6.pp4713-4723

Bakhtiari, A. A., Navid, H., Mehri, J., & Bochtis, D. D. (2011). Optimal route planning of agricultural field operations using ant colony optimization. *Agricultural Engineering International: CIGR Journal*, *13*(4), 1–16.

BPS. (2024). *Jumlah dan Tingkat Pertumbuhan Penduduk Indonesia 2024*. https://www.bps.go.id/id/statistics-table/2/MTk3NSMy/jumlah-penduduk-pertengahan-tahun--ribu-jiwa-.html

Chandomi-Castellanos, E., Escobar-Gomez, E. N., Aguilar Marroquin-Cano, S. F., Hernandez-De- Leon, H. R., Velazquez-Trujillo, S., Sarmiento-Torres, J. A., & De-Coss-Parez, C. V. (2022). Modified Simulated Annealing Hybrid Algorithm to Solve the Traveling Salesman Problem. *2022 8th International Conference on Control, Decision and Information Technologies, CoDIT 2022*, 1536–1541. https://doi.org/10.1109/CoDIT55151.2022.9804145

Cisternas, I., Velásquez, I., Caro, A., & Rodríguez, A. (2020). Systematic literature review of implementations of precision agriculture. *Computers and Electronics in Agriculture*, *176*(July), 105626. https://doi.org/10.1016/j.compag.2020.105626

Conesa-Muñoz, J., Bengochea-Guevara, J. M., Andujar, D., & Ribeiro, A. (2016). Route planning for agricultural tasks: A general approach for fleets of autonomous vehicles in site-specific herbicide applications. *Computers and Electronics in Agriculture*, *127*, 204–220. https://doi.org/10.1016/j.compag.2016.06.012

Furchi, A., Lippi, M., Carpio, R. F., & Gasparri, A. (2023). Route Optimization in Precision Agriculture Settings: A Multi-Steiner TSP Formulation. *IEEE Transactions on Automation Science and Engineering*, *20*(4), 2551–2568. https://doi.org/10.1109/TASE.2022.3204584

Löffler, S., Becker, I., & Hofsted, P. (2024). Enhancing Constraint Optimization Problems with Greedy Search and Clustering: A Focus on the Traveling Salesman Problem. *International Conference on Agents and Artificial Intelligence*, *3*(Icaart), 1170–1178. https://doi.org/10.5220/0012453000003636

Muhie, S. H. (2022). Novel approaches and practices to sustainable agriculture. *Journal of Agriculture and Food Research*, *10*(August), 100446.

https://doi.org/10.1016/j.jafr.2022.100446

Muriyatmoko, D., Djunaidy, A., & Muklason, A. (2024). Heuristics and Metaheuristics for Solving Capacitated Vehicle Routing Problem: An Algorithm Comparison. *Procedia Computer Science*, *234*, 494–501. https://doi.org/10.1016/j.procs.2024.03.032

Nazeer, I., Umer, S., Rout, R. K., & Tanveer, M. (2024). Artificial intelligence-based smart agricultural systems for saffron cultivation with integration of Unmanned Aerial Vehicle imagery and deep learning approaches. *Computers and Electrical Engineering*, *119*(PA), 109542. https://doi.org/10.1016/j.compeleceng.2024.109542

Ribeiro Junior, F. M., Bianchi, R. A. C., Prati, R. C., Kolehmainen, K., Soininen, J. P., & Kamienski, C. A. (2022). Data reduction based on machine learning algorithms for fog computing in IoT smart agriculture. *Biosystems Engineering*, *223*, 142–158. https://doi.org/10.1016/j.biosystemseng.2021.12.021

Rodríguez-Esparza, E., Masegosa, A. D., Oliva, D., & Onieva, E. (2024). A new Hyper-heuristic based on Adaptive Simulated Annealing and Reinforcement Learning for the Capacitated Electric Vehicle Routing Problem. *Expert Systems with Applications*, *252*(PB), 124197. https://doi.org/10.1016/j.eswa.2024.124197

Samuel, M., Yahya, K., Aldababsa, M., Amer, A., Dofan, J. A. R., Merchan-Cruz, E. A., & Hafez, M. (2023). A Review on Deformable Voronoi Diagrams for Robot Path Planning in Dynamic Environments. *2nd International Engineering Conference on Electrical, Energy, and Artificial Intelligence, EICEEAI 2023*, 1–8. https://doi.org/10.1109/EICEEAI60672.2023.10590452

