

TUGAS AKHIR

PEMBUATAN PROGRAM HITUNG PERATAAN PARAMETER UNTUK POLIGON MENGUNAKAN BORLAND C++ BUILDER



**Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Pendidikan Sarjana Strata Satu Bidang Teknik Geodesi**

Disusun oleh :

**YUANITA KRISMA MILANI
01.25.012**

**JURUSAN TEKNIK GEODESI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
INSTITUT TEKNOLOGI NASIONAL
MALANG
2006**

**PEMBUATAN PROGRAM HITUNG PERATAAN PARAMETER
UNTUK POLIGON MENGGUNAKAN BORLAND C++ BUILDER**

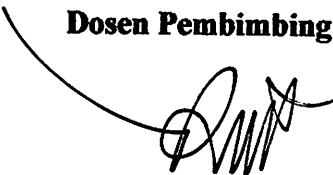
TUGAS AKHIR

*Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Pendidikan Sarjana Strata Satu (S-1)
Bidang Teknik Geodesi*

**Oleh
YUANITA KRISMA MILANI
0125012**

Menyetujui

Dosen Pembimbing I



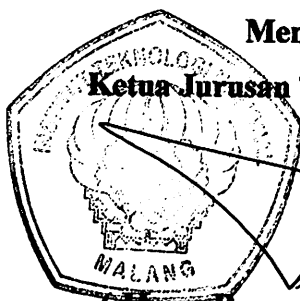
(Ir. Rinto Sasongko, MT)

Dosen Pembimbing II



(Ir. Moh. Nurhadi, MT)

Mengetahui



Ketua Jurusan Teknik Geodesi S-1

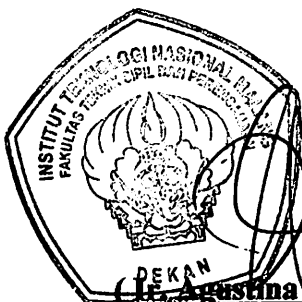
11/10/06

(Hery Purwanto, ST, MSc)

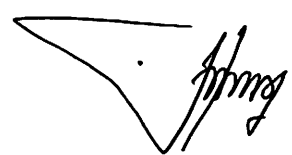
Dipertahankan di depan Panitia Penguji Tugas Akhir Jurusan Teknik Geodesi, Fakultas Teknik Sipil dan Perencanaan, Institut Teknologi Nasional Malang, dan diterima untuk memenuhi salah satu syarat guna memperoleh gelar Sarjana Strata Satu (S-1) Bidang Teknik Geodesi

Pada hari / tanggal : Rabu, 27 September 2006

Panitia Ujian Tugas Akhir


Ketua

(Ir. Agustina Nhrul Hidayati, MTP)
Dekan F. T. S. P

Sekretaris

(Hery Purwanto, ST, MSc)
Ketua Jurusan Teknik Geodesi S-1

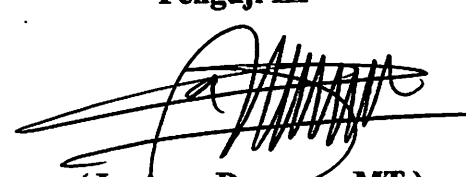
Anggota Penguji

Penguji I

(Ir. Rinto Sasongko, MT)

Penguji II

(Hery Purwanto, ST, MSc)

Penguji III

(Ir. Agus Darpeno, MT)

KATA PENGANTAR

Puji syukur kami panjatkan kehadiran Allah SWT karena atas segala limpahan Rahmat dan Bimbingan-Nya lah penulis dapat menyelesaikan Tugas Akhir dengan judul **“Pembuatan Program Hitung Perataan Parameter Untuk Poligon Menggunakan Borland C++ Builder”** yang diajukan sebagai salah satu syarat untuk menyelesaikan Program Pendidikan Sarjana Strata Satu (S-1), Bidang Teknik Geodesi di Institut Teknologi Nasional Malang.

Terselesainya laporan tugas akhir ini tidak terlepas dari bantuan moril, dorongan, serta bimbingan dari semua pihak. Untuk itu penulis menyampaikan banyak terima kasih kepada :

1. Bapak Prof. Dr. Eng. Ir. Abraham Lomi, MSEE selaku Rektor Institut Teknologi Nasional Malang
2. Ibu Ir. Agustina Nurul Hidayati, MTP selaku Dekan Fakultas Teknik Sipil dan Perencanaan, Institut Teknologi Nasional Malang
3. Bapak Hery Purwanto, ST, MSc selaku Ketua Jurusan Teknik Geodesi S-1
4. Bapak Ir. Rinto Sasongko, MT selaku Dosen Pembimbing I
5. Bapak Ir. Moh. Nurhadi, MT selaku Dosen Pembimbing II
6. Bapak Christian T. Siahaan, ST selaku Dosen Wali sekaligus Sekretaris Jurusan Teknik Geodesi S-1
7. Dosen-dosen Jurusan Teknik Geodesi dan bagian recording Institut Teknologi Nasional Malang
8. Pihak keluarga atas segalanya

9. Teman-teman angkatan 2001 atas dukungan, bantuan, kerjasama, dan motivasinya ... Semoga tetap kompak ya
10. Teman-teman Pusteg (pengurus lama maupun baru) atas support, bantuan, advise, dan semangatnya selama ini Untuk kedepannya semoga Pusteg lebih maju lagi.....
11. Semua pihak yang telah membantu terselesainya tugas akhir ini

Dalam penyusunan Tugas Akhir ini masih banyak kekurangan-kekurangan yang harus diperbaiki. Untuk itu, penulis mengharapkan segala bentuk masukan dan koreksi untuk kemajuan bersama. Semoga Tugas Akhir ini dapat bermanfaat bagi kita semua sebagai tambahan informasi dan pengetahuan.

Malang, Oktober 2006

Penulis

LEMBAR PERSEMBAHAN

Subhanallah.....

Subhanallah.....

Subhanallah.....

Tiada kata yang dapat diucapkan selain memuji Nama Mu ...Ya Allah.....

Syukur Alhamdulillah..... atas segala nikmat yang Engkau berikan

Shalawat dan salam senantiasa tercurah kepada Nabi besar

Muhammad SAW, keluarga, dan para sahabatnya

"The man just can try but Allah SWT knows what the best thing for us so do something by ikhlas hoping Ridho Allah SWT"

This is dedicated to :

- ✦ All of My Family ... Ayahanda dan Ibunda tercinta (Ir. Andik Suparmin, MM and Endang Dwi Hernani).. Terima kasih untuk cinta, doa, pengorbanan, dan support dalam masa-masa penantian yang melelahkan pada studi ananda... Semoga kado kecil ini dapat sedikit membuat kalian tersenyum
To My Twins Brother (Yoki 'n Yoka) thanks ya.... Hopping that Allah always gives you the best thing in your life
- ✦ Kel Belajarku (Jati 'n Ugi) thanks so much atas semangat, support, pengalaman2 selama ini..... (Ga terasa qt dah satu kelompok mulai semester dua lho) Jat.... Selamat ya smg dpt kerjaan sesuai keinginanmu... Buat Ugi cpt selesaian TA nya jgn proyek melulu.... Ayo, kamu pasti bisa... kok

