BAB IV
HASIL DAN PEMBAHASAN

4.1 Tinjauan Umum

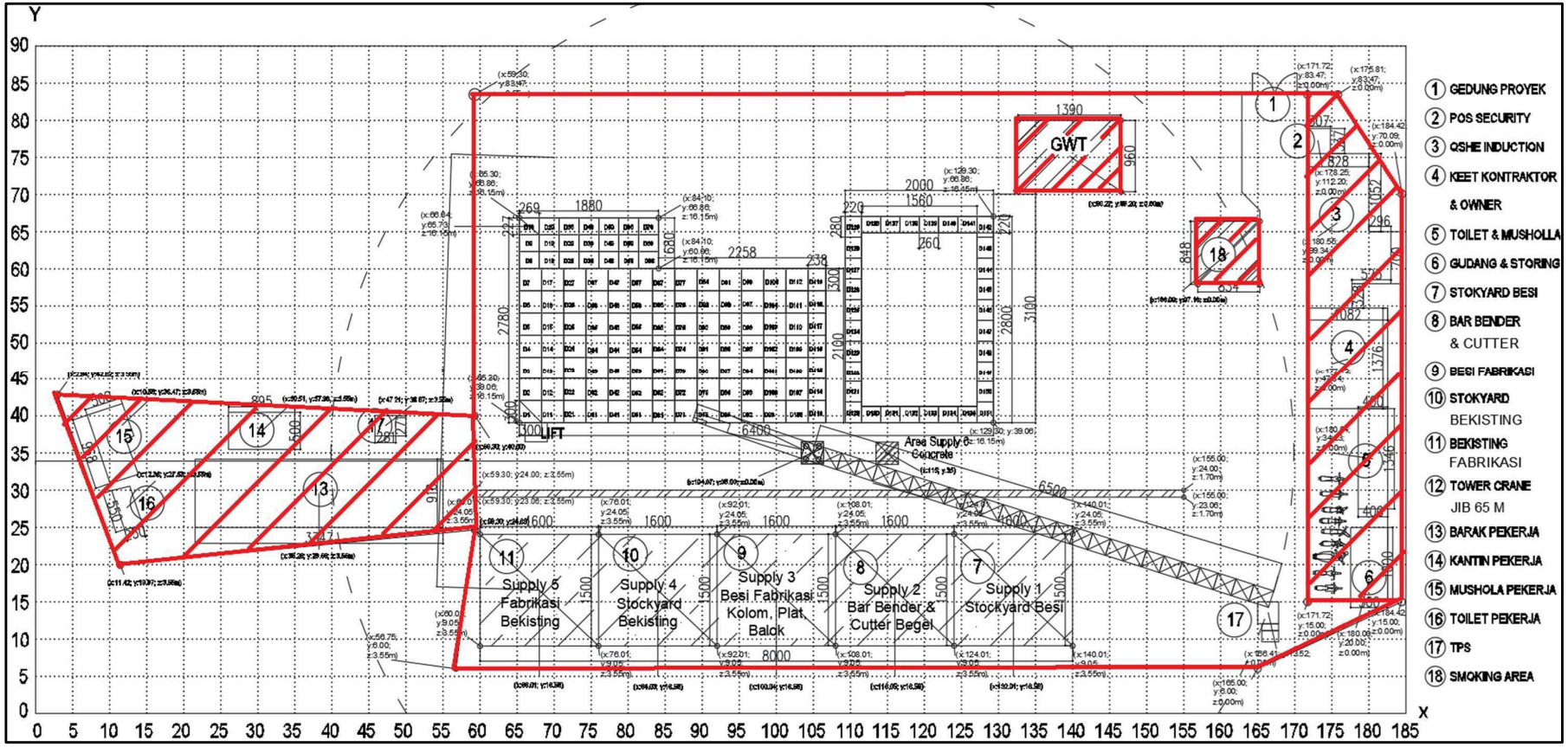
Untuk menganalisis optimasi penggunaan Tower Crane berbasis Genetika Algoritma, diperlukan beberapa tahapan penyelesaian yang melibatkan perhitungan secara teoritis serta mempertimbangkan kondisi aktual di lapangan. Adapun data-data berikut ini yang dibutuhkan untuk mendukung proses analisis tersebut.

4.1.1 Input Data Eksisting

Input Data pada model Eksisting dibagi dalam beberapa data, diantaranya:

a. Batasan-batasan dalam area Lokasi

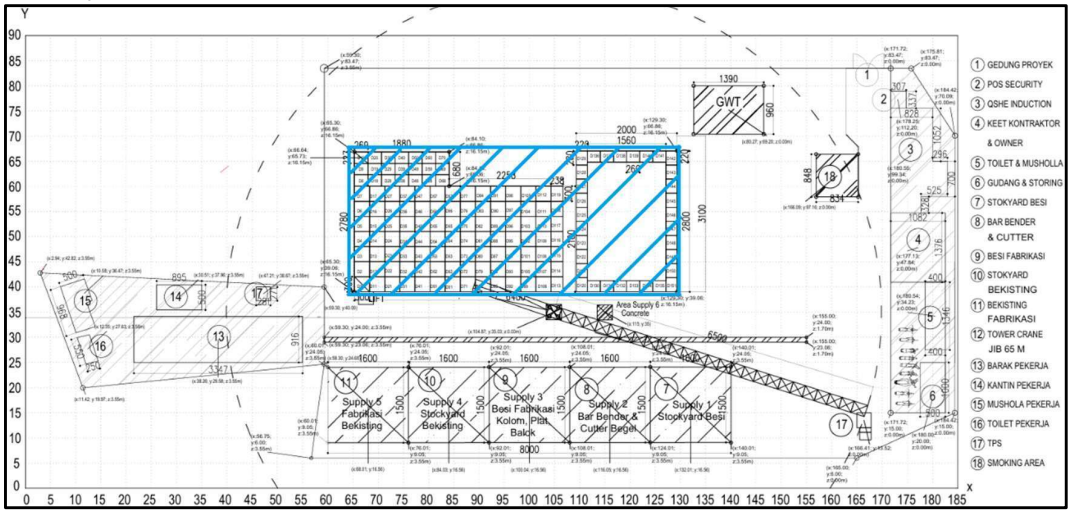
Batasan dalam area proyek konstruksi sangat memengaruhi sebuah perencanaan, desain, dan operasionalnya dikarenakan pada penelitian ini akan dianalisa untuk menyesuaikan proyek dengan kondisi aktual di lapangan. Semua batasan lokasi yang tidak boleh dilanggar harus ditentukan terlebih dahulu sebagai acuan dasar dalam perhitungan dan pengambilan keputusan, memastikan bahwa seluruh kegiatan proyek berjalan selaras dengan kondisi lapangan yang sebenarnya. Dengan demikian, perencanaan dan desain yang dihasilkan akan lebih realistis, efisien, dan meminimalkan risiko kesalahan di kemudian hari. Adapun area Batasan di Lokasi Gedung Kuliah Bersama V Universitas Muhammadiyah Malang diantaranya di lokasi Eksisting ini mencakup area luar lokasi proyek konstruksi, bangunan Gedung yang akan didirikan sebagai acuan titik permintaan (*demand*), letak fasilitas pendukung yang lokasi wilayahnya tidak bisa dilanggar pada kantor karyawan (*direksi keet*), kantin, dan barak pekerja. Berikut ini adalah visualisasi batasan-batasan pada area Lokasi yang ditunjukkan garis arsiran merah pada **Gambar 4.1**.



Gambar 4. 1 Batasan Area Direksi Keet, Barak Pekerja, dan Area Luar Proyek

b. Data *Demand Point*

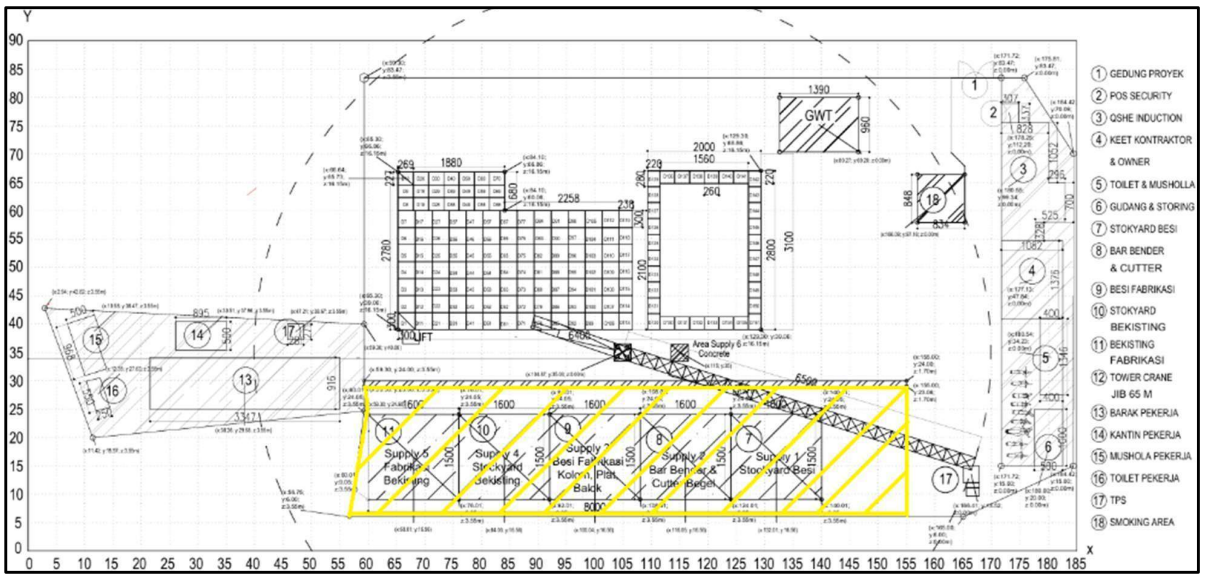
Berdasarkan desain struktur bangunan pada kondisi aktual, data lokasi dari setiap titik permintaan (*demand point*) ditentukan. Setiap titik ini, diberikan nama D = 1, 2, ..., z, yang memiliki koordinat spesifik pada sumbu X dan Y. Untuk menyederhanakan, area lantai bangunan dibagi menjadi beberapa bagian. Titik pusat (*Centroid*) dari setiap titik *demand* kemudian dianggap sebagai representasi dari seluruh bagian tersebut, dan merupakan titik yang harus dilayani dalam perencanaan. Berikut visualisasi titik permintaan (*demand*) pada area Lokasi yang ditunjukkan garis arsiran biru yang dapat dilihat pada **Gambar 4.2**.