Sharma, A., Jain, A., Gupta, P., & Chowdary, V. (2021). Machine Learning Applications for Precision Agriculture: A Comprehensive Review. *IEEE Access*, *9*, 4843–4873. https://doi.org/10.1109/ACCESS.2020.3048415

Srivastava, K., Bhutoria, A. J., Sharma, J. K., Sinha, A., & Pandey, P. C. (2019). UAVs technology for the development of GUI based application for precision agriculture and environmental research. *Remote Sensing Applications: Society and Environment*, *16*(May), 100258. https://doi.org/10.1016/j.rsase.2019.100258

Srivastava, K., Pandey, P. C., & Sharma, J. K. (2020). An approach for route optimization in applications of precision agriculture using uavs. *Drones*, *4*(3), 1–24. https://doi.org/10.3390/drones4030058

Tahilyani, S., Saxena, S., Karras, D. A., Kant Gupta, S., Kumar Dixit, C., & Haralayya, B. (2022). Deployment of Autonomous Vehicles in Agricultural and using Voronoi Partitioning. *IEEE International Conference on Knowledge Engineering and Communication Systems, ICKES 2022*, 1–5. https://doi.org/10.1109/ICKECS56523.2022.10060773

Utamima, A., Reiners, T., & Ansaripoor, A. H. (2019). Optimisation of agricultural routing planning in field logistics with Evolutionary Hybrid Neighbourhood Search. *Biosystems Engineering*, *184*, 166–180. https://doi.org/10.1016/j.biosystemseng.2019.06.001

Widiastuti, D. P., Hatta, M., Aziz, H., Permana, D., Santari, P. T., Rohaeni, E. S., Ahmad, S. N., Bakrie, B., Tan, S. S., & Rakhmani, S. I. W. (2024). Peatlands management for sustainable use on the integration of maize and cattle in a circular agriculture system in West Kalimantan, Indonesia. *Heliyon*, *10*(10), e31259. https://doi.org/10.1016/j.heliyon.2024.e31259

Zangina, U., Buyamin, S., Abidin, M. S. Z., & Mahmud, M. S. A. (2021). Agricultural rout planning with variable rate pesticide application in a greenhouse environment. *Alexandria Engineering Journal*, *60*(3), 3007–3020. https://doi.org/10.1016/j.aej.2021.01.010

LAMPIRAN

Tabel Lampiran 1 Hasil seluruh Uji Coba *Hybrid Simulated Annealing*

No	*Initial temp*	*Cooling rate*	Iterasi maksimum	2-opt interval	*Cost* (meter)	*Runtime* (detik)
1	100	0,995	1000	10	1451,23	30,69
2	100	0,995	1000	10	1445,19	31,37
3	100	0,995	1000	10	1500,86	32,19
4	100	0,995	1000	10	1487,47	32,62
5	100	0,995	1000	10	1481,87	35,32
6	100	0,995	1000	10	1466,14	33,14
7	100	0,995	1000	10	1447,39	36,1
8	100	0,995	1000	10	1489,23	41,42
9	100	0,995	1000	10	1452,5	34,16
10	100	0,995	1000	10	1483,43	32,93
11	200	0,99	1000	10	1453,31	31,84
12	200	0,99	1000	10	1439,53	30,8
13	200	0,99	1000	10	1476,13	32,11
14	200	0,99	1000	10	1443,88	30,02
15	200	0,99	1000	10	1468,64	33,36
16	200	0,99	1000	10	1448,57	41,54
17	200	0,99	1000	10	1477,09	33,66
18	200	0,99	1000	10	1468,96	36,99
19	200	0,99	1000	10	1456,19	35,19
20	200	0,99	1000	10	1515,44	32,54
21	100	0,97	1000	10	1510,74	13,12
22	100	0,97	1000	10	1505,36	13,28
23	100	0,97	1000	10	1498,2	13,22
24	100	0,97	1000	10	1453,17	13,46
25	100	0,97	1000	10	1501,3	14,93
26	100	0,97	1000	10	1463,34	14,45
27	100	0,97	1000	10	1489,63	14,7
28	100	0,97	1000	10	1544,56	13,9
29	100	0,97	1000	10	1471,58	14,32
30	100	0,97	1000	10	1482,53	14,9
31	100	0,995	2000	10	1428,77	58,7
32	100	0,995	2000	10	1439,08	61,57
33	100	0,995	2000	10	1432,03	65,17
34	100	0,995	2000	10	1451,91	65,16
35	100	0,995	2000	10	1442,62	66,89
36	100	0,995	2000	10	1458,73	67,89
37	100	0,995	2000	10	1510,67	102,13
38	100	0,995	2000	10	1464,27	80,41
39	100	0,995	2000	10	1444,58	65,82
40	100	0,995	2000	10	1466,48	124,85
41	100	0,995	1000	5	1489,02	58,13
42	100	0,995	1000	5	1492,12	67,27