- ✦ My Best Friends (Yani 'n Ifa) Smg prsahabatan qt sampai nanti....thanks for all. Dengerin aq curhat, support 'n advisenya, kenangan2nya....Yan, focus dulu ke TA ya lupain dulu abangmu... ntar jg balik sendiri Semangat ya, cpt lulus. Fa, smg langgeng sm Mas Aeng, cpt selesaian TA Insyaallah ntar tak bantu kok..
- ✦ To My Partner in Praktikum (Deni) akhirnya qt lulus juga ya.. Sorry sering ngrepotin.. Thanks untuk support 'n cerita2nya... Ga nyangka ternyata kamu dewasa juga...
- ✦ To Angkatan 2001 Kel Kedung (jati,ugi,deni,awenk,dani,lanang, yani,santi,'n christien) Thanks utk cerita2nya slm di Kepanjen... To Kel Kepanjen (Desi,Asni,Tini,Novi,Amin dkk), Kel Penarukan (Ifa,Lina,Roni dkk), Kel Cepoko (Tyas,Danang dkk) thanks for all, guys Buat pasangan '01 (Desi-Sukron, Roni-Lina, Tyas-Danang, Amin-Tini) smg langgeng sampai nanti
- ✦ To Pusteg's Family (Mas Win,Aan,Pius,Pipo,Hani,Saf,Yani,M Ermi,Ifa,,Enggar,Siti) thanks for all
- ✦ To Angk '98 (Mas Ruli dkk), '00 (Mbak Yuyun dkk), '02 (Duddy dkk), '03 (Indra dkk), '04 (Desi dkk) thanks doanya
- ✦ Pak Rinto makasih bimbingannya, Pak Hery thanks bukunya
- ✦ Untuk Bapak2 Dosen Geodesi makasih banyak atas bimbingannya
- ✦ For all thanks so much 'n sorry kalo nita banyak khilafnya... (Maaf kalo ga bisa sebutin satu persatu)

For the last Don't forget me..... if you need help, please come to my house on Jin. Ikan Arwana N-15 Malang

DAFTAR ISI

	HALAMAN
LEMBAR PERSETUJUAN	i
LEMBAR PENGESAHAN	ii
KATA PENGANTAR	iii
LEMBAR PERSEMBAHAN	iv
DAFTAR ISI	vi
DAFTAR GAMBAR	x

BAB I PENDAHULUAN

I.1. Latar Belakang	1
I.2. Rumusan Masalah	2
I.3. Tujuan Penelitian	3
I.4. Batasan Masalah	3
I.5. Manfaat Penelitian	3
I.6. Tinjauan Pustaka	3
I.7. Metode Penelitian	5

BAB II DASAR TEORI

II.1. Kerangka Kontrol Horisontal (KKH)	6
---	---

II.1.1.	Klasifikasi KKH	7
II.1.2.	Metode Pengukuran Poligon	9
II.2.	Hitung Perataan KKH	12
II.2.1.	Hitung Perataan Kuadrat Terkecil	13
II.2.2.	Hitung Perataan Parameter	15
II.2.3.	Tahapan Hitung Perataan Parameter Poligon	23
II.2.3.1.	Poligon Terbuka Terikat Sempurna	23
II.2.3.2.	Poligon Tertutup	34
II.2.3.3.	Ellips Kesalahan	47
II.3.	Borland C++ Builder	49
II.3.1	Menu Utama C++ Builder	50
II.3.1.1.	Menu File	50
II.3.1.2.	Menu Edit	52
II.3.1.3.	Menu Search	53
II.3.1.4.	Menu View	54
II.3.1.5.	Menu Project	55
II.3.1.6.	Menu Run	56
II.3.1.7.	Menu Component	57
II.3.1.8.	Menu Database	58
II.3.1.9.	Menu Tools	59
II.3.1.10	Menu Help	60
II.3.2.	Menu Toolbar	60

BAB III PELAKSANAAN PENELITIAN

III.1. Data Penelitian	63
III.2. Peralatan Penelitian	66
III.3. Diagram Alir Penelitian	67
III.4. Pembuatan Program	70
III.4.1. Rancangan Program	71
III.4.2. Rancangan Tampilan (<i>Design Interface</i>)	82

BAB IV HASIL DAN ANALISA

IV.1. Tampilan Program Hitung Perataan Poligon	90
IV.1.1 Menu File	91
IV.1.1.1. Sub Menu New	92
IV.1.1.2. Sub Menu Open	94
IV.1.1.3. Sub Menu Open as Bitmap	95
IV.1.1.4. Sub Menu Save as Bitmap	95
IV.1.1.5. Sub Menu Print	96
IV.1.1.6. Sub Menu Setting Printer	96
IV.1.1.7. Sub Menu Exit	97
IV.1.2 Menu Hasil Perhitungan	97
IV.1.3 Menu Option	99
IV.1.4 Menu Help	101
IV.2. Analisa Hasil Hitung Perataan Poligon	103

IV.3. Uji Ketelitian	109
IV.3.1 Poligon Terbuka Terikat Sempurna	109
IV.3.2 Poligon Tertutup	117
IV.3.2.1. Dengan Sudut Dalam	117
IV.3.2.2. Dengan Sudut Luar	121

BAB V PENUTUP

V.1. Kesimpulan	126
V.2. Saran	129

DAFTAR PUSTAKA

LAMPIRAN

DAFTAR GAMBAR

HALAMAN

BAB II DASAR TEORI

Gambar 2.1	Poligon Terbuka Terikat Sempurna	10
Gambar 2.2	Poligon Tertutup dengan Sudut Dalam	11
Gambar 2.3	Poligon Tertutup dengan Sudut Luar	11
Gambar 2.4	Linierisasi Jarak	17
Gambar 2.5	Linierisasi Persamaan Pengukuran Sudut	19
Gambar 2.6	Poligon Terbuka Terikat Sempurna	23
Gambar 2.7	Poligon Tertutup dengan Sudut Dalam	35
Gambar 2.8	Poligon Tertutup dengan Sudut Luar	36
Gambar 2.9	Ellips Kesalahan.....	48
Gambar 2.10	Tampilan Program C++ Builder	49
Gambar 2.11	Menu Utama	50
Gambar 2.12	Menu File	50
Gambar 2.13	Menu Edit	53
Gambar 2.14	Menu Search	54
Gambar 2.15	Menu View	55
Gambar 2.16	Menu Project	56
Gambar 2.17	Menu Run	57

Gambar 2.18	Menu Component	58
Gambar 2.19	Menu Database	59
Gambar 2.20	Menu Tools	59
Gambar 2.21	Menu Help	60
Gambar 2.22	Menu Toolbar	60
Gambar 2.23	Komponen Standard	61

BAB III PELAKSANAAN PENELITIAN

Gambar 3.1	Poligon Terbuka Terikat Sempurna	63
Gambar 3.2	Poligon Tertutup dengan Sudut Dalam	64
Gambar 3.3	Poligon Tertutup dengan Sudut Luar	65
Gambar 3.4	Diagram Alir Penelitian	67
Gambar 3.5	Diagram Alir Poligon Terbuka Terikat Sempurna	72
Gambar 3.6	Diagram Alir Poligon Tertutup	76
Gambar 3.7	Diagram Alir Paket Program	80
Gambar 3.8	Form Depan (<i>Cover</i>)	83
Gambar 3.9	Form Menu Utama	84
Gambar 3.10	Form Menu File	85
Gambar 3.11	Form Menu Hasil Perhitungan	86
Gambar 3.12	Form Menu Option	87
Gambar 3.13	Form Menu Help	88
Gambar 3.14	Label dan Edit	89

BAB IV HASIL DAN ANALISA

Gambar 4.1	Form Depan (<i>Cover</i>).....	90
Gambar 4.2	Form Menu Utama	91
Gambar 4.3	Form Menu File	92
Gambar 4.4	Form Sub Menu New	93
Gambar 4.5	Form Input Data Poligon Terbuka Terikat Sempurna	93
Gambar 4.6	Form Input Data Poligon Tertutup	94
Gambar 4.7	Form Sub Menu Open	94
Gambar 4.8	Form Sub Menu Open as Bitmap	95
Gambar 4.9	Form Sub Menu Save as Bitmap	95
Gambar 4.10	Form Sub Menu Print	96
Gambar 4.11	Form Sub Menu Setting Printer	96
Gambar 4.12	Form Sub Menu Exit	97
Gambar 4.13	Form Menu Hasil Perhitungan	97
Gambar 4.14	Form Hasil Poligon Terbuka Terikat Sempurna	98
Gambar 4.15	Form Hasil Poligon Tertutup Sudut Dalam	98
Gambar 4.16	Form Hasil Poligon Tertutup Sudut Luar	99
Gambar 4.17	Form Menu Option	99
Gambar 4.18	Form Sub Menu Font	100
Gambar 4.19	Form Sub Menu Background Color	100
Gambar 4.20	Form Sub Menu Foreground Color	101
Gambar 4.21	Form Menu Help	101
Gambar 4.22	Form Sub Menu Petunjuk Penggunaan	102