Gambar 4. 2 Lokasi *Demand Point*

c. Data *Supply Point*

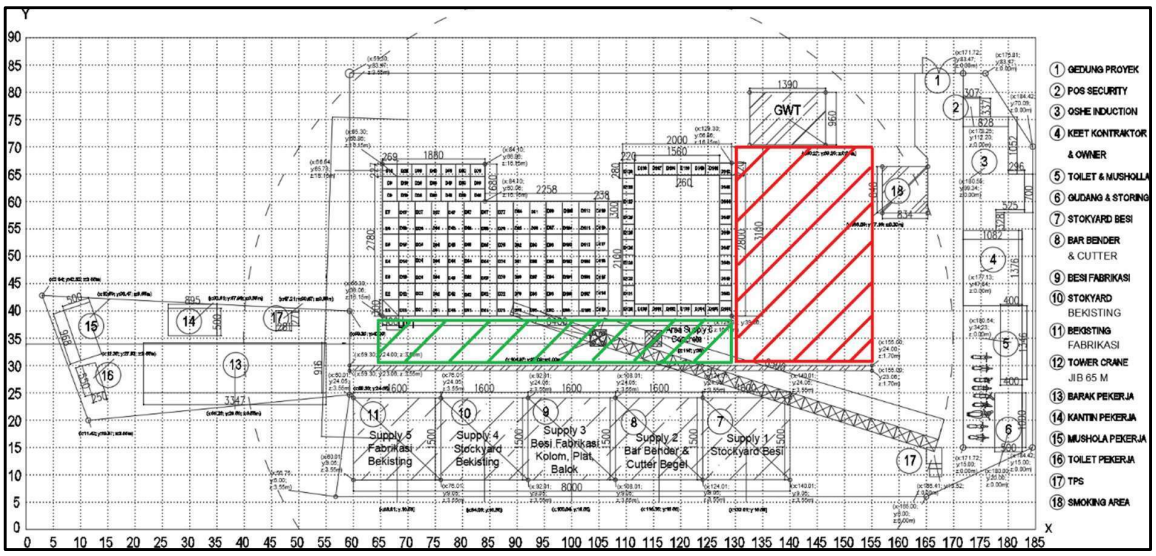
Lokasi untuk area penyimpanan material yang disebut titik suplai (*supply point*). Titik-titik ini digambarkan pada denah proyek. Lokasi-lokasi tersebut tidak boleh berada di area terlarang pada bangunan yang akan didirikan, kantor direksi, atau area barak pekerja seperti ditunjukkan garis arsiran kuning yang dapat dilihat pada **Gambar 4.3**.



Gambar 4. 3 Lokasi Supply Point

d. Data Lokasi Tower Crane

Data lokasi Tower Crane yang dimasukkan berupa koordinat titik penempatan dan jangkauan radiusnya. Letak Tower Crane Eksisting di area proyek merupakan hasil pertimbangan kontraktor pelaksana terhadap efisiensi operasional (jangkauan kerja), aspek keselamatan bangunan, dan kondisi ruang yang tersedia. Letak posisi Tower Crane yang telah ditentukan kemudian dipetakan ke dalam rencana sistem **Koordinat Kartesius Local.** Berdasarkan Site Layout, koordinat Tower Crane Eksisting adalah x:104.87 m, y:35.03 m. Sedangkan Tower Crane yang akan direncanakan jangkauan radiusnya sebesar 60meter. Alasan peneliti memilih jangkauan radius Tower Crane sebesar 60meter dikarenakan pemilihan jarak terjauh antara Tower Crane dengan titik permintaan maupun titik supply seperti **garis arsiran hijau** pada **Gambar 4.4**. Adapun **garis arsiran merah** yang dilabelkan sebagai area transportasi yang tidak bisa di lakukan pemilihan terhadap penempatan Tower Crane.



Gambar 4. 4 Denah Penempatan Lokasi Tower Crane

e. Elevasi lantai pada Proyek

Data Elevasi lantai pada proyek GKB V Universitas Muhammadiyah Malang pada **Tabel 4.2** sebagai berikut:

Tabel 4. 1 Elevasi lantai proyek GKB V Universitas Muhammadiyah Malang

No	Model Penelitian	Tinggi Lantai (m)	Elevasi (m)
1.	Lantai 5	4.2	16.15
2.	Lantai 6	4.2	20.35
3.	Lantai 7	4.2	24.55
4.	Lantai 8	4.2	28.75
5.	Lantai 9	4.2	32.95
6.	Lantai 10	4.2	37.15
7.	Lantai 11	4.2	41.35
8.	Lantai 11 Atap	4.7	46.05

f. Spesifikasi Tower Crane Yang Direncanakan

Tower Crane yang digunakan pada penelitian ini adalah Tower Crane 6520/10 E nomor seri 1012 TC1700650. Dalam pemodelan penelitian ini, radius eksisting awal yang digunakan adalah 65meter dengan kapasitas angkut maksimal 2 ton. Apabila radius tersebut terlalu besar, maka akan disesuaikan menjadi radius dengan rencana 60meter, yang akan meningkatkan kapasitas

angkut menjadi 2,5 ton. Alat berat Tower Crane yang digunakan diharapkan mampu mengoptimalkan dari segi biaya dan waktu dianalisis menggunakan aplikasi MATLAB. Untuk lebih jelasnya, spesifikasi alat yang digunakan pada **Tabel 4.2** atau **Lampiran 1**.

Tabel 4. 2 Spesifikasi Teknis Tower Crane Zoomlion 6520

Tower Crane 6520/10 E seri 1012 TC1700650		
Panjang Jib	Eksisting	Rencana
	65meter	60meter
Kapasitas angkut	2 ton	2,5 ton
Kecepatan Hoisting	80 m/menit	100 m/menit
Kecepatan Slewing	0,7 r/menit	
Kecepatan Trolleying	55 m/menit	
Frekuensi	50Hz	
Daya Listrik – tegangan	415 V – 50 Hz	
Daya mesin	60 kW	

Sumber: (*https://cranemarket.comspecs/tower-cranes/zoomlion/tc6520-10*)

Gambar 4. 5 Tower Crane 6520/10 E nomor seri 1012TC1700650

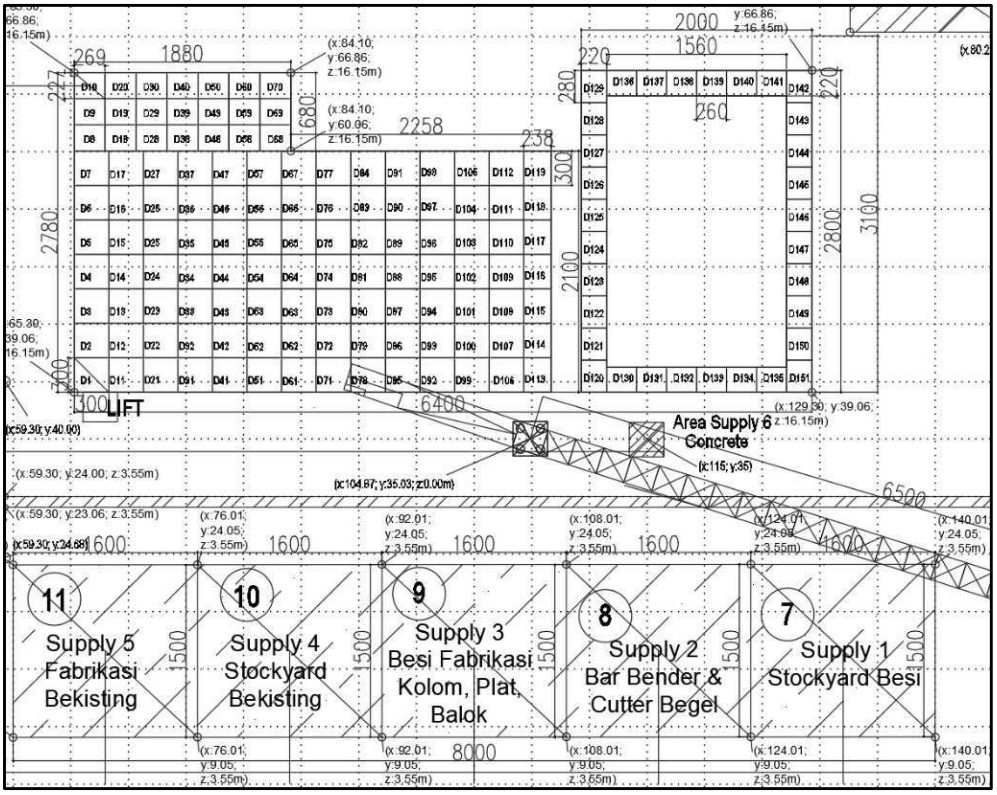
4.1.2 Zonasi Area Penelitian Eksisting

Analisis data difokuskan dengan membagi area penelitian menjadi titik area *Demand* dan Supply yang berbeda-beda. Setiap *Demand* dan *Supply* ini memiliki ukuran spesifik yang tertera pada contoh **Tabel 4.3** dan disajikan lebih lanjut dalam **Lampiran 6**.