43	100	0,995	1000	5	1458,24	67,16
44	100	0,995	1000	5	1445,51	62,62
45	100	0,995	1000	5	1490,1	66,06
46	100	0,995	1000	5	1484,48	77,06
47	100	0,995	1000	5	1465,32	71,1
48	100	0,995	1000	5	1514,99	66,36
49	100	0,995	1000	5	1472,09	68,48
50	100	0,995	1000	5	1452,07	64,72
51	500	0,999	3000	20	1466,12	58,39
52	500	0,999	3000	20	1472,38	63,18
53	500	0,999	3000	20	1475,86	64,2
54	500	0,999	3000	20	1453,44	60,75
55	500	0,999	3000	20	1475,4	65,2
56	500	0,999	3000	20	1466,12	66,34
57	500	0,999	3000	20	1505,22	66,02
58	500	0,999	3000	20	1473,2	66,92
59	500	0,999	3000	20	1475,2	68,63
60	500	0,999	3000	20	1492,25	63,49
61	500	0,995	3000	20	1433,72	53,07
62	500	0,995	3000	20	1430,01	56,52
63	500	0,995	3000	20	1483,61	53,8
64	500	0,995	3000	20	1445,71	55,51
65	500	0,995	3000	20	1465,09	79,44
66	500	0,995	3000	20	1440,16	85,64
67	500	0,995	3000	20	1471,26	56,46
68	500	0,995	3000	20	1444,36	53,21
69	500	0,995	3000	20	1460,05	53,32
70	500	0,995	3000	20	1439,67	52,28
71	500	0,995	3000	5	1434,54	194,97
72	500	0,995	3000	5	1481,41	208,03
73	500	0,995	3000	5	1441,52	211,83
74	500	0,995	3000	5	1462,29	207,53
75	500	0,995	3000	5	1467,2	214,83
76	500	0,995	3000	5	1435,32	210,53
77	500	0,995	3000	5	1429,03	207,25
78	500	0,995	3000	5	1428,51	208,58
79	500	0,995	3000	5	1467,41	211,45
80	500	0,995	3000	5	1439,36	207,62
81	200	0,995	3000	5	1437,53	182,52
82	200	0,995	3000	5	1425,22	186,08
83	200	0,995	3000	5	1427,3	175,55
84	200	0,995	3000	5	1442,98	175,55
85	200	0,995	3000	5	1446,55	165,01
86	200	0,995	3000	5	1461,43	165,89
87	200	0,995	3000	5	1434,52	158,61
88	200	0,995	3000	5	1441,07	168,28
89	200	0,995	3000	5	1440,98	190,63
90	200	0,995	3000	5	1434	182,35

91	500	0,99	3000	10	1448,17	59,6
92	500	0,99	3000	10	1465,68	57,1
93	500	0,99	3000	10	1463,4	56,36
94	500	0,99	3000	10	1488,64	54,38
95	500	0,99	3000	10	1436,43	53,69
96	500	0,99	3000	10	1462,01	55,26
97	500	0,99	3000	10	1441,94	52,47
98	500	0,99	3000	10	1429,17	56
99	500	0,99	3000	10	1492,17	55,91
100	500	0,99	3000	10	1435,67	55,53