Gambar 4.23	Form Sub Menu About	102
Gambar 4.24	Hasil Test β_1 Poligon Terbuka Terikat Sempurna.....	103
Gambar 4.25	Hasil Test β_1	104
Gambar 4.26	Hasil Test β_2 (dengan matriks P)	104
Gambar 4.27	Hasil Test β_2 (tanpa matriks P)	105
Gambar 4.28	Hasil Test β_2 (dengan matriks P)	105
Gambar 4.29	Hasil Test β_2 (tanpa matriks P)	106
Gambar 4.30	Hasil Test β_1 Poligon Tertutup Sudut Dalam	106
Gambar 4.31	Hasil Test β_1 Poligon Tertutup Sudut Luar	107
Gambar 4.32	Hasil Test β_2 (dengan matriks P)	107
Gambar 4.33	Hasil Test β_2 (tanpa matriks P)	108
Gambar 4.34	Hasil Test β_2 (dengan matriks P)	108
Gambar 4.35	Hasil Test β_2 (tanpa matriks P)	109

BAB I

PENDAHULUAN

I.1. Latar Belakang

Kemajuan teknologi yang semakin pesat berdampak positif terhadap perkembangan penentuan posisi titik/obyek di permukaan bumi termasuk bentuk jaringan titik kontrol horisontal, mulai dari pengukuran hingga proses data (hitung perataan).

Sebelum adanya komputer, untuk menentukan posisi horisontal melalui proses hitung perataan data ukuran dilakukan dengan perhitungan manual, sehingga proses hitung perataan tersebut membutuhkan waktu yang lama (tidak efisien). Setelah berkembang teknologi komputer baik hardware maupun softwarenya, manusia dapat memanfaatkan komputer untuk membantu menyelesaikan proses hitung perataan data ukuran sehingga posisi horisontal titik menjadi lebih mudah ditentukan.

Posisi titik-titik kontrol horisontal dinyatakan dengan komponen lintang geodetik (ϕ), bujur geodetik (λ) di atas ellipsoid acuan sedangkan pada bidang datar atau bidang proyeksi dinyatakan dengan X, Y atau E, N. Metode pengukuran Kerangka Kontrol Horisontal itu sendiri dapat dilakukan dengan dua cara yaitu metode konvensional dan modern. Tetapi dalam penulisan ini, metode yang digunakan adalah metode konvensional, khususnya poligon.

Dalam proses hitung perataan poligon baik pada poligon terbuka terikat sempurna maupun poligon tertutup “idealnya” menggunakan perataan kuadrat terkecil (*least square adjustment*), misalnya dengan metode parameter. Tetapi selama ini yang digunakan dalam proses hitung perataan poligon adalah metode boudith. Dengan metode perataan parameter, data jarak dan sudut ukuran secara otomatis akan diratakan bersama-sama sehingga hasil perhitungannya juga lebih teliti dibandingkan dengan metode boudith.

Dari uraian diatas, maka proses hitung perataan poligon dengan jumlah titik yang banyak tidak lagi menjadi masalah apabila telah ada paket program hitungan tersebut. Salah satu alternatif adalah dengan memanfaatkan bahasa pemrograman Borland C++ Builder untuk mempermudah dan mempercepat proses hitung perataan poligon (poligon terbuka terikat sempurna dan poligon tertutup) sehingga diharapkan akan diperoleh efisiensi dan hasil yang akurat.

I.2. Rumusan Masalah

Berdasarkan uraian di atas dapat dirumuskan suatu permasalahan dalam penelitian ini, yaitu bagaimana memvisualisasikan proses hitung perataan poligon dalam bahasa pemrograman Borland C++ Builder.

I.3. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk membuat program hitung perataan poligon dengan metode parameter menggunakan bahasa pemrograman Borland C++ Builder.

I.4 Batasan Masalah

Cakupan penelitian yang dilakukan adalah pembuatan program dalam mengautomatisasikan perhitungan poligon terbuka terikat sempurna dan poligon tertutup menggunakan perataan kuadrat terkecil (*least square adjustment*) metode parameter. Data-data yang digunakan telah direduksi ke bidang datar dan sistem koordinatnya adalah koordinat kartesian (X, Y).

I.5. Manfaat Penelitian

Dengan menggunakan bantuan bahasa pemrograman Borland C++ Builder, diharapkan hasil dari penelitian ini nantinya dapat memudahkan pengguna untuk melakukan proses hitung perataan parameter poligon dalam rangka penentuan nilai koordinat X, Y dari titik-titik poligon.

I.6. Tinjauan Pustaka

Poligon merupakan salah satu metode pengukuran konvensional pada Kerangka Kontrol Horizontal yang dapat dibedakan menjadi dua yaitu poligon terbuka dan poligon tertutup. Salah satu contoh poligon terbuka adalah poligon

terbuka terikat sempurna. (*Djawahir, 1993, Kerangka Kontrol Horisontal, PT. Petrakonsulindo Utama, Yogyakarta*)

Pada hitung perataan poligon (poligon terbuka terikat sempurna maupun poligon tertutup) dilakukan perataan terhadap jarak dan sudut observasi dengan menggunakan prinsip perataan kuadrat terkecil (*least square adjustment*) metode parameter. (*Wolf, P.R., 1980, Adjustment Computation, Second Edition, PBL. Publishing Co, Winconsin*)

Hasil hitung perataan poligon dengan menggunakan prinsip *least square adjustment* metode parameter dianggap mempunyai ketelitian yang baik walaupun proses perhitungannya lebih panjang dan rumit. (*Aprianto,A., 1995, Studi tentang Perataan Traverse Metode Parameter dan Metode Crandall dengan Variasi Ketelitian Jarak, Laporan Tugas Akhir Jurusan Teknik Geodesi Fakultas Teknik Sipil dan Perencanaan ITN Malang, Malang*)

Untuk mempermudah proses hitung perataan poligon tersebut perlu dibuat suatu program yang mampu menghitung perataan poligon (terbuka terikat sempurna maupun poligon tertutup) yang secara otomatis bisa didapatkan posisi horisontalnya. Borland C++ Builder merupakan pengembangan dari bahasa C dan keistimewaannya adalah bahasa ini mendukung pemrograman yang berorientasi objek serta lebih fleksibel dibandingkan dengan bahasa pemrograman yang lain. (*Pramono, D., 1998, Mudah Menguasai C++ Builder, Jilid II, PT. Elex Media Komputindo, Jakarta*)

I.7. Metode Penelitian

Metode penelitian yang dilakukan adalah sebagai berikut :

1.7.1 Studi Literatur

Studi literatur ini dilakukan untuk memperoleh pemahaman mengenai hitung perataan (*adjustment computation*) pada kerangka kontrol horisontal dan bahasa pemrograman Borland C++ Builder untuk mempermudah proses hitung perataannya.

1.7.2 Praktek Laboratorium

Studi laboratorium ini dilakukan untuk mempelajari dan mengaplikasikan teori-teori dari Borland C++ Builder, memproses data, evaluasi (pengujian) hasil proses dan pembahasan serta penyajian hasil proses.