Tabel 4. 3 Contoh Titik *Demand* dan *Supply* Eksisting Lantai 5

Lantai	*Demand*	b (m)	d (m)
Lantai 5	D1	3.00	3.00
	D2	3.00	3.00
	D3	3.00	3.00
	D4	3.00	3.00
	D5	3.00	3.00
	D6	3.00	3.00
	D7	3.00	3.00
	D8	2.27	2.69
	D9	2.27	2.69
	D10	2.27	2.69
Stockyard Besi (*Supply* 1)		16.00	15.00

Berikut adalah Visualisasi Denah Penempatan Tower Crane Eksisting, titik *supply* dan titik *demand* ukuran pada **Gambar 4.6**.



Gambar 4. 6 Visualisasi Penempatan Tower Crane dan Area yang dituju

4.2 Langkah-langkah Analisa

4.2.1 Menganalisa Titik Koordinat Tengah

Pencarian titik koordinat merupakan tahapan awal untuk menganalisis area titik supply, titik demand, dan fasilitas lainnya serta memodelkan proses konstruksi. Titik Koordinat *supply* dan *demand* diperoleh dari titik *centroid* atau titik tengah bangun ruang penyusun. Salah satu perhitungan Centroid *demand* 1 lantai 5 seperti berikut:

$$Xc = \frac{b}{2} = \frac{3.00}{2} = 1.50 \text{ m}$$

$$Yc = \frac{d}{2} = \frac{3.00}{2} = 1.50 \text{ m}$$

$$D1x = Xc + jarak\ horizontal\ dari\ sumbu\ y\ ke\ sisi\ d$$

$$= 1.50\ m + 65.30 \text{ m}$$

$= 66.80$ m

$D1y = Yc + jarak\ vertikal\ dari\ sumbu\ x\ ke\ sisi\ b$

$= 1.50\ m + 39.06$ m

$= 40.56$ m

Keterangan:

Yc = Jarak dari tepi ke titik tengah *demand* pada sumbu x (m)

Xc = Jarak dari tepi ke titik tengah *demand* pada sumbu y (m)

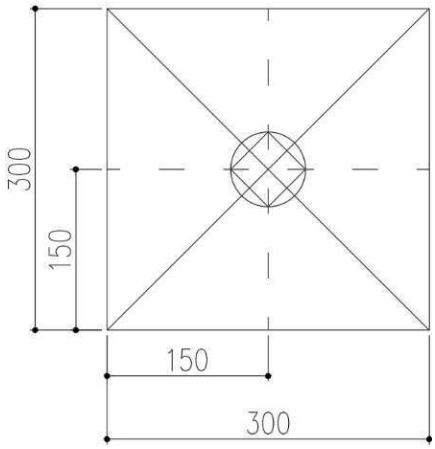
*D*1x = Jarak horizontal dari sumbu y ke centroid *demand*

*D*1y = Jarak vertikal dari sumbu x ke centroid *demand*

b = panjang persegi panjang (m)

d = lebar persegi panjang (m)

Visualisasi Centroid titik *demand* 1 di Lantai 5 disajikan pada **Gambar 4.6**.



Gambar 4. 7 Centroid pada *Demand* 1 Lantai 5

Sebelum analisis dimulai, batas-batas fisik area pencarian dipetakan ke dalam sistem koordinat lokal yang telah ditetapkan. Pemetaan ini menghasilkan sekumpulan titik koordinat yang akan menjadi dasar analisis untuk mengoptimasi. Rincian hasil pemetaan area *Supply dan* Tower Crane pada **Tabel 4.4.**

Tabel 4. 4 Hasil pemetaan area *Supply* dan Tower Crane

Area	Titik UL (kiri atas)		Titik UR (kanan atas)		Titik DL (kiri bawah)		Titik DR (kanan bawah)	
	X	Y	X	Y	X	Y	X	Y
	(m)	(m)	(m)	(m)	(m)	(m)	(m)	(m)
Supply 1	124.01	24.05	140.01	24.05	124.01	9.05	140.01	9.05
Supply 2	108.01	24.05	124.01	24.05	108.01	9.05	124.01	9.05
Supply 3	92.01	24.05	108.01	24.05	92.01	9.05	108.01	9.05
Supply 4	76.01	24.05	92.01	24.05	76.01	9.05	92.01	9.05
Supply 5	60.01	24.05	76.01	24.05	60.01	9.05	76.01	9.05
Supply 6	113.47	36.46	116.47	36.46	113.47	33.46	116.47	33.53
TCO	103.37	36.53	106.37	36.53	103.37	33.53	106.37	33.53

Keterangan:

Supply 1 = Stockyard Besi

Supply 2 = Bar Bender & Cutter Begel

Supply 3 = Besi Fabrikasi Kolom, Plat, dan Balok

Supply 4 = Stockyard Bekisting

Supply 5 = Bekisting Fabrikasi

Supply 6 = Beton Segar

TCO = Tower Crane

Berikut ini terdapat jumlah titik *Demand* point (D) untuk setiap lantai pada **Tabel 4.5**.

Tabel 4. 5 Jumlah titik *Demand point* (D) untuk setiap lantai

Lantai	Jumlah *Demand* (D)
Lantai 5	151 titik

Lantai	Jumlah *Demand* (D)
Lantai 6	151 titik
Lantai 7	126 titik
Lantai 8	119 titik
Lantai 9	119 titik
Lantai 10	75 titik
Lantai 11	57 titik
Lantai 11 Atap	12 titik

4.2.2 Kuantitas Material

Kuantitas Material yang digunakan pada penelitian ini berupa beton segar (*concrete*), besi tulangan (*reinforcement*), dan bekisting (*formwork*). Kuantitas didapatkan dari hasil volume pekerjaan yang berada pada **Lampiran 6** seperti pada hasil **Tabel 4.6.**

Tabel 4. 6 Kuantitas Material

No	Material yang digunakan	Kuantitas (ton)
1	Beton Segar (*Concrete*)	8585.34778
2	Besi Tulangan (*Reinforcement*)	237.30774
3	Bekisting (*Formwork*)	754.72700

4.2.3 Siklus Material

Pada siklus terhadap pengiriman sebuah material ditentukan oleh kapasitas angkut alat berat Tower Crane dengan radius area kerja yang dipakai. Dalam penelitian ini, radius area yang dipakai pada eksisting Tower Crane sebesar 65meter dengan kapasitas 2ton dan Rencana sebesar 60meter dengan kapasitas 2.5ton dengan mempertimbangkan sebagai alternatif jika radius eksisting ternyata kelebihan jangkauan, dan kapasitas angkat menjadi perhatian utama untuk mendapatkan kapasitas angkat yang lebih besar dan efisiensi operasional yang lebih baik. Semakin pendek radius kerja Tower Crane, semakin besar stabilitasnya dan semakin aman operasinya, terutama saat mengangkat beban

berat. Meskipun perbedaan 5meter mungkin tidak terlihat drastis, dalam kondisi operasional radius area kerja yang lebih pendek dapat memberikan margin keamanan yang lebih baik dan mengurangi risiko kecelakaan. Berikut ini analisa pengiriman material dalam penelitian ini dengan menggunakan cara:

1. Untuk menganalisis **siklus pengiriman material** dalam satuan ton, kami mengacu pada data kuantitas material keseluruhan dari kapasitas angkut maksimum tower crane. Hasil perhitungan material dan kapasitas *crane* ini disajikan dalam **Tabel 4.8** dengan mengikuti langkah-langkah sebagai berikut:

$$NS1 = \frac{QS1}{C65}$$

$$= \frac{8585.34778}{2\ ton} = 4292.674\ kali$$

$$= 4293\ kali$$

Keterangan:

*NS*1 = jumlah siklus

*QS*1 = kuantitas muatan yang dibawa (ton)

C65 = kapasitas muatan *Tower Crane* Eksisting (ton)

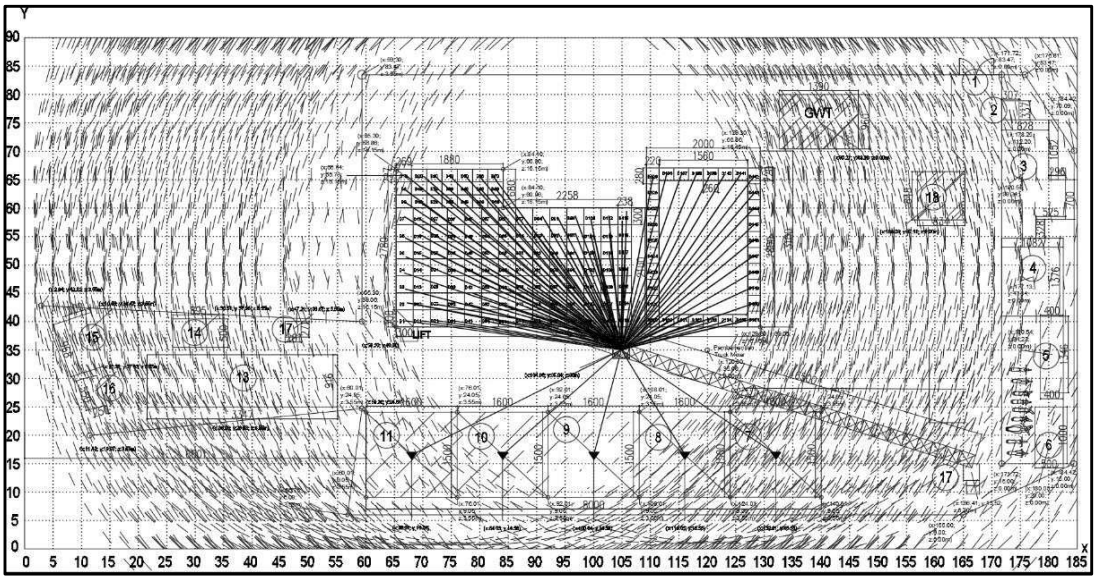
Tabel 4. 7 Siklus pengiriman material besi tulangan, beton segar, dan bekisting