Tabel Lampiran 2 Hasil Uji Coba Optimasi OR-Tools

No	Cost (meter)	*Runtime* (detik)
1	1424,86	186,05
2	1424,86	186,01
3	1424,86	186
4	1424,86	186,01
5	1424,86	186
6	1424,86	186
7	1424,86	186
8	1424,86	186,01
9	1424,86	186
10	1424,86	186

Tabel Lampiran 3 Hasil Proses *Grid Search* Klasterisasi Lloyd

Inner	*Clusters*	*Coverage* (%)	*Overlap* (%)	Loss
inner_1	5	18,51936985	0	20317,27927
inner_1	6	22,46929244	0	18433,03185
inner_1	7	26,24728739	0	16718,38785
inner_1	8	29,99964557	0	15100,14886
inner_1	9	33,73175366	0	13574,44142
inner_1	10	37,38446349	0	12162,11624
inner_1	11	41,0561081	0	10823,14718
inner_1	12	44,80024846	0	9541,037712
inner_1	13	48,50999865	0	8353,660718
inner_1	14	52,14085985	0	7271,491888
inner_1	15	55,7100118	0	6284,809164
inner_1	16	59,39856742	0,097555617	5341,536274
inner_1	17	62,91314015	0,078151772	4523,185558
inner_1	18	66,76625892	0,097264724	3709,563511
inner_1	19	70,22350926	0,465316008	3041,522082
inner_1	20	73,74883222	0,410129495	2451,134456
inner_1	21	77,17268729	0,733083981	1934,47267
inner_1	22	79,65071701	1,872584833	1570,883135
inner_1	23	82,14890995	3,267037947	1235,976267
inner_1	24	83,63105355	4,507972358	1043,830144

inner_1	25	86,02773971	5,710301914	789,8676438
inner_1	26	88,08519224	5,908929273	624,4462063
inner_1	27	90,20827782	8,616011392	417,2286662
inner_1	28	90,96333925	10,95633707	326,7715519
inner_1	29	93,62904888	10,81553519	206,1214485
inner_1	30	94,98756073	13,35664986	119,5077432
inner_1	31	95,39572337	14,7267994	91,4047343
inner_1	32	96,73321065	16,24279521	46,13232581
inner_1	33	97,07026005	19,47303768	26,02781775
inner_1	34	97,78555668	23,47328431	26,77498157
inner_1	35	98,61070831	24,82150159	29,03727178
inner_1	36	98,31256302	28,66301483	83,59015668
inner_1	37	99,34824526	30,69504822	115,6584092
inner_1	38	99,43371006	35,0905758	228,6875308
inner_1	39	99,66801857	38,27080344	334,1528932
inner_1	40	99,67023141	41,67473539	470,120396
inner_1	41	99,71952858	45,44140936	647,501303
inner_1	42	99,8605825	49,1473128	849,6241552
inner_1	43	99,9144305	53,41501874	1116,585444
inner_1	44	99,89878443	57,40822694	1399,406177
inner_1	45	99,91136241	60,99061209	1680,253849
inner_1	46	99,94539409	64,55120454	1984,818771
inner_1	47	99,92336684	69,99486854	2499,504498
inner_1	48	99,93228379	73,37878007	2849,307918
inner_1	49	99,97968857	77,24849678	3277,391621
inner_1	35	98,61070831	24,82150159	29,03727178
inner_1	36	98,31256302	28,66301483	83,59015668
inner_1	37	99,34824526	30,69504822	115,6584092
inner_1	38	99,43371006	35,0905758	228,6875308
inner_1	39	99,66801857	38,27080344	334,1528932
inner_1	40	99,67023141	41,67473539	470,120396
inner_1	41	99,71952858	45,44140936	647,501303
inner_1	42	99,8605825	49,1473128	849,6241552
inner_1	43	99,9144305	53,41501874	1116,585444
inner_1	44	99,89878443	57,40822694	1399,406177
inner_1	45	99,91136241	60,99061209	1680,253849
inner_1	46	99,94539409	64,55120454	1984,818771
inner_1	47	99,92336684	69,99486854	2499,504498
inner_1	48	99,93228379	73,37878007	2849,307918
inner_1	49	99,97968857	77,24849678	3277,391621
inner_2	5	18,53700666	0	20308,65785
inner_2	6	22,10138712	0	18604,58166
inner_2	7	25,94393031	0	16852,90437
inner_2	8	29,57329225	0	15279,76349
inner_2	9	33,06020901	0	13842,80685
inner_2	10	36,69721797	0	12421,72664
inner_2	11	40,25305291	0	11109,09306
inner_2	12	43,67325843	0	9918,105449