BAB II

DASAR TEORI

II.1. Kerangka Kontrol Horisontal (KKH)

Kerangka Kontrol Horisontal (KKH) adalah bagian dari Kerangka Kontrol Geodesi (KKG) yang merupakan jaringan titik-titik yang diketahui koordinatnya dalam suatu sistem sebagai acuan untuk koordinasi semua kegiatan survey dan pemetaan di suatu daerah (*Djawahir, Ir., MSc., Kerangka Kontrol Horisontal, 1993*). Pengertian koordinasi semua kegiatan survey dan pemetaan (dalam hal ini) adalah pengkaitan atau penyambungan setiap hasil survey dan pemetaan dengan hasil survey dan pemetaan yang lain. Untuk pengkaitan dan penyambungan ini, tiap kegiatan survey dan pemetaan yang menyangkut posisi horisontal harus mengacu pada titik-titik kontrol horisontal dalam sistem KKH yang tunggal. Titik-titik kontrol horisontal dalam sistem jaringan tersebut ditentukan koordinatnya dengan metode dan ketelitian menurut spesifikasi tertentu. Selain sebagai acuan/titik ikat, Kerangka Kontrol Horisontal (KKH) juga mempunyai fungsi sebagai titik kontrol. Sebagai titik kontrol, minimal harus ada dua titik yang digunakan untuk mengontrol data-data hasil pengukuran agar diketahui kesalahannya.

Acuan yang digunakan dalam Kerangka Kontrol Horisontal (KKH) adalah ellipsoid yang merupakan bentuk geometrik pendekatan daripada bumi. Untuk posisi titik-titik kontrol horisontal dinyatakan dengan komponen lintang

geodetik (ϕ), bujur geodetik (λ) di atas ellipsoid acuan sedangkan pada bidang datar atau bidang proyeksi dinyatakan dengan X dan Y atau Easting, Northing (E dan N).

II.1.1. Klasifikasi Kerangka Kontrol Horisontal (KKH)

Sistem Kerangka Kontrol Horisontal (KKH) dalam suatu daerah/negara biasanya diklasifikasikan menurut jenjang ketelitiannya (dari ketelitian tinggi ke yang lebih rendah) yaitu orde 1, orde 2, dan orde 3. FGCC (*The United State Federal Geodetic Control Comitte*) membagi lagi KKH orde 2 dan orde 3 masing-masing dalam dua kelas yaitu kelas I dan II. Ketelitian KKH sering dinyatakan dengan ketelitian relatif yaitu angka perbandingan antara simpangan baku jarak dua titik kontrol yang dihubungkan dengan jarak itu sendiri. Apabila jarak antara dua titik kontrol horizontal hasil perataan adalah D dan simpangan bakunya adalah σ , maka ketelitian relatifnya adalah σ/D .

KKH orde 1 meliputi seluruh wilayah negara dengan distribusi titik kontrol yang relatif jarang. KKH orde 2 menyambung KKH orde 1 dengan distribusi titik kontrol yang lebih rapat daripada distribusi titik kontrol orde 1. KKH orde 3 menyambung KKH orde 2 dan orde 1 dengan distribusi titik kontrol lebih rapat daripada titik-titik kontrol orde 2. Klasifikasi KKH berdasarkan ketelitian relatif yang diterapkan di United State serta anjuran penggunaannya adalah sebagai berikut :

1. Orde 1

Ketelitian relatif 1 : 100.000

Penggunaannya untuk :

- Jaringan kontrol nasional utama;
- Studi geodinamik;
- Uji coba peralatan pertahanan udara;
- Studi penampilan kendaraan udara;
- Survey kontrol daerah metropolitan;
- Survey kontrol pada proyek rekayasa ketelitian tinggi yang terbentang jauh memanjang.

2. Orde 2 Kelas I

Ketelitian relatif 1 : 50.000

Penggunaannya :

- Sebagai bagian dari jaringan kontrol nasional dan merupakan kerangka dasar untuk perapatan jaringan;
- Untuk jaringan kontrol di daerah lahan berharga tinggi.

Orde 2 Kelas II

Ketelitian relatif 1 : 20.000

Penggunaannya :

- Sebagai bagian dari jaringan kontrol nasional;
- Untuk survey kontrol untuk lalu lintas air di darat (kanal, terusan, danau), jalan raya antar kota, konstruksi, dan pembagian lahan.

3. Orde 3 Kelas I dan II

Ketelitian relatif orde 3 untuk kelas I adalah 1 : 10.000 sedangkan untuk kelas II adalah 1 : 5.000.

Penggunaannya :

- Tidak selalu bagian dari kerangka kontrol horisontal;
- Untuk survey kontrol pemetaan topografi dan hidrografi.

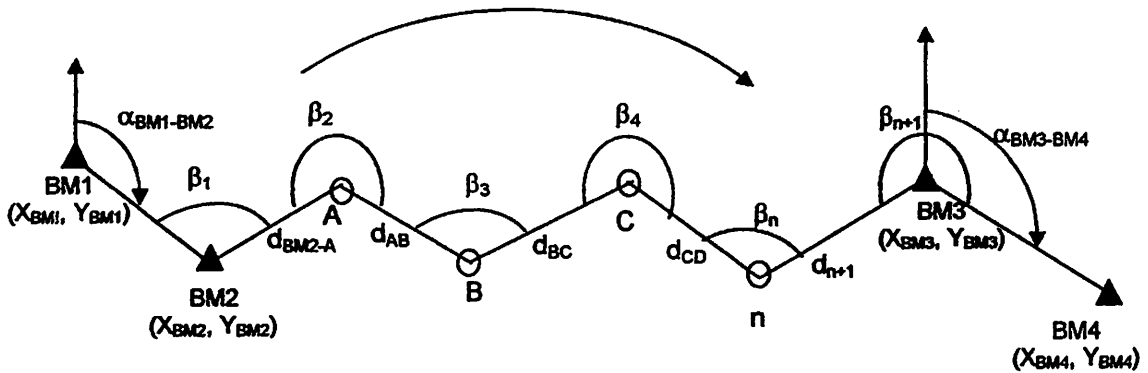
II.1.2. Metode Pengukuran Poligon

Poligon merupakan satu alternatif disamping triangulasi dan trilaterasi yang umum diterapkan untuk pembentukan KKH. Dengan metode poligon ini, titik-titik kontrol yang akan ditentukan koordinat geodetiknya dirangkai dalam jaringan segi banyak. Besaran-besaran yang diukur adalah semua jarak dan sudut horisontal (terbentang antara dua sisi pada tiap titik poligon). Dengan koordinat dan azimuth geodetik pada titik awal (*datum*), maka koordinat titik-titik poligon dapat ditentukan. Berdasarkan bentuk geometrisnya, poligon terbagi dua yaitu poligon terbuka dan poligon tertutup. Dalam penulisan tugas akhir ini, metode poligon yang digunakan adalah poligon terbuka terikat sempurna dan poligon tertutup.

1. Poligon Terbuka Terikat Sempurna

Poligon terbuka terikat sempurna adalah poligon terbuka yang mempunyai empat titik ikat/tetap yaitu dua buah titik ikat di sisi awal poligon dan dua buah titik ikat di sisi akhir poligon, sehingga akan didapatkan dua buah azimuth yaitu azimuth di sisi awal dan azimuth di sisi akhir poligon yang diperoleh melalui perhitungan.

Poligon ini cocok digunakan dalam pengukuran yang memerlukan ketelitian tinggi. (arah hitungan searah dengan jarum jam)



Gambar 2.1. Poligon Terbuka Terikat Sempurna

Keterangan gambar 2.1 adalah :

BM_1, BM_2, BM_3, BM_4 : Titik ikat/tetap

A, B, C, n : Titik-titik yang akan dihitung koordinatnya

$d_{BM2-A}, d_{AB}, d_{BC}, d_{CD}, d_{n+1}$: Jarak horisontal

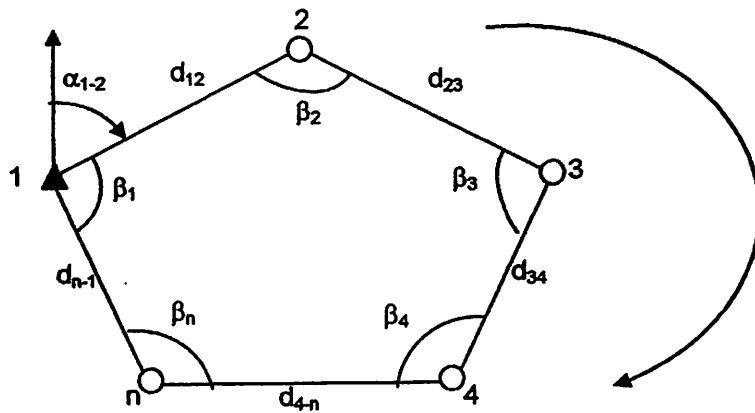
$\beta_1, \beta_2, \beta_3, \beta_4, \beta_n, \beta_{n+1}$: Sudut-sudut horisontal