No.	Material	Notasi	Jumlah siklus pengiriman	
			65 m (Eksisting) (kali)	60 m (Rencana) (kali)
1	Beton Segar (*Concrete*) (A1)	S6	4293	3434
2	Besi Tulangan (*Reinforcement*) (A2)	S1 S2 S3	119	95
3	Bekisting (*Formwork*) (A3)	S4 S5	377	302

4.3 Analisa dan Model pada penempatan Tower Crane Eksisting

4.3.1 Eksisting Radius Area Kerja Tower Crane 65 meter

Jangkauan Tower Crane eksisting dianggap memadai jika mampu mencakup semua titik suplai dan permintaan. Area jangkauan spesifik untuk Tower Crane dengan radius 65meter seperti **Gambar 4.8** dan juga detail gambar pada **Lampiran 5.**



Gambar 4. 8 Feasible Area pada Tower Crane Eksisting dengan radius 65m

4.4 Analisa dan Model Numerik Posisi Tower Crane Eksisting

a. Analisa Waktu Hoisting (Tv)

Berikut ini adalah contoh analisa waktu *Hoisting* untuk titik permintaan pada Lantai 5 Tv1 yang dilakukan**.**

Diketahui:

- Jumlah demand lantai 5 = 151 titik
- Ketinggian lantai 5 = 16.15 meter
- Kecepatan *Hoisting* = 80 m/menit (spesifikasi alat berat)
- Material per Kolom1 besi = 831.11 kg = 0.83 ton

1. Waktu *Hoisting*

$$Tv = \frac{Zh5}{Vh} = \frac{jarak\ hoisting\ lantai\ 5}{Kecepatan\ Hoisting\ Eksisting} = \frac{16.15}{80} = 0.202\ menit$$

Tv = 0.202 $menit$ x Jumlah titik Demand untuk lantai 5

= 0.202 $menit\ x$ 151

= 30.48 $menit$

2. Daya Angkut Waktu Angkat Besi Tulangan

Waktu angkat Besi Tulangan = berat material per kolom / waktu siklus

= 0.83 ton / 0.202 menit

= 4.11 ton/menit

b. Analisa Waktu *Slewing* (T*w*)

1. Analisa Jarak-jarak Radial

Berikut ini analisa jarak-jarak radial waktu slewing pada Lantai 5 di titik *Demand* 1.

Diketahui:

- Posisi titik *demand* pada sumbu x = 66.80 m
- Posisi titik *demand* pada sumbu y = 40.56 m
- Posisi titik *supply* pada sumbu x = 132.01 m
- Posisi titik *supply* pada sumbu y = 16.56 m
- Posisi titik *Tower Crane* pada sumbu x = 104.87 m
- Posisi titik *Tower Crane* pada sumbu y = 35.03 m

$$\rho(Di1) = \sqrt{(XDi - XCr)^2 + (YDi - YCr)^2}$$

$$= \sqrt{(Demand1\ x\ lt.5 - Crane\ x)^2 + (Demand1\ y\ lt.5 - Crane\ y)^2}$$

$$= \sqrt{(66.80\ m - 104.87\ m)^2 + (40.56\ m - 35.03\ m)^2}$$

$$= 38.47\ m$$

$$\rho(Si1) = \sqrt{(XSi - XCr)^2 + (YSi - YCr)^2}$$

$$= \sqrt{(Supply1\ x\ lt.5 - Crane\ x)^2 + (Supply1\ y\ lt.5 - Crane\ y)^2}$$

$$= \sqrt{(132.01\ m - 104.87\ m)^2 + (16.56\ m - 35.03\ m)^2}$$

$$= 32.83\ m$$

2. Analisa Jarak-jarak Horizontal

Proses Kedua ialah analisa awal waktu *slewing* menghitung jarak-jarak horizontal dari titik *supply* ke titik *demand (li)* pada Lantai 5 di titik *Supply* 1.

Diketahui:

- Posisi titik centroid *demand* sumbu x = 66.80 m
- Posisi titik *demand* pada sumbu y = 40.56 m
- Posisi titik *supply* pada sumbu x = 132.01 m
- Posisi titik *supply* pada sumbu y = 16.56 m

$$li1 = \sqrt{(XDi - XSi)^2 + (YDi - YSi)^2}$$

$$= \sqrt{(D1x\ lt.5 - Supply1\ x)^2 + (D1y\ lt.5 - Supply1\ y)^2}$$

$$= \sqrt{(66.80 - 132.01)^2 + (40.56 - 16.56)^2}$$

$$= 69.49\ m$$

3. Besar Perpindahan Jib

Proses Ketiga ialah analisa menghitung besar perpindahan jib (θ) dengan didasarkan pada data jarak yang telah dihitung sebelumnya.

Diketahui:

- Jarak Horizontal Supply 1 pada li1 = 69.49 m
- Jarak radial *demand* ke titik Tower Crane = 38.47 m
- Jarak radial *supply* ke titik Tower Crane = 32.83 m

$$\theta = arc\ cos\left(\frac{li1^2 - \rho(Di1)2.\rho(Si1)^2}{2.\rho(Di1).\rho(Si1)}\right)$$

$$= arc\ cos\left(\frac{69.49^2 - 38.47^2 - 32.83^2}{2\ x\ 38.47\ x\ 32.83}\right)$$

$$= 0.45\ rad$$

4. Analisa Waktu Slewing (T*w*)

Proses Perhitungan waktu putar (*slewing*) dilakukan setelah semua data penting seperti jarak radial, jarak horizontal, dan perpindahan jib yang didapatkan dari analisa awal. Salah satu contoh analisis waktu putar Tw1 untuk *supply* 1 pada lantai 5 adalah sebagai berikut:

Diketahui:

- Kecepatan slewing = 0.6 r/menit
- Besar perpindahan jib = 0.45 rad

$$Tw = \frac{1}{w} * arc\cos\ (\theta)$$

$$= \frac{1}{0.6} * 0.45$$

$$= 0.76\ menit$$

c. Analisa Waktu *Trolleying* (Ta)

Salah satu contoh analisis waktu Trolleying (Ta1) pada *supply*1 lantai 5 sebagai berikut:

Diketahui:

- Jarak radial *demand* ke titik Tower Crane = 38.47 m
- Jarak radial *supply* ke titik Tower Crane = 32.83 m
- Kecepatan Trolleying = 55 m/menit

$$Ta = \frac{|\rho(Dj) - \rho\ (Sj)|}{Va}$$

$$= \frac{|Jarak\ radial\ Demand\ ke\ TC - Jarak\ radial\ Supply\ ke\ TC|}{Kecepatan\ Trolleying}$$

$$= \frac{|38.47 - 32.83|}{55}$$

$$= 0.10 \text{ menit}$$

4.5 Hasil Waktu Eksisting *Hoisting*, *Slewing*, dan *Trolleying*

Berikut ini adalah hasil dari rekapitulasi dari Eksisting waktu *Hoisting, Slewing*, dan *Trolleying* pada **Tabel 4.8** dan hasil keseluruhan bisa dilihat pada **Lampiran 6.**

Tabel 4. 8 Hasil Eksisting Waktu Pergerakan

No	Waktu Pergerakan	Eksisting
1	Waktu *Hoisting*	811.61 menit
2	Waktu *Slewing*	7913.52 menit
3	Waktu *Trolleying*	1061.58 menit

4.6 Analisis Waktu Siklus Kerja alat berat Tower Crane (TT)

Analisis waktu kerja pengangkatan material menunjukkan bahwa total waktu operasional Tower Crane yang ada saat ini pada **Tabel 4.9** berikut ini:

$$TT = \sum_{m=1}^{M} (2Tv + 2Tw + 2Ta) + N)$$

$$= ((2 * 811.61 + 2 * 7913.52 + 2 * 1061.58) + 4293) +$$

$$(2 * 811.61 + 2 * 7913.52 + 2 * 1061.58) + 119) +$$

$$(2 * 811.61 + 2 * 7913.52 + 2 * 1061.58) + 377))$$

$$= 63.508{,}95 \text{ menit}$$

$$= 1.058{,}48 \text{ jam}$$

Tabel 4. 9 Detail Analisa Waktu Kerja Total Tower Crane Eksisting

No	Keterangan	Jumlah Siklus (kali)	Waktu siklus Pergerakan (menit)	Waktu Pengiriman Material (menit)
1	Beton Segar (*Concrete*)	4293	19.573,42	23.866,09
2	Besi Tulangan (*Reinforcement*)	119		19.692,07
3	Bekisting (*Formwork*)	377		19.950,78
Total Waktu Pengiriman				63.508,95

4.7 Analisis Biaya

4.7.1 Biaya Bahan Bakar Tower Crane

1. Perhitungan kebutuhan bahan bakar solar Tower Crane per jam didasarkan pada beberapa faktor. Mengacu pada data dari (Rostiyanti, 2008), koefisien bahan bakar yang digunakan adalah **0,04**. Daya mesin Tower Crane adalah **60 hp** (*https://cranemarket.comspecs/tower-cranes/zoomlion/tc6520-10*), dan faktor penggunaan alat sebesar **0,83** (Kementrian Pekerjaan Umum dan Perumahan Rakyat, 2022). Berikut adalah rincian perhitungannya:

   Fuel = *fcoef x hp x Fa*

   = 0.04 x 60 x 0.83

   = 1.992 galon/jam = 7.540 liter/jam

2. Kebutuhan total solar Tower Crane. Berdasarkan waktu kerja total Tower Crane eksisting selama 1.058,48 jam dan konsumsi solar sebesar 7.540 liter/jam maka total kebutuhan solar adalah yang dihitung seperti berikut:

   T*Fuel* = *Fuel* x Total Waktu

   = 7.540 liter/jam x 1.058.48 jam

   = 7.980,96 liter

3. Biaya solar total. Dengan harga solar industri sebesar Rp.15.000 per liter, total biaya solar yang dibutuhkan untuk mengoperasikan Tower Crane selama periode tersebut dihitung seperti berikut:

   Biaya solar = T*Fuel* x Harga Solar per liter

   = 7.980,96liter x Rp.15.000,00 per liter

   = Rp 119.714.376,78

4.7.2 Analisis Biaya Sewa Tower Crane

Biaya sewa Tower Crane dihitung berdasarkan durasi total penggunaan tower crane (dalam jam) dengan biaya sewa per jam. Dengan biaya sewa bulanan Rp.90.737.362 dan dikonversi menjadi biaya per jam Rp. 126.024 (AHSP KOTA MALANG 2024). Perhitungannya adalah sebagai berikut:

Biaya Sewa = TT x biaya sewa per jam

= 1.058,48 jam x Rp. 126.024

= Rp. 133.394.326,36

4.7.3 Analisis Biaya Tenaga Kerja Operator dan Spotter

Analisis biaya tenaga kerja mencakup perhitungan total gaji yang harus dibayarkan kepada operator dan spotter disesuaikan dengan total waktu operasional Tower Crane. Berikut rincian perhitungannya pada **Tabel 4.10**.

Tabel 4. 10 Analisa Upah pekerja Operator dan Spotter Tower Crane Eksisting

No.	Pekerja	Jumlah Pekerja (orang)	Durasi Kerja (jam)	Gaji per jam	Jumlah Gaji
1	Operator	2	1.058,48	Rp. 17.240,00	Rp. 36.496.478,44
2	Spotter	2		Rp. 15.298,00	Rp. 32.385.332,20
				Total Gaji	Rp. 68.881.810,64

4.8 Analisa dan Model Genetika Algoritma Tower Crane Rencana

4.8.1 *Input Initial* Parameter Genetika Algoritma

Pseudo-code (pengkodean) dapat dibuat berdasarkan seluruh tahapan analisis Genetika Algoritma yang telah diperoleh. **Gambar 4.9** menunjukkan hasil tahapan Genetika Algoritma, di mana urutan pertama adalah Input Parameter. Analisa lengkap dapat dilihat pada **Lampiran 8**.

Jumlah populasi = 200 (npop)
Panjang individu (bit) = 98 (nindiv)
Jumlah Iterasi = 1000 (niter)
Jumlah Elit = 20 (nelits)
Probabilitas *Crossover* = 0.7 (pcross)
Laju Mutasi = 1/nindiv (nmutat)
Jumlah titik *Crossover* = nindiv/2 (ncross)

```
%=========================================================================
% OPTIMASI_WAKTU_DAN_BIAYA_PENGGUNAAN_TOWER_CRANE_BERBASIS_GENETIKA
% ALGORITMA_PADA_PROYEK_PEMBANGUNAN_GKB_V_UNIVERSITAS_MUHAMMADIYAH_MALANG
% Tugas_Akhir_code_by_Akbar_Fitra_Jauhari_(2121055)
%=========================================================================

clear
clc

%% Input Parameter %%
% Initial Parameters
npop = 200;          % Number of individuals in the population
nindiv = 98;         % Length of individuals
niter = 1000;        % Number of iteration
nelits = 20;         % Number of elites
pcross = 0.7;        % Crossover probability
rmutat = 1/nindiv;   % Mutation rate
ncross = nindiv/2;   % Number of crossover point

% Rencana Tower Crane Parameters
tcr = 60;            % Rencana Tower Crane radius (m)
```

Gambar 4. 9 *Pseudo-code Input Initial* Parameter

4.8.2 *Input* Parameter Tower Crane

Parameter input untuk alat berat Tower dimasukkan setelah parameter Genetika Algoritma selesai diinput. Setelah seluruh data diperoleh, proses dapat dilanjutkan ke penyusunan *Pseudo-code*, seperti ditunjukkan pada **Gambar 4.10**.

Rencana Radius Tower Crane = 60 meter (tcr)
Material Handling Paramater = 100 m/min (kecepatan *Hoisting*)

= 0.7 rad/min (kecepatan *Slewing*)

= 55 m/min (kecepatan *Trolleying*)

Jumlah siklus Beton = 3434 (ncs1)

Jumlah siklus Besi Tulangan = 95 (ncs2)

Jumlah siklus Bekisting = 302 (ncs3)

```
% Rencana Tower Crane Parameters
tcr = 60;            % Rencana Tower Crane radius (m)

% Material Handling Parameters
vh = 100;            % Hoisting speed (100 m/min)
vw = 0.7;            % Slewing speed (0.7 rad/min)
va = 55;             % Trolley speed (55 m/min)

% Material Weights
ncs1 = 3434;         % Jumlah siklus Beton (Concrete)
ncs2 = 95;          % Jumlah siklus Baja Tulangan (Reinforcement)
ncs3 = 302;          % Jumlah siklus Bekisting (Formwork)

% Initial Conditions
hheight = 0;          % Initial height
ttotal  = 0;          % Initial total time
```

Gambar 4. 10 *Pseudo-code Input* Parameter Tower Crane

Selanjutnya ialah *input* koordinat pada tinggi bangunan per lantai dan Koordinat *Centroid* Titik *Demand* per lantai seperti **Gambar 4.11.**

```
% Coordinates Input
h5       = 16.15; %Satuan meter
h6       = 20.35; %Satuan meter
h7       = 24.55; %Satuan meter
h8       = 28.75; %Satuan meter
h9       = 32.95; %Satuan meter
h10      = 37.15; %Satuan meter
h11      = 41.35; %Satuan meter
h11atap  = 46.05; %Satuan meter
h_alls   = {h5, h6, h7, h8, h9, h10, h11, h11atap};

% Centroid Coordinates (data yang sudah ada)
dxdy_lantai_11_atap = [75.30 40.16; 78.10 40.16; 80.90 40.16; 75.40 42.56; 75.40 45.36; 75.40 48.36; 78.40 42.56; 78.40 45.36
dxdy_lantai_11 = [63.80 42.56; 63.80 45.36; 63.80 48.36; 63.80 51.36; 63.80 54.36; 63.80 57.36; 69.80 42.56; 69.80 45.36; 69.
dxdy_lantai_10 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.75 49.31; 66.75 51.94; 66.75 54.71; 66.75 57.48; 69.80 40.56; 69.
dxdy_lantai_9 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.80 49.56; 66.80 52.56; 66.80 55.56; 66.80 58.56; 66.64 61.19; 66.6
dxdy_lantai_8 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.80 49.56; 66.80 52.56; 66.80 55.56; 66.62 58.44; 66.62 61.19; 66.6
dxdy_lantai_7 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.80 49.56; 66.80 52.56; 66.80 55.56; 66.80 58.56; 66.73 61.39; 66.7
dxdy_lantai_6 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.80 49.56; 66.80 52.56; 66.80 55.56; 66.80 58.56; 66.64 61.19; 66.6
dxdy_lantai_5 = [66.80 40.56; 66.80 43.56; 66.80 46.56; 66.80 49.56; 66.80 52.56; 66.80 55.56; 66.80 58.56; 66.64 61.19; 66.6
dxdy_all = {dxdy_lantai_5, dxdy_lantai_6, dxdy_lantai_7, dxdy_lantai_8, dxdy_lantai_9, dxdy_lantai_10, dxdy_lantai_11, dxdy_l
```

Gambar 4. 11 *Pseudo-code* Parameter koordinat dan Centroid Koordinat Titik *Demand* per lantai

4.8.3 Pembuatan Fungsi (*function*) Inisiasi Kode Biner

Pembuatan Fungsi (*function*) digunakan dalam pemodelan untuk mempermudah penulisan dan pengulangan kode. Selain itu, fungsi beberapa langkah dijalankan berulang kali dengan *input* yang berbeda selama tahapannya sama. Insiasi Kode Biner bertujuan menerjemahkan kromosom menjadi nilai yang akan dipakai dalam analisis. *Pseudo code* Inisiasi Kode Biner dapat dilihat pada **Gambar 4.12**.

```
% Binary Encoding
M = zeros(7, 1);
for n = 7:-1:1                    % Input individu length
    M(n,1) = 2^(7-n);
end

% Initialize storage for results
best_fitness_history = zeros(1, niter);
avg_fitness_history = zeros(1, niter);
best_idx_history = zeros(1, niter);
best_coordinates_history = zeros(niter, 14);
```

```
%% Generate Initial Population %%
for iter = 1:niter
    % First population (if iter == 1)
    if iter == 1
        for i = 1:npop
            indiv = randi([0,1], 1, nindiv);
            fitval = newfitness_value_complete(indiv, M, h_alls, dxdy_all, tcr, vh, vw, va, ncs1, ncs2, ncs3);
            population(i).individu = indiv;
            population(i).newfitness_value_complete = fitval;
        end
    else
        % Check if each offspring exists and concatenate accordingly
            if exist('offspring', 'var')
                combined_population = [population offspring mutated_offspring];
            else
                combined_population = [population mutated_offspring];
            end
```

Gambar 4. 12 *Pseudo-code function* Inisiasi Kode Biner

4.8.4 Iterasi

Untuk menjalankan pengulangan kode (*looping*), parameter yang sudah diinput di awal dipakai pada setiap iterasi yang sudah ditetapkan.