inner_2	13	47,34265269	0	8718,388678
inner_2	14	50,73550678	0	7680,970877
inner_2	15	54,33524087	0,027654735	6654,705254
inner_2	16	58,34610909	0,058584553	5602,799934
inner_2	17	61,89999644	0,072416918	4751,939381
inner_2	18	65,03920123	0,648108008	4041,268076
inner_2	19	68,50064019	0,592732569	3353,271034
inner_2	20	71,65166902	0,712301754	2782,898911
inner_2	21	74,7633614	1,049804607	2269,773689
inner_2	22	78,30058481	1,262023575	1763,705619
inner_2	23	80,91562233	2,255237442	1407,517012
inner_2	24	83,26250167	3,363583626	1117,2019
inner_2	25	85,52210511	4,353064467	873,6549129
inner_2	26	88,48508308	4,727413219	631,0318418
inner_2	27	89,08975984	7,370141389	516,6133497
inner_2	28	91,78995656	8,431060504	336,0548009
inner_2	29	93,17762226	10,25665341	234,5673168
inner_2	30	94,4662867	11,69323604	160,8682761
inner_2	31	95,65075042	14,39772499	88,13340117
inner_2	32	96,44619694	15,55280819	57,66606361
inner_2	33	97,16125552	18,76522757	25,70007365
inner_2	34	97,8989736	20,66080019	13,67959266
inner_2	35	98,41268785	23,27451887	18,2811534
inner_2	36	99,21612603	25,94517384	37,18846721
inner_2	37	99,28603848	29,32670776	88,51670079
inner_2	38	99,61707958	31,74253192	138,32694
inner_2	39	99,70032656	35,2405043	232,5423839
inner_2	40	99,71668781	39,52748322	381,5633983
inner_2	41	99,84897068	43,42888309	548,9809922
inner_2	42	99,81891811	47,06367567	732,5409128
inner_2	43	99,8242699	51,0863156	966,4516608
inner_2	44	99,91832255	54,50915221	1190,9016
inner_2	45	99,91274192	58,37907187	1472,975999
inner_2	46	99,91777078	61,97281794	1761,737731
inner_2	47	99,93486619	66,86285056	2196,139489
inner_2	48	99,94179869	71,91899791	2695,592507
inner_2	49	99,95651472	76,05750996	3142,450096
inner_3	5	44,90133891	0	9507,587362
inner_3	6	53,09694738	0	6999,689035
inner_3	7	62,45527813	0,028980338	4627,660048
inner_3	8	70,04373972	1,11188779	3048,893372
inner_3	9	78,80472404	1,754676087	1680,611013
inner_3	10	84,91256809	4,318034834	928,8158362
inner_3	11	89,15194741	8,394591892	487,7262324
inner_3	12	92,57940177	11,64557279	234,992288
inner_3	13	94,99199405	17,16855898	83,25742917
inner_3	14	97,12648351	20,59372676	25,12380258
inner_3	15	98,34576949	27,56429384	65,42797699

inner_3	16	99,34798255	33,77725069	191,0880169
inner_3	17	99,7657287	41,43348557	459,558953
inner_3	18	99,9302106	50,2114988	912,7492712
inner_3	19	99,99285816	60,15118239	1612,117601
inner_3	20	99,99856473	69,73148268	2473,220376
inner_3	21	100	80,13689516	3616,446159
inner_3	22	100	91,74367635	5147,155096
inner_3	23	100	103,9675046	7050,54183
inner_3	24	100	114,7428067	8976,199417
inner_3	25	100	130,3475325	12176,57792
inner_3	26	100	142,5531561	15019,27608
inner_3	27	100	154,1780758	18003,75602
inner_3	28	100	171,3575141	22909,09707
inner_3	29	100	183,2890764	26663,32248
inner_3	30	100	201,5107279	32946,14434
inner_3	31	100	218,3343758	39336,52462
inner_3	32	100	235,078097	46258,58781
inner_3	33	100	257,6491912	56477,13807
inner_3	34	100	275,1906451	65122,26534
inner_3	35	100	292,439371	74223,21089
inner_3	36	100	306,9114075	82318,15578
inner_3	37	100	336,4642135	100149,5985
inner_3	38	100	350,8404208	109455,3841
inner_3	39	100	370,9547108	123169,209
inner_3	40	100	398,8692976	143541,9447
inner_3	41	100	406,1804672	149135,3532
inner_3	42	100	441,6575534	177795,0924
inner_3	43	100	459,1545237	192856,6957
inner_3	44	100	495,240565	225853,5947
inner_3	45	100	505,807346	236008,7775
inner_3	46	100	547,8136933	278587,2948
inner_3	47	100	582,7449029	316681,8257
inner_3	48	100	592,7734765	328069,4554
inner_3	49	100	616,4649506	355770,4373