$\alpha_{BM1-BM2}, \alpha_{BM3-BM4}$: Azimuth awal dan akhir poligon

2. Poligon Tertutup

Poligon tertutup adalah suatu poligon yang titik awal dan titik akhir pada titik yang sama. Selain itu nilai azimuth awal harus sama dengan azimuth akhirnya. Pada poligon tertutup untuk menghitung azimuth suatu titik dapat menggunakan sudut dalam maupun sudut luar. Apabila menggunakan sudut dalam, maka azimuth suatu titik didapatkan dari azimuth awal (α_{awal}) + 180° - sudut dalam (β), sedangkan apabila menggunakan sudut luar maka azimuth suatu titik didapatkan dari azimuth awal (α_{awal}) + sudut luar (β) - 180° . (arah hitungan searah dengan jarum jam)

a. Poligon Tertutup dengan Sudut Dalam

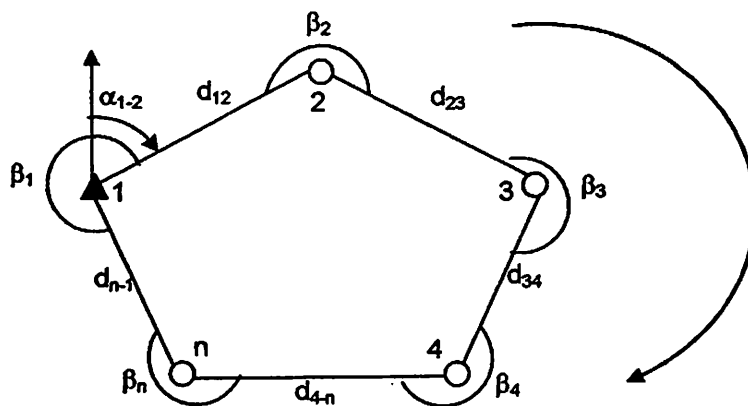


Gambar 2.2. Poligon tertutup dengan sudut dalam

Keterangan gambar 2.2 adalah :

- Titik 1 : Titik ikat/tetap
- Titik 2, 3, 4, n : Titik-titik poligon
- $d_{12}, d_{23}, d_{34}, d_{45}, d_{n-1}$: Jarak horisontal
- $\beta_1, \beta_2, \beta_3, \beta_4, \beta_n$: Sudut horisontal (sudut dalam)

b. Poligon Tertutup dengan Sudut Luar



Gambar 2.3. Poligon tertutup dengan sudut luar

Keterangan gambar 2.3 adalah :

Titik 1	: Titik ikat/tetap
Titik 2, 3, 4, n	: Titik-titik poligon
$d_{12}, d_{23}, d_{34}, d_{45}, d_{n-1}$	: Jarak horisontal
$\beta_1, \beta_2, \beta_3, \beta_4, \beta_n$	: Sudut horisontal (sudut luar)

II.2. Hitung Perataan Kerangka Kontrol Horisontal

Perataan merupakan cara memberikan koreksi pada hasil pengukuran sehingga diperoleh nilai pengamatan yang memenuhi persyaratan atau aturan. Hitung perataan jaringan kontrol horisontal ini dimaksudkan untuk mendapatkan koordinat geodetik (X,Y) dari titik-titik kontrol horisontal beserta ketelitiannya dari data ukuran lapangan. Hitungan perataan Kerangka Kontrol Horisontal (KKH) ini umumnya dilaksanakan dalam dua tahap yaitu : (*Djawahir,Ir., MSc., Kerangka Kontrol Horisontal, 1993*)

1. Tahap hitungan koordinat pendekatan;
2. Tahap perataan koordinat.

Hitungan koordinat titik-titik kontrol horisontal dapat dilakukan secara parsial, titik demi titik secara independen atau dalam blok-blok jaringan yang mencakup sejumlah titik kontrol. Teknik yang umum dilakukan terutama pada masa lalu sebelum era komputer berkapasitas besar adalah hitungan koordinat pendekatan titik-titik kontrol dalam blok-blok jaringan dengan jumlah titik terbatas, misalnya antara dua kontrol azimuth atau dua kontrol basis.

Prosedur yang umum dilaksanakan adalah pertama, reduksi data ukuran (jarak, sudut, dan azimuth) ke permukaan ellipsoid acuan atau bidang peta. Kedua, data ukuran tereduksi tersebut dikoreksi/diratakan agar memenuhi kondisi geometrik jaringan dalam blok yang bersangkutan. Ketiga, dengan data ukuran tereduksi yang telah dikoreksi tersebut dihitung koordinat pendekatan titik-titik kontrol dalam jaringan.

Tugas akhir ini hanya akan membahas masalah hitung perataan poligon. Karena poligon merupakan salah satu metode pengukuran di dalam Kerangka Kontrol Horisontal (KKH), maka prosedur hitung perataannya sama dengan hitung perataan KKH. Pada tugas akhir ini data ukuran dianggap benar (tanpa dilakukan reduksi) sehingga data ukuran tersebut langsung dihitung koordinat pendekatan titik-titik kontrolnya dalam jaringan.

Ketelitian data koordinat pendekatan hasil hitungan diatas ditentukan antara lain oleh rumus yang dipakai dan letak titik relatif terhadap titik awal (*datum*), makin jauh dari titik awal hitungan atau makin banyak rangkaian titik yang memisahkannya dari titik awal hitungan maka ketelitiannya makin menurun. Dengan kata lain, koordinat pendekatan yang dihitung dengan teknik diatas menghasilkan data koordinat dengan ketelitian yang tidak merata. Tugas pemerataan ketelitian selanjutnya dilakukan dengan proses perataan kuadrat terkecil (*least square adjustment*).

II.2.1. Hitung Perataan Kuadrat Terkecil (*Least Square Adjustment*)

Pada metode hitung perataan kuadrat terkecil (*least square adjustment*) besarnya koreksi yang diberikan untuk setiap hasil ukuran harus memenuhi

persyaratan bahwa jumlah kuadrat dari koreksi yang diberikan pada setiap hasil pengukuran adalah minimum.

Bentuk matriksnya :

$$\mathbf{V}^T \mathbf{P} \mathbf{V} = \text{minimum (dengan bobot/weight)}$$

atau

$$\mathbf{V}^T \mathbf{V} = \text{minimum (tanpa bobot/unweight)}$$

Persoalan umum dari hitung perataan yang menyatakan hubungan antara hasil ukuran dengan besaran yang akan ditentukan dapat dinyatakan dengan model matematik yang berbentuk :

$$\mathbf{F} = (\mathbf{L} + \mathbf{V}, \mathbf{X}) = 0 \quad \dots\dots\dots (2.1)$$

dimana : $\mathbf{V}$ = vektor koreksi

$\mathbf{L}$ = vektor ukuran

$\mathbf{X}$ = vektor parameter

Dengan menggunakan prinsip $\mathbf{V}^T \mathbf{P} \mathbf{V} = \text{minimum (weight)}$ maupun dengan $\mathbf{V}^T \mathbf{V} = \text{minimum (unweight)}$, harga $\mathbf{X}$ yang paling baik dan sesuai dengan hasil ukuran dapat ditentukan. Metode hitung perataan kuadrat terkecil (*least square adjustment*) ditinjau dari segi operasi matematisnya atau dari persamaan-persamaan yang memperlihatkan hubungan antara besaran yang akan ditentukan dengan hasil ukuran dibedakan menjadi :

- ❖ Hitung Perataan Bersyarat;
- ❖ Hitung Perataan Parameter;
- ❖ Hitung Perataan Kombinasi

Pada perataan poligon ini, metode hitung perataan kuadrat terkecil (*least square adjustment*) yang dianjurkan adalah Hitung Perataan Parameter.