4.8.5 Pengkodean (*Fungsi get_coordinates*)

Proses pengkodean dari setiap variabel menjadi representasi biner dengan panjang bit tertentu. Panjang bit ini akan menentukan solusi yang dihasilkan. Proses ini diimplementasikan melalui fungsi get_coordinates seperti pada contoh **Gambar 4.13**.

```
%% Function Definitions %%
function coordinates = newget_coordinates(indiv, M)
% Inisialisasi koordinat
    x1 = 132.01 + (indiv(:,1:7) * M) * 16 / (2^7-1); % Absis reinforcing steel
    y1 = 16.56 + (indiv(:,8:14) * M) * 15 / (2^7-1); % Ordinat reinforcing steel
    x2 = 116.05 + (indiv(:,15:21) * M) * 16 / (2^7-1); % Absis reinforcing steel
    y2 = 16.56 + (indiv(:,22:28) * M) * 15 / (2^7-1); % Ordinat reinforcing steel
    x3 = 100.04 + (indiv(:,29:35) * M) * 16 / (2^7-1); % Absis reinforcing steel
    y3 = 16.56 + (indiv(:,36:42) * M) * 15 / (2^7-1); % Ordinat reinforcing steel
    x4 = 84.03 + (indiv(:,43:49) * M) * 16 / (2^7-1); % Absis formwork
    y4 = 16.56 + (indiv(:,50:56) * M) * 15 / (2^7-1); % Ordinat formwork
    x5 = 68.01 + (indiv(:,57:63) * M) * 16 / (2^7-1); % Absis formwork
    y5 = 16.56 + (indiv(:,64:70) * M) * 15 / (2^7-1); % Ordinat formwork
    x6 = 115 + (indiv(:,71:77) * M) * 3 / (2^7-1); % Concrete
    y6 = 35 + (indiv(:,78:84) * M) * 3 / (2^7-1); % Concrete
    x7 = 104.87 + (indiv(:,85:91) * M) * 3 / (2^7-1); % Absi
    y7 = 35.03 + (indiv(:,92:98) * M) * 3 / (2^7-1); % Ordinat TC

    % Nilai batas bawah untuk setiap koordinat
    upper_bounds = [2.94, 42.82, 59.30, 40.00, 59.30, 24.00, 155.00, 24.00, 59.30,
    intervals = [8.48, 22.85, 0, 15.32, 0, 0.94, 0, 0.94, 0, 2.55, 77.47, 6.72, 77

    % Nilai kecil untuk perbandingan floating-point
    epsilon = 1e-14;

    % Memastikan semua koordinat tidak sama dengan batas bawahnya
    for i = 1:size(indiv, 1)
        coords = [x1(i), y1(i), x2(i), y2(i), x3(i), y3(i), x4(i), y4(i), x5(i), y
        for n = 1:14
            if abs(coords(n) - upper_bounds(n)) < epsilon
                indiv(i, (n-1)*7+1:n*7) = randi([0, 1], 1, 7);
                coords(n) = upper_bounds(n) + (indiv(i, (log-1)*7+1:n*7) * M) * in
            end
        end
        % Update koordinat dengan nilai yang baru
        [x1(i), y1(i), x2(i), y2(i), x3(i), y3(i), x4(i), y4(i), x5(i), y5(i), x6(
    end

    % Output koordinat
coordinates = [x1, y1, x2, y2, x3, y3, x4, y4, x5, y5, x6, y6, x7, y7];
end
```

Gambar 4. 13 Pengkodean *get_coordinates*

4.8.6 Evaluasi Nilai *fitness*

Evaluasi pada Nilai *fitness* merupakan *objective function* yang didapatkan dari analisa Waktu *Hoisting*, Waktu *Slewing*, dan aktu *Trolleying*. *Pseudo-code* evaluasi nilai fitnes bisa dilihat pada contoh **Gambar 4.14** hingga **Gambar 4.17**.

```
function fitval = newfitness_value_complete(indiv, M, h_alls, dxdy_all, tcr, vh, vw, va, ncs1, ncs2, ncs3);
    % Get coordinates
    coordinates = newget_coordinates(indiv, M);

   %% Kodingan fungsi fitness_value
  % Initialize fitval to a default value
    fitval = 0;

    % Penalty coefficient
    penalty_coeff = 1e7;

    % Initialize penalty term
    penalty = 0;

    % Initialize default value for saving
    tw1 = 0;
    tw2 = 0;
    tw3 = 0;
    tw4 = 0;
    tw5 = 0;
    tw6 = 0;

    % Initialize accumulated TV value
    accumulated_tv = 0;
```

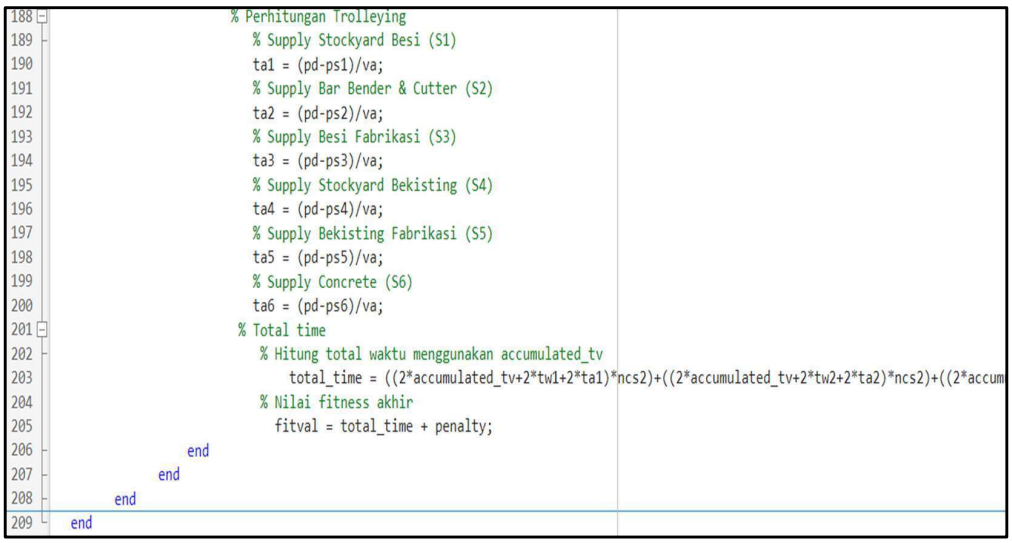
Gambar 4. 14 Fungsi Fitness_Value

```
function fitval = newfitness_value_complete(indiv, M, h_alls, dxdy_all, tcr, vh, vw, va, ncs1, ncs2, ncs3);
    % Get coordinates
    coordinates = newget_coordinates(indiv, M);

   %% Kodingan fungsi fitness_value
% Initialize fitval to a default value
    fitval = 0;

    % Penalty coefficient
    penalty_coeff = 1e7;

    % Initialize penalty term
    penalty = 0;

    % Initialize default value for saving
    tw1 = 0;
    tw2 = 0;
    tw3 = 0;
    tw4 = 0;
    tw5 = 0;
    tw6 = 0;

    % Initialize accumulated TV value
    accumulated_tv = 0;
```

Gambar 4. 15 Analisa Waktu Hoisting

```
        % Looping Centroid Floor 5 to 11atap
            for c = 1:size(dxdy_current, 1)
                dx = dxdy_current(c, 1); % X coordinate
                dy = dxdy_current(c, 2); % Y coordinate
                % Perhitungan Waktu Slewing
                  % Supply Stockyard Besi (S1)
                     pd = sqrt((dx - coordinates(13))^2 + (dy - coordinates(14))^2);
                     ps1 = sqrt((coordinates(:,1) - coordinates(:,13)).^2 + (coordinates(:,2) - coordinates(:,14)).^2);
                     if any (pd > tcr) || any (ps1 > tcr)
                        violation_pd = pd - tcr;
                     % Jalankan Violation
                       violation_ps1 = ps1 - tcr;
                       violation = violation_pd + violation_ps1;
                       penalty = penalty + penalty_coeff * violation^2;
                     else
                        pd = pd;
                        ps1 = ps1;
                     end
                        li1 = sqrt((dx-coordinates(:,1)).^2+(dy-coordinates(:,2)).^2);
                     if li1 + ps1 > pd && ps1 + pd > li1 && pd + li1 > ps1
                        teta1 = acos((li1^2 - pd^2 - ps1^2) / (2*pd*ps1));
                     else
                        gradienpd = (dy + coordinates(:,14)) ./ (dx + coordinates(:,13));
                        gradienps1 = (coordinates(:,2) + coordinates(:,14)) ./ (coordinates(:,1) + coordinates(:,13));
                        teta1 = atan((gradienpd + gradienps1) / (1 + gradienpd .* gradienps1));
                     end
                     tw1 = teta1/vw;
                     supply_1 = [pd ps1 li1 teta1 tw1];
```

Gambar 4. 16 Analisa Waktu *Slewing*



```
                    % Perhitungan Trolleying
                       % Supply Stockyard Besi (S1)
                       ta1 = (pd-ps1)/va;
                       % Supply Bar Bender & Cutter (S2)
                       ta2 = (pd-ps2)/va;
                       % Supply Besi Fabrikasi (S3)
                       ta3 = (pd-ps3)/va;
                       % Supply Stockyard Bekisting (S4)
                       ta4 = (pd-ps4)/va;
                       % Supply Bekisting Fabrikasi (S5)
                       ta5 = (pd-ps5)/va;
                       % Supply Concrete (S6)
                       ta6 = (pd-ps6)/va;
                    % Total time
                       % Hitung total waktu menggunakan accumulated_tv
                           total_time = ((2*accumulated_tv+2*tw1+2*ta1)*ncs2)+((2*accumulated_tv+2*tw2+2*ta2)*ncs2)+((2*accum
                       % Nilai fitness akhir
                         fitval = total_time + penalty;
                end
            end
        end
    end
```