Tabel Lampiran 4 Hasil Uji Coba *Grid Search Hybrid Simulated Annealing*

Initial Temp	*Cooling Rate*	*Max Iter*	*2-opt Interval*	*Best Cost*	*Duration (s)*
100	0,97	1000	5	1544.81	48.10
100	0,97	1000	10	1497.96	22.16
100	0,97	1000	20	1486.21	12.27
100	0,97	2000	5	1472.38	48.74
100	0,97	2000	10	1469.35	22.27
100	0,97	2000	20	1522.79	10.95
100	0,97	3000	5	1470.37	47.35
100	0,97	3000	10	1536.38	24.38
100	0,97	3000	20	1495.44	12.92

100	0,99	1000	5	1439.88	98.17
100	0,99	1000	10	1519.18	49.18
100	0,99	1000	20	1444.47	25.54
100	0,99	2000	5	1463.82	136.43
100	0,99	2000	10	1469.92	71.06
100	0,99	2000	20	1512.92	35.32
100	0,99	3000	5	1442.06	136.37
100	0,99	3000	10	1448.32	66.18
100	0,99	3000	20	1447.01	38.41
100	0,995	1000	5	1456.25	102.78
100	0,995	1000	10	1466.10	48.84
100	0,995	1000	20	1494.98	26.20
100	0,995	2000	5	1454.62	198.54
100	0,995	2000	10	1450.63	98.90
100	0,995	2000	20	1504.04	51.63
100	0,995	3000	5	1443.17	274.37
100	0,995	3000	10	1456.69	133.14
100	0,995	3000	20	1461.94	68.49
100	0,999	1000	5	1493.24	104.88
100	0,999	1000	10	1457.42	52.69
100	0,999	1000	20	1502.64	29.02
100	0,999	2000	5	1453.84	208.93
100	0,999	2000	10	1456.50	107.22
100	0,999	2000	20	1501.56	55.68
100	0,999	3000	5	1447.00	304.60
100	0,999	3000	10	1459.23	153.18
100	0,999	3000	20	1472.47	83.45
200	0,97	1000	5	1493.95	50.71
200	0,97	1000	10	1490.62	25.24
200	0,97	1000	20	1492.83	12.46
200	0,97	2000	5	1545.64	50.83
200	0,97	2000	10	1447.99	23.16
200	0,97	2000	20	1500.06	14.53
200	0,97	3000	5	1479.95	48.39
200	0,97	3000	10	1485.18	25.44
200	0,97	3000	20	1487.07	11.90
200	0,99	1000	5	1474.25	102.25
200	0,99	1000	10	1471.82	51.79
200	0,99	1000	20	1482.11	26.21
200	0,99	2000	5	1471.30	141.41
200	0,99	2000	10	1435.32	74.19
200	0,99	2000	20	1453.79	38.34
200	0,99	3000	5	1452.88	144.17
200	0,99	3000	10	1436.39	76.17
200	0,99	3000	20	1444.04	36.34
200	0,995	1000	5	1480.94	103.69
200	0,995	1000	10	1473.40	53.94
200	0,995	1000	20	1465.06	29.00