II.2.2. Hitung Perataan Parameter

Dalam hitung perataan parameter, model matematik yang digunakan adalah fungsi matematik yang menghubungkan data ukuran dengan parameter yang akan ditentukan yaitu koordinat titik-titik kontrol. Prinsip dari perataan parameter adalah parameter yang dicari / dihitung nilainya merupakan koreksi yang langsung diberikan kepada nilai pendekatan. Nilai pendekatan (parameter pendekatan) dapat ditentukan bebas, tetapi pada umumnya dihitung dengan menggunakan data pengamatan. Model matematik (persamaan parameter) dapat ditentukan/dibentuk berdasarkan nilai pengamatan yang dinyatakan sebagai fungsi dari nilai parameter.

Dalam bentuk matriks, untuk fungsi linier ditulis :

$$\hat{L} = \hat{A} \cdot \hat{X} \dots\dots\dots (2.2)$$

dimana : $\hat{L}$ = matriks pengamatan

$\hat{A}$ = matriks koefisien

$\hat{X}$ = matriks parameter

$$\hat{L} = L + V$$

$$\hat{X} = X^o + \Delta X$$

dari persamaan diatas, maka :

$$L + V = A (X^o + \Delta X)$$

$$= A X^o + A. \Delta X$$

$$V = A. \Delta X + (A X^o - L)$$

$$\text{atau } V = A. \Delta X - (L - A X^o)$$

$$\text{Jika : } A X^o - L = F \quad \text{atau} \quad L - A X^o = F,$$

$F =$ matriks konstanta

Maka model matematik persamaan parameter dapat ditulis :

$$V = A X + F$$

$$X = -(A^T P A)^{-1} (A^T P F)$$

atau

$$V = A X - F$$

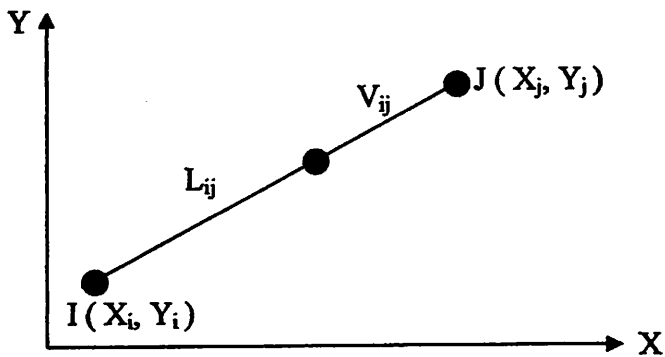
$$X = (A^T P A)^{-1} (A^T P F) \dots\dots\dots (2.3)$$

Jika bobot pengukuran (P) diasumsikan sama, maka matriks P merupakan unit matriks (matriks satuan) sehingga tidak terpengaruh terhadap proses hitung perkalian matriks. Berdasarkan hal tersebut, maka dari model $V = A X - F$ dapat ditulis menjadi :

$$X = (A^T A)^{-1} (A^T F) \dots\dots\dots (2.4)$$

Di dalam metode parameter, untuk mencari besarnya nilai dari matriks koefisien (matriks A) dilakukan dengan proses linierisasi terhadap persamaan pengukuran jarak dan sudut. (Wolf, P.R., 1980, *Adjustment Computation, Second Edition*)

1. Linierisasi Persamaan Pengukuran Jarak



Gambar 2.4. Linierisasi Jarak

Pada gambar 2.4, diukur jarak dari titik I ke J (L_{ij}). Karena L_{ij} merupakan hasil ukuran, maka memiliki kesalahan dalam pengukuran ($V_{L_{ij}}$) sehingga nilai hasil ukuran yang definitif adalah $\overline{L_{ij}} = L_{ij} + V_{L_{ij}}$ (2.5)

Karena garis L_{ij} berada pada sistem koordinat planimetris, maka berlaku rumus jarak dalam koordinat. Pada koordinat titik I dan J terdapat X_i , Y_i , X_j , dan Y_j sehingga :

$$\overline{L_{ij}} = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2} \dots\dots\dots (2.6)$$

atau

$$L_{ij} + V_{L_{ij}} = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2} \dots\dots\dots (2.7)$$

Persamaan (2g) adalah fungsi non linier dari variable unknown X_i , Y_i , X_j , dan Y_j dapat ditulis seperti :

$$L_{ij} + V_{Lij} = F(X_i, Y_i, X_j, Y_j) \dots\dots\dots (2.8)$$

$$F(X_i, Y_i, X_j, Y_j) = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2}$$

Fungsi non linier tersebut harus dilinierkan dengan Deret Taylor dan penyelesaian data ukuran dengan variasi koordinat. Deret Taylor dari fungsi jarak dengan variasi koordinat adalah :

$$F(X_i, Y_i, X_j, Y_j) = F(X_i^o, Y_i^o, X_j^o, Y_j^o) + \left[\frac{\partial F}{\partial X_i} \right]^o dX_i + \left[\frac{\partial F}{\partial Y_i} \right]^o dY_i + \left[\frac{\partial F}{\partial X_j} \right]^o dX_j + \left[\frac{\partial F}{\partial Y_j} \right]^o dY_j \dots\dots\dots (2.9)$$

$\left[\frac{\partial F}{\partial X_i} \right]^o$, $\left[\frac{\partial F}{\partial Y_i} \right]^o$, $\left[\frac{\partial F}{\partial X_j} \right]^o$, $\left[\frac{\partial F}{\partial Y_j} \right]^o$ adalah turunan parsial (derivatif parsial) dari fungsi jarak sehingga :

$$X_i = X_i^o + dX_i \qquad X_j = X_j^o + dX_j$$

$$Y_i = Y_i^o + dY_i \qquad Y_j = Y_j^o + dY_j$$

Turunan parsial dari fungsi U, dengan substitusi persamaan (2.8) dan (2.9) selanjutnya akan menghasilkan linierisasi persamaan pengukuran jarak linierisasi seperti berikut ini :

$$K_{Lij} + V_{Lij} = \left[\frac{X_i^\circ - X_j^\circ}{IJ^\circ} \right] dX_i + \left[\frac{Y_i^\circ - Y_j^\circ}{IJ^\circ} \right] dY_i + \left[\frac{X_j^\circ - X_i^\circ}{IJ^\circ} \right] dX_j + \left[\frac{Y_j^\circ - Y_i^\circ}{IJ^\circ} \right] dY_j \dots\dots\dots (2.10)$$

Dimana : $K_{Lij} = L_{ij} - L_{ij}^\circ$

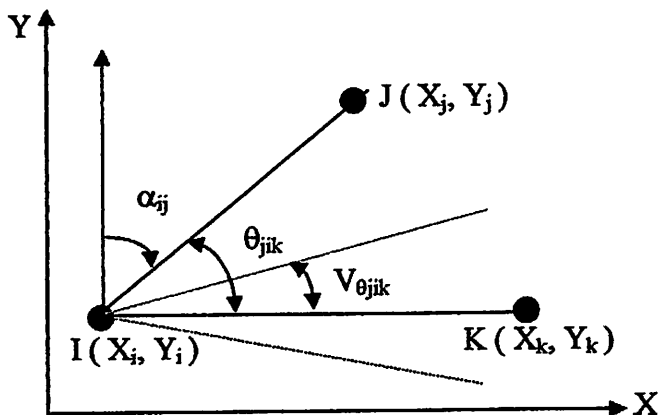
$$IJ^\circ = \sqrt{(X_j^\circ - X_i^\circ)^2 + (Y_j^\circ - Y_i^\circ)^2}$$