Gambar 4. 17 Analisa Waktu *Trolleying*

4.8.7 Elitisme

Fungsi *run_elitisme* menjalankan proses *Elitisme* sebelum seleksi induk. Hal ini bertujuan untuk mengulang pemilihan kromosom secara efisien dan menjaga kode utama tetap bersih dengan memisahkannya. Berikut ini terdapat *Pseudo-code Elitisme* seperti **Gambar 4.18**.

```
function elite_individuals = newrun_elitism(population, nelits)
    % Get the number of individuals in the population
    pop_size = length(population);

    % Initialize variables to store elite individuals and their fitness values
    elite_individuals = struct('individu', {}, 'newfitness_value_complete', {});
    elite_fitness = zeros(1, nelits);

    % Find the elite individuals
    for i = 1:nelits
        best_fitness = Inf;
        best_index = 0;

        for j = 1:pop_size
            if population(j).newfitness_value_complete < best_fitness
                best_fitness = population(j).newfitness_value_complete;
                best_index = j;
            end
        end

        elite_individuals(i).individu = population(best_index).individu;
        elite_individuals(i).newfitness_value_complete = population(best_index).newfitness_value_complete;
        elite_fitness(i) = best_fitness;

        % Remove the selected elite individual from the population
        population(best_index) = [];
        pop_size = pop_size - 1;
    end
end
```

Gambar 4. 18 *Pseudo-code Elitisme*

4.8.8 Seleksi Induk

Untuk menjadi induk penyilangan, kromosom dalam populasi dipilih menggunakan teknik roda *roulette*. Pertama untuk memperluas area pencarian lewat pemilihan acak. Kedua untuk memberikan peluang lebih besar kepada kromosom dengan nilai *fitness* yang lebih tinggi dibandingkan *fitness*-nya rendah. *Pseudo-code* khusus untuk seleksi induk dengan metode roda roulette dalam Genetika Algoritma seperti **Gambar 4.19**.

```
function parents = newget_parents(population)

    fitness_data = zeros(1, length(population));

    % Calculate the fitness values for each individual
    for i = 1:length(population)
        fitness_data(i) = population(i).newfitness_value_complete;
    end

    % Convert fitness values to selection probabilities
    selection_prob = 1 ./ fitness_data;
    total_prob = sum(selection_prob);
    selection_prob = selection_prob / total_prob;

    % Perform roulette wheel selection for parent 1
    cumulative_prob = cumsum(selection_prob);
    r1 = rand();
    selected_index1 = find(r1 <= cumulative_prob, 1, 'first');

    % Perform roulette wheel selection for parent 2
    r2 = rand();
    selected_index2 = find(r2 <= cumulative_prob, 1, 'first');

    % Ensure that the two selected parents are different
    while selected_index1 == selected_index2
        r2 = rand();
        selected_index2 = find(r2 <= cumulative_prob, 1, 'first');
    end

    % Return the selected parents
    parents = [population(selected_index1), population(selected_index2)];

end
```

Gambar 4. 19 *Pseudo-code* Seleksi Induk dan *Roulette Wheel*

4.8.9 *Crossover* (Penyilangan) dan Mutasi

Penyilangan (*crossover*) dilakukan untuk menghasilkan kromosom baru (anak) dari kromosom induk dengan harapan dapat meningkatkan nilai *fitness* populasi. Sementara itu, mutasi bertujuan untuk menyegarkan dan menjaga keragaman populasi. Pemodelan ini menerapkan mutasi *bit flipping* yang membalikkan bit terpilih pada kromosom. Baik penyilangan maupun mutasi terjadi berdasarkan probabilitas: probabilitas mutasi (pmutat) dan probabilitas penyilangan (1/npop atau 0,01). *Pseudo-code crossover* (penyilangan) dapat dilihat pada **Gambar 4.20**, dan mutasi pada **Gambar 2.21 dan Gambar 2.22**.

```
function offspring = newnpoint_crossover(parents, nindiv, ncross)

    % Extract parent1 and parent2 from the parents struct
    parent1 = parents(1).individu;
    parent2 = parents(2).individu;

    % Get the length of the parents
    len = nindiv;

    % Generate n random crossover points
    crossover_points = sort(randperm(len-1, ncross));

    % Initialize the offspring
    offspring1 = parent1;
    offspring2 = parent2;

    % Perform crossover
    for i = 1:ncross
        if mod(i, 2) == 1
            % Odd-indexed segments: swap genetic material
            if i == 1
                offspring1(1:crossover_points(i)) = parent2(1:crossover_points(i));
                offspring2(1:crossover_points(i)) = parent1(1:crossover_points(i));
            else
                offspring1(crossover_points(i-1)+1:crossover_points(i)) = parent2(crossover_points(i-1)+1:crossover_points(i));
                offspring2(crossover_points(i-1)+1:crossover_points(i)) = parent1(crossover_points(i-1)+1:crossover_points(i));
            end
        end
    end

    % Swap the last segment if ncross is odd
    if mod(ncross, 2) == 1
        offspring1(crossover_points(end)+1:end) = parent2(crossover_points(end)+1:end);
        offspring2(crossover_points(end)+1:end) = parent1(crossover_points(end)+1:end);
    end

    % Create the offspring struct array
    offspring = struct('individu', {}, 'newfitness_value_complete', {});
    offspring(1).individu = offspring1;
    offspring(2).individu = offspring2;
end
```

Gambar 4. 20 *Pseudo-code crossover* (penyilangan)

```
function mutated_offspring = newbitflip_mutation(offspring, rmutat)

    % Initialize mutated_offspring struct array
    mutated_offspring = struct('individu', {}, 'newfitness_value_complete', {});

    % Perform bitflip mutation for each offspring
    for i = 1:length(offspring)
        % Get the length of the individual
        len = length(offspring(i).individu);

        % Generate random numbers for each bit
        rand_nums = rand(1, len);

        % Create a mask for bits to be mutated
        mutation_mask = rand_nums < rmutat;

        % Perform bitflip mutation
        mutated_individu = offspring(i).individu;
        mutated_individu(mutation_mask) = ~mutated_individu(mutation_mask);

        % Assign mutated individual to mutated_offspring struct array
        mutated_offspring(i).individu = mutated_individu;
        mutated_offspring(i).newfitness_value_complete = offspring(i).newfitness_value_complete;
    end

end
```

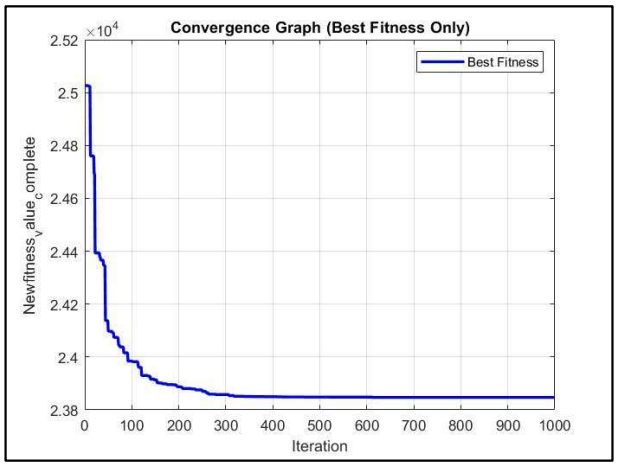
Gambar 4. 21 *Pseudo-code* Mutasi

```
function mutated_offspring = newbitflip_mutation_parents(parents, rmutat)
    % Extract parent1 and parent2 from the parents struct
    offspring1 = parents(1).individu;
    offspring2 = parents(2).individu;

    % Get the length of the individuals
    len1 = length(offspring1);
    len2 = length(offspring2);

    % Generate random numbers for each bit of offspring1
    rand_nums1 = rand(1, len1);
    % Create a mask for bits to be mutated in offspring1
    mutation_mask1 = rand_nums1 < rmutat;
    % Perform bitflip mutation on offspring1
    mutated_offspring1 = offspring1;
    mutated_offspring1(mutation_mask1) = ~offspring1(mutation_mask1);

    % Generate random numbers for each bit of offspring2
    rand_nums2 = rand(1, len2);
    % Create a mask for bits to be mutated in offspring2
    mutation_mask2 = rand_nums2 < rmutat;
    % Perform bitflip mutation on offspring2
    mutated_offspring2 = offspring2;
    mutated_offspring2(mutation_mask2) = ~offspring2(mutation_mask2);

    % Create the mutated offspring struct
    mutated_offspring = struct('individu', {}, 'fitness_value', {});
    mutated_offspring(1).individu = mutated_offspring1;
    mutated_offspring(2).individu = mutated_offspring2;
    mutated_offspring(1).fitness_value = parents(1).fitness_value;
    mutated_offspring(2).fitness_value = parents(2).fitness_value;
end
```

Gambar 4. 22 *Pseudo-code* Mutasi *Parents*

4.9 Hasil (Output) Matrix Laboratory (MATLAB)

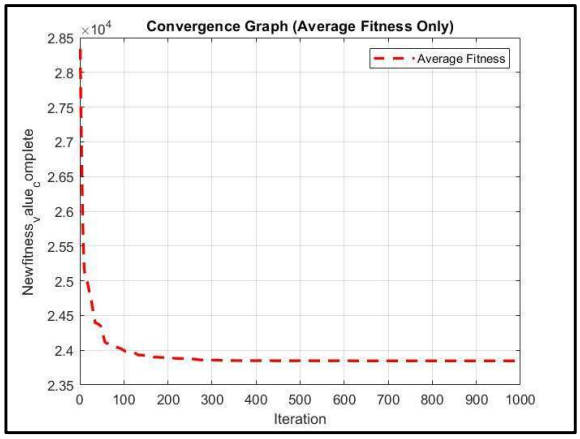
Berdasarkan hasil yang dijalankan atau Run menggunakan Genetika Algoritma pada Matrix Laboratory (MATLAB) dari kodingan Rencana dituangkan dalam bentuk grafik. Algoritma berhasil menemukan solusi terbaik pada iterasi ke-612 dengan hasil *Output* analisa Genetika Algoritma dapat dilihat pada **Lampiran 9**. Visualisasi Grafik pada Gambar **4.23** menunjukkan bahwa algoritma telah berjalan dengan baik.