200	0,995	2000	5	1443.17	203.48
200	0,995	2000	10	1440.85	99.26
200	0,995	2000	20	1458.57	52.47
200	0,995	3000	5	1431.11	287.81
200	0,995	3000	10	1463.35	145.63
200	0,995	3000	20	1442.74	70.50
200	0,999	1000	5	1442.19	111.61
200	0,999	1000	10	1500.27	62.22
200	0,999	1000	20	1488.64	36.01
200	0,999	2000	5	1481.93	215.03
200	0,999	2000	10	1472.25	113.54
200	0,999	2000	20	1470.47	60.54
200	0,999	3000	5	1453.46	315.93
200	0,999	3000	10	1481.44	169.28
200	0,999	3000	20	1459.85	87.72
500	0,97	1000	5	1487.02	52.56
500	0,97	1000	10	1522.50	25.98
500	0,97	1000	20	1549.44	15.41
500	0,97	2000	5	1446.07	53.76
500	0,97	2000	10	1466.26	25.12
500	0,97	2000	20	1473.30	12.98
500	0,97	3000	5	1496.26	52.15
500	0,97	3000	10	1508.03	27.96
500	0,97	3000	20	1509.54	13.64
500	0,99	1000	5	1481.44	106.88
500	0,99	1000	10	1494.57	53.82
500	0,99	1000	20	1514.16	28.87
500	0,99	2000	5	1446.25	160.60
500	0,99	2000	10	1444.81	78.38
500	0,99	2000	20	1541.95	41.38
500	0,99	3000	5	1445.28	154.90
500	0,99	3000	10	1433.19	80.77
500	0,99	3000	20	1458.30	44.64
500	0,995	1000	5	1463.79	105.16
500	0,995	1000	10	1510.59	58.39
500	0,995	1000	20	1498.24	28.21
500	0,995	2000	5	1457.03	201.99
500	0,995	2000	10	1470.72	107.77
500	0,995	2000	20	1464.42	55.89
500	0,995	3000	5	1426.87	303.91
500	0,995	3000	10	1448.48	152.62
500	0,995	3000	20	1473.20	80.15
500	0,999	1000	5	1481.61	123.18
500	0,999	1000	10	1474.91	67.38
500	0,999	1000	20	1522.93	34.21
500	0,999	2000	5	1478.39	233.25
500	0,999	2000	10	1473.33	129.16
500	0,999	2000	20	1481.72	71.52

500	0,999	3000	5	1464.51	335.08
500	0,999	3000	10	1468.83	182.07
500	0,999	3000	20	1468.49	100.38

Dokumentasi Kode yang digunakan

https://github.com/naufalaschafitz/TugasAkhir

BIODATA PENULIS

Penulis dilahirkan di Bandar Lampung, 27 Juni 2003, merupakan anak pertama dari 4 bersaudara. Penulis telah menempuh pendidikan formal yaitu di TK Miftahul Jannah Semarang, SD Nasima, SDIT Ihsanul Fikri, SMP Ali Maksum, MAN 1 Yogyakarta, dan SMA Al-Azhar 4. Setelah lulus dari SMAN tahun 2021, Penulis mengikuti seleksi SBMPTN dan diterima di Departemen Sistem Informasi FTEIC - ITS pada tahun 2021 dan terdaftar dengan NRP 5026211074.

Penulis memiliki pengalaman profesional sebagai *Human Resource Intern* di Otoritas Jasa Keuangan (OJK) pada awal tahun 2025, di mana penulis berperan dalam proyek-proyek terkait evaluasi sumber daya manusia dan pengembangan sistem internal. Dalam upaya meningkatkan kompetensi, penulis juga telah memperoleh beberapa sertifikasi di bidang teknologi informasi, antara lain Google *Data Analytics Specialization* dan Cisco DevNet Associate, serta mengikuti beberapa program *microlearning* yang berkaitan dengan data analytics dan software engineering.

Di lingkungan kampus, penulis aktif dalam organisasi Himpunan Mahasiswa Sistem Informasi (HMSI) ITS sebagai Kepala Departemen *External Affairs*, serta sebelumnya menjabat sebagai Staf *Research Technologies and Application* HMSI. Penulis juga pernah menjadi Staf *Sponsorship* pada kegiatan Information System Expo (ISE) 2022, menunjukkan keterlibatan dalam berbagai aktivitas pengembangan soft skill dan jejaring profesional. Penulis dapat dihubungi melalui email di naufalaschafitz@gmail.com.