Sehingga dari persamaan (2.10) akan didapatkan rumusan persamaan berikut :

$$\frac{\partial F}{\partial X_i} = \left[\frac{X_i^\circ - X_j^\circ}{(L_{ij}^\circ)} \right] , \quad \frac{\partial F}{\partial Y_i} = \left[\frac{Y_i^\circ - Y_j^\circ}{(L_{ij}^\circ)} \right]$$

$$\frac{\partial F}{\partial X_j} = \left[\frac{X_j^\circ - X_i^\circ}{(L_{ij}^\circ)} \right] , \quad \frac{\partial F}{\partial Y_j} = \left[\frac{Y_j^\circ - Y_i^\circ}{(L_{ij}^\circ)} \right] \dots\dots (2.11)$$

2. Linierisasi Persamaan Pengukuran Sudut



Gambar 2.5. Linierisasi Sudut

Pada gambar 2.5 diatas, persamaan observasi sudut adalah sebagai berikut :

$$\begin{aligned}\overline{\theta_{ijk}} &= \theta_{ijk} + V_{\theta_{ijk}} \\ &= \left[\arctan \frac{X_k - X_i}{Y_k - Y_i} - \arctan \frac{X_j - X_i}{Y_j - Y_i} \right] \dots\dots\dots (2.12)\end{aligned}$$

$$\theta_{ijk} + V_{\theta_{ijk}} = \left[\arctan \frac{X_k - X_i}{Y_k - Y_i} - \arctan \frac{X_j - X_i}{Y_j - Y_i} \right] \dots\dots\dots (2.13)$$

Kemudian dari persamaan (2.13), fungsi $\theta_{ijk} + V_{\theta_{ijk}}$ dengan variabel $X_i, Y_i, X_j, Y_j, X_k, Y_k$ menjadi :

$$\theta_{ijk} + V_{\theta_{ijk}} = U(X_i, Y_i, X_j, Y_j, X_k, Y_k) \dots\dots\dots (2.14)$$

$$U(X_i, Y_i, X_j, Y_j, X_k, Y_k) = \left[\arctan \frac{X_k - X_i}{Y_k - Y_i} - \arctan \frac{X_j - X_i}{Y_j - Y_i} \right]$$

Karena persamaan (2.14) merupakan persamaan non linier maka harus dilinierkan dengan menggunakan *Deret Taylor* menjadi :

$$\begin{aligned}U(X_i, Y_i, X_j, Y_j, X_k, Y_k) &= U^o(X_i^o, Y_i^o, X_j^o, Y_j^o, X_k^o, Y_k^o) + \\ &\left[\frac{du}{dX_i} \right]^o dX_i + \left[\frac{du}{dY_i} \right]^o dY_i + \left[\frac{du}{dX_j} \right]^o dX_j + \left[\frac{du}{dY_j} \right]^o dY_j + \left[\frac{du}{dX_k} \right]^o dX_k + \\ &\left[\frac{du}{dY_k} \right]^o dY_k \dots\dots\dots (2.15)\end{aligned}$$

Turunan parsial dari fungsi U , dengan substitusi persamaan (2.14) dan (2.13) akan menghasilkan linierisasi persamaan pengukuran sudut seperti berikut :

$$\frac{dp_j}{du} = \left[\frac{p_j^{\#}(\circ)_z}{X_j^{\circ} - Y_j^{\circ}} \right] dp_j$$

$$\frac{dX_j}{du} = \left[\frac{p_j^{\#}(\circ)_z}{Y_j^{\circ} - X_j^{\circ}} \right] dX_j$$

$$\frac{dp_i}{du} = \left[\frac{p_i^{\#}(\circ)_z}{X_i^{\circ} - X_j^{\circ}} - \frac{p_j^{\#}(\circ)_z}{X_j^{\circ} - Y_j^{\circ}} \right] dp_i$$

$$\frac{dX_i}{du} = \left[\frac{p_i^{\#}(\circ)_z}{Y_i^{\circ} - Y_j^{\circ}} - \frac{p_j^{\#}(\circ)_z}{Y_j^{\circ} - X_j^{\circ}} \right] dX_i$$

(2.17) seperti berikut ini :

Sehingga dari persamaan (2.16) akan didapatkan rumusan persamaan

$$d_{jk}^{\circ} = \sqrt{(X_j^{\circ} - X_k^{\circ})^2 + (Y_j^{\circ} - Y_k^{\circ})^2}$$

$$d_{ij}^{\circ} = \sqrt{(X_j^{\circ} - X_i^{\circ})^2 + (Y_j^{\circ} - Y_i^{\circ})^2}$$

$$\theta_{jk}^{\circ} = \left[\arctan \frac{X_k^{\circ} - X_j^{\circ}}{Y_k^{\circ} - Y_j^{\circ}} - \arctan \frac{X_i^{\circ} - X_j^{\circ}}{Y_i^{\circ} - Y_j^{\circ}} \right]$$

dimana : $K_{ijk} = \theta_{ijk} - \theta_{jik}^{\circ}$

$$\left[\frac{X_i^{\circ} - X_k^{\circ}}{Y_i^{\circ} - Y_k^{\circ}} \right] dp_k \dots \dots \dots (2.16)$$

$$K_{ijk} + V_{ijk} = \left[\frac{Y_j^{\circ} - Y_i^{\circ}}{Y_j^{\circ} - Y_k^{\circ}} - \frac{(IJ)_z^{\circ}}{(IK)_z^{\circ}} \right] dX_i + \left[\frac{X_i^{\circ} - X_j^{\circ}}{X_i^{\circ} - X_k^{\circ}} - \frac{(IJ)_z^{\circ}}{(IK)_z^{\circ}} \right] dp_j + \left[\frac{Y_k^{\circ} - Y_i^{\circ}}{Y_k^{\circ} - Y_j^{\circ}} \right] dX_k +$$

$$\frac{du}{dX_k} = \left[\frac{Y_k^{\circ} - Y_l^{\circ}}{(d_{lk}^{\circ})^2} \right] dX_k$$

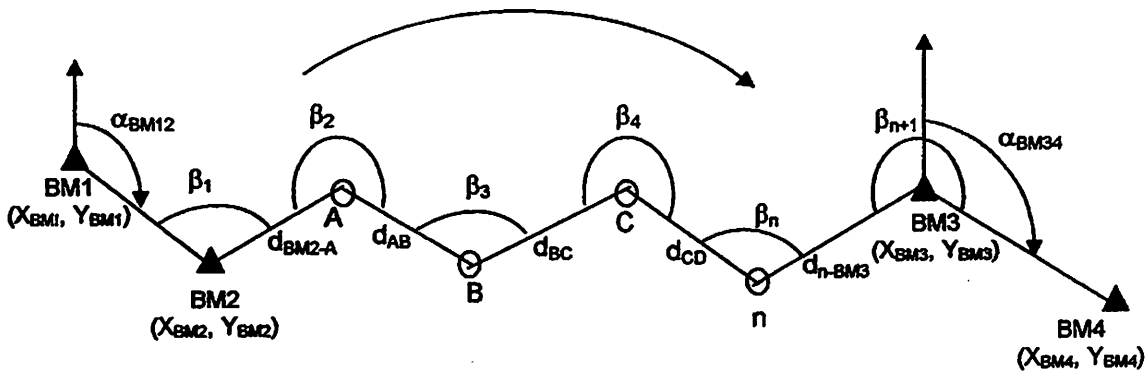
$$\frac{du}{dY_k} = \left[\frac{X_l^{\circ} - X_k^{\circ}}{(d_{lk}^{\circ})^2} \right] dY_k \dots\dots\dots (2.17)$$

Persamaan (2.17) harus diubah menjadi sistem radian dengan cara mengalikan dengan nilai rho (ρ) = 206265". Proses linierisasi ini bertujuan untuk mendapatkan nilai matriks koefisien (matriks A).

II.2.3. Tahapan Hitung Perataan Parameter untuk Poligon

II.2.3.1. Poligon Terbuka Terikat Sempurna

Tahapan hitung perataan parameter untuk poligon terbuka terikat sempurna adalah sebagai berikut :



Gambar 2.6. Poligon Terbuka Terikat Sempurna

1. Menghitung azimuth pendekatan yang dimulai dari azimuth dua koordinat awal (azimuth awal / $\alpha_{BM1-BM2}$) sampai azimuth dua koordinat akhir poligon ($\alpha_{BM3-BM4}$) dengan rumus sebagai berikut :

$$\alpha_{BM1-BM2} = \arctg \left[\frac{X_{BM2} - X_{BM1}}{Y_{BM2} - Y_{BM1}} \right]$$

$$\alpha_{BM2-A}^{\circ} = \alpha_{BM1-BM2} + \beta_1 - 180^{\circ} \quad \alpha_{AB}^{\circ} = \alpha_{BM2-A}^{\circ} + \beta_2 - 180^{\circ}$$

$$\alpha_{BC}^{\circ} = \alpha_{AB}^{\circ} + \beta_3 - 180^{\circ} \quad \alpha_{C-n}^{\circ} = \alpha_{BC}^{\circ} + \beta_4 - 180^{\circ}$$

$$\alpha_{n-BM3}^{\circ} = \alpha_{C-n}^{\circ} + \beta_n - 180^{\circ}$$

$$\alpha_{BM3-BM4} = \arctg \left[\frac{X_{BM4} - X_{BM3}}{Y_{BM4} - Y_{BM3}} \right] \dots\dots\dots (2.17)$$

2. Menghitung koordinat pendekatan titik poligon dimana koordinat titik BM2 sebagai koordinat awal (X_a, Y_a), dengan rumus sebagai berikut :

$$X_{a+1}^{\circ} = X_a + d_{a-(a+1)} \sin \alpha_{a-(a+1)}$$

$$Y_{a+1}^{\circ} = Y_n + d_{a-(a+1)} \cos \alpha_{a-(a+1)}$$

Sehingga :

$$X_A^{\circ} = X_2 + d_{BM2-A} \sin \alpha_{BM2-A}^{\circ}$$

$$Y_A^{\circ} = Y_2 + d_{BM2-A} \cos \alpha_{BM2-A}^{\circ}$$

$$X_B^{\circ} = X_A^{\circ} + d_{AB} \sin \alpha_{AB}^{\circ}$$

$$Y_B^{\circ} = Y_A^{\circ} + d_{AB} \cos \alpha_{AB}^{\circ}$$

$$X_C^{\circ} = X_B^{\circ} + d_{BC} \sin \alpha_{BC}^{\circ}$$

$$Y_C^{\circ} = Y_B^{\circ} + d_{BC} \cos \alpha_{BC}^{\circ}$$

$$X_n^{\circ} = X_C^{\circ} + d_{C-n} \sin \alpha_{C-n}^{\circ}$$

$$Y_n^{\circ} = Y_C^{\circ} + d_{C-n} \cos \alpha_{C-n}^{\circ} \quad \dots\dots\dots (2.18)$$

3. Menghitung jarak pendekatan (d°) dan sudut horisontal pendekatan (β°) dengan rumus sebagai berikut :

➤ Jarak pendekatan (d°)

$$d_{a-(a+1)}^{\circ} = \sqrt{(X_{a+1}^{\circ} - X_a)^2 + (Y_{a+1}^{\circ} - Y_a)^2}$$

sehingga :

$$d_{BM2-A}^{\circ} = \sqrt{(X_A^{\circ} - X_{BM2})^2 + (Y_A^{\circ} - Y_{BM2})^2}$$

$$d_{AB}^{\circ} = \sqrt{(X_B^{\circ} - X_A^{\circ})^2 + (Y_B^{\circ} - Y_A^{\circ})^2}$$

$$\begin{aligned}
d_{BC}^{\circ} &= \sqrt{(X_C^{\circ} - X_B^{\circ})^2 + (Y_C^{\circ} - Y_B^{\circ})^2} \\
d_{C-n}^{\circ} &= \sqrt{(X_n^{\circ} - X_C^{\circ})^2 + (Y_n^{\circ} - Y_C^{\circ})^2} \\
d_{n-BM3}^{\circ} &= \sqrt{(X_{BM3} - X_n^{\circ})^2 + (Y_{BM3} - Y_n^{\circ})^2} \dots\dots\dots (2.19)
\end{aligned}$$

➤ Sudut pendekatan (β°)

$$\begin{aligned}
\beta_1^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_A^{\circ} - X_{BM2}}{Y_A^{\circ} - Y_{BM2}} \right] - \arctg \left[\frac{X_{BM2} - X_{BM1}}{Y_{BM2} - Y_{BM1}} \right] \\
\beta_2^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_B^{\circ} - X_A^{\circ}}{Y_B^{\circ} - Y_A^{\circ}} \right] - \arctg \left[\frac{X_A^{\circ} - X_{BM2}}{Y_A^{\circ} - Y_{BM2}} \right] \\
\beta_3^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_C^{\circ} - X_B^{\circ}}{Y_C^{\circ} - Y_B^{\circ}} \right] - \arctg \left[\frac{X_B^{\circ} - X_A^{\circ}}{Y_B^{\circ} - Y_A^{\circ}} \right] \\
\beta_4^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_D^{\circ} - X_C^{\circ}}{Y_D^{\circ} - Y_C^{\circ}} \right] - \arctg \left[\frac{X_C^{\circ} - X_B^{\circ}}{Y_C^{\circ} - Y_B^{\circ}} \right] \\
\beta_n^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_{BM3} - X_n^{\circ}}{Y_{BM3} - Y_n^{\circ}} \right] - \arctg \left[\frac{X_n^{\circ} - X_C^{\circ}}{Y_n^{\circ} - Y_C^{\circ}} \right] \\
\beta_{n+1}^{\circ} &= 180^{\circ} + \arctg \left[\frac{X_{BM4} - X_{BM3}}{Y_{BM4} - Y_{BM3}} \right] - \arctg \left[\frac{X_{BM3} - X_n^{\circ}}{Y_{BM3} - Y_n^{\circ}} \right] \dots\dots\dots (2.20)
\end{aligned}$$

4. Menghitung elemen matriks A yang didapatkan dari hasil linierisasi jarak dan sudut. Untuk linierisasi sudut dikalikan dengan nilai rho (ρ) = 206265"/rad.

➤ Linierisasi jarak

$$d_{BM2-A} = d_{BM2-A}^{\circ} + \left[\frac{\delta d}{\delta X_{BM2}} \right] \Delta X_{BM2} + \left[\frac{\delta d}{\delta Y_{BM2}} \right] \Delta Y_{BM2} + \left[\frac{\delta d}{\delta X_A} \right] \Delta X_A +$$

$$\left[\frac{\delta d}{\delta Y_A} \right] \Delta Y_A$$

$$\frac{\delta d}{\delta X_{BM2}} = - \left[\frac{X_A^{\circ} - X_{BM2}}{d_{BM2-A}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_{BM2}} = - \left[\frac{Y_A^{\circ} - Y_{BM2}}{d_{BM2-A}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_A} = + \left[\frac{X_A^{\circ} - X_{BM2}}{d_{BM2-A}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_A} = + \left[\frac{Y_A^{\circ} - Y_{BM2}}{d_{BM2-A}^{\circ}} \right]$$

$$d_{AB} = d_{AB}^{\circ} + \left[\frac{\delta d}{\delta X_A} \right] \Delta X_A + \left[\frac{\delta d}{\delta Y_A} \right] \Delta Y_A + \left[\frac{\delta d}{\delta X_B} \right] \Delta X_B + \left[\frac{\delta d}{\delta Y_B} \right] \Delta Y_B$$

$$\frac{\delta d}{\delta X_A} = - \left[\frac{X_B^{\circ} - X_A^{\circ}}{d_{AB}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_A} = - \left[\frac{Y_B^{\circ} - Y_A^{\circ}}{d_{AB}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_B} = + \left[\frac{X_B^{\circ} - X_A^{\circ}}{d_{AB}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_B} = + \left[\frac{Y_B^{\circ} - Y_A^{\circ}}{d_{AB}^{\circ}} \right]$$

$$d_{BC} = d_{BC}^{\circ} + \left[\frac{\delta d}{\delta X