Gambar 4. 23 Grafik Nilai Fitness

4.10 Nilai Fitness Rata-Rata

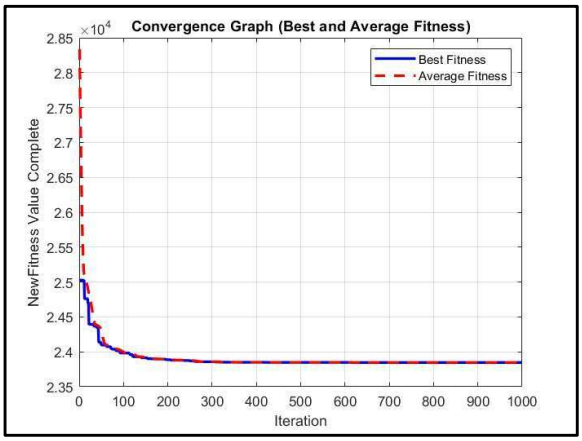
Nilai rata-rata pada iterasi awal menunjukkan penurunan yang signifikan. Penurunan nilai Fitness pada iterasi ke-21 dan ke-22, yakni 24760.73 dan 24393.58, disebabkan oleh banyaknya individu yang melanggar batasan jarak-jarak radial. Grafik pada **Gambar 4.24** menunjukkan bahwa nilai fitness rata-rata memiliki perubahan yang signifikan dari nilai terbesar ke terkecil.



Gambar 4. 24 Grafik Nilai Fitness Rata-rata

4.11 Konvergensi Nilai Fitness

Genetika Algoritma dalam penelitian ini mencapai solusi optimal pada iterasi ke-612, ditunjukkan oleh stabilnya nilai *fitness* rata-rata dan terbaik. **Gambar 4.25** memvisualisasikan proses konvergensi yang membuktikan efektivitas solusi parameter yang digunakan.



Gambar 4. 25 Grafik Konvergensi Utama

4.12 Analisa Total Waktu Kerja alat berat Tower Crane (TT)

Analisis waktu kerja pengangkatan material menunjukkan bahwa total waktu operasional Tower Crane yang ada saat ini pada **Tabel 4.11** berikut ini:

$$TT = \sum_{m=1}^{M} (2Tv + 2Tw + 2Ta) + N$$

$$= ((18.537{,}03 + 3434) + (18.537{,}03 + 95) + (18.537{,}03 + 302))$$

$$= 59.442{,}05 \text{ menit}$$

$$= 990{,}70 \text{ jam}$$

Tabel 4. 11 Detail Analisis Total waktu kerja Tower Crane

No	Material	Jumlah Siklus	Waktu Siklus Pergerakan dari GA	Waktu Pengiriman Material
		(kali)	(menit)	(menit)
1	Beton Segar	3434	18.537,03	21.971,17
2	Besi Tulangan	95		18.631,96
3	Bekisting	302		18.838,92
Total Waktu Pengiriman				59.442,05

4.13 Analisis Rencana Biaya

4.13.1 Analisa Biaya Bahan Bakar

1. Perhitungan kebutuhan bahan bakar solar Tower Crane per jam didasarkan pada beberapa faktor. Mengacu pada data dari (Rostiyanti, 2008), koefisien bahan bakar yang digunakan adalah **0,04**. Daya mesin Tower Crane adalah **60 hp** (*https://cranemarket.comspecs/tower-cranes/zoomlion/tc6520-10*), dan faktor penggunaan alat sebesar **0,83** (Kementrian Pekerjaan Umum dan Perumahan Rakyat, 2022). Berikut adalah rincian perhitungannya:

   Fuel = *fcoef x hp x Fa*

   = 0.04 x 60 x 0.83

= 1.992 galon/jam

= 7.540 liter/jam

2. Kebutuhan total solar Tower crane. Berdasarkan waktu kerja total Tower Crane Rencana selama 990.70 jam dan konsumsi solar sebesar 7.540 liter/jam maka total kebutuhan solar adalah yang dihitung seperti berikut:

   T*Fuel* = *Fuel* x Total Waktu

   = 7.540 x 990,70 jam

   = 7.469,88 liter

3. Biaya solar total. Dengan harga solar industri sebesar Rp.15.000 per liter, total biaya solar yang dibutuhkan untuk mengoperasikan tower crane adalah seperti berikut:

   Biaya solar = T*Fuel* x Harga Solar per liter

   = 7.469,88liter x Rp.15.000 per liter

   = Rp 112.048.268,04

4.13.2 Analisa Biaya Sewa Tower Crane

Biaya sewa Tower Crane dihitung berdasarkan durasi total penggunaan tower crane (dalam jam) dikalikan dengan biaya sewa per jam. Dengan biaya sewa bulanan sebesar Rp.90.737.362,35 dikonversi menjadi biaya per jam Rp. 126.024,11 (AHSP KOTA MALANG 2024). Perhitungannya adalah sebagai berikut:

Biaya Sewa = Total Waktu x biaya sewa per jam

= 990,70 x Rp. 126.024,11

= Rp. 124.852.199,35

4.13.3 Analisa Biaya Pekerja *Operator* dan *Spotter*

Analisis biaya tenaga kerja mencakup perhitungan total gaji yang harus dibayarkan kepada operator dan spotter disesuaikan dengan total waktu operasional Tower Crane. Berikut rincian perhitungan untuk pekerja pada **Tabel 4.12**.

Tabel 4. 12 Analisa Upah pekerja Operator dan Spotter Tower Crane Eksisting

No.	Pekerja	Jumlah Pekerja (orang)	Durasi Kerja (jam)	Gaji per jam	Jumlah Gaji
1	Operator	2	990.70	Rp. 17.240,00	Rp. 34.159.365,89
2	Spotter	2		Rp. 15.298,00	Rp. 30.311.483,72
				Total Gaji	Rp. 64.470.849,61

4.13.4 Validasi Model & Pembahasan

Hasil dari pengkodingan menunjukkan bahwa koordinat yang dihasilkan tidak sepenuhnya sama dengan data Eksisting. Terdapat pergeseran penempatan posisi Tower Crane sebesar 2.92meter dari posisi eksisting dan untuk penempatan area supply terdapat pergeseran koordinat yang berbeda beda yang disajikan pada **Tabel 4.13**.

Tabel 4. 13 Perbandingan Koordinat yang dihasilkan

Variabel	Nama	Centroid Koordinat				Pergeseran koordinat X
		Eksisting (m)		Rencana (m)		
		X	Y	X	Y	(m)
Stockyard Besi	S1	132.01	16.56	133.51	16.56	1.50
Bar Bender & Cutter	S2	116.05	16.56	117.51	16.56	1.46
Besi Fabrikasi	S3	100.04	16.56	101.51	16.56	1.47
Stockyard Bekisting	S4	84.03	16.56	85.51	16.56	1.48
Bekisting Fabrikasi	S5	68.01	16.56	69.51	16.56	1.50
Beton Segar	S6	115	35	115.09	35.00	0.09
Tower Crane	TCO	104.87	35.03	107.56	36.17	2.92

Hasil optimasi penempatan posisi Tower Crane menunjukkan peningkatan efisiensi yang signifikan. Secara keseluruhan, selisih waktu kerja total rencana dan eksisting berkurang 6.40%. Pengurangan waktu ini berdampak pada berkurangnya total jam kerja Tower Crane untuk menyelesaikan semua pengiriman material.

Perbandingan waktu siklus dari kedua kondisi tersebut dapat dilihat pada **Tabel 4.14.**

Tabel 4. 14 Selisih antara waktu pada kondisi eksisting dan Hasil Optimasi

No	Waktu Pergerakan	Waktu		Selisih (%)
		Eksisting	Rencana	
1	Waktu *Hoisting*	811.61 menit	649.29 menit	20.00
2	Waktu *Slewing*	7913.52 menit	8056.31 menit	1.77
3	Waktu *Trolleying*	1061.58 menit	1140.63 menit	6.93
4	Waktu Siklus	19573.42 menit	18537.03 menit	5.29
5	Waktu Kerja Total (jam)	1058.48 jam	990.70 jam	6.40

Setelah itu, terdapat perbandingan biaya yang dapat dilihat pada **Tabel 4.15**.

Tabel 4. 15 Selisih antara biaya pada kondisi eksisting dan Hasil Optimasi

No	Biaya	Kondisi		Selisih (%)
		Eksisting (Rp)	Rencana (Rp)	
1	Biaya Bahan Bakar Solar	119.714.376,78	112.048.268,04	6.40
2	Biaya Sewa Tower Crane	133.394.326,36	124.852.199,35	6.40
3	Biaya Gaji Operator	36.496.478,44	34.159.365,89	6.40
	Biaya Gaji Spotter	32.385.332,20	30.311.483,72	6.40
4	Total Biaya	321.990.513,78	301.371.317,01	6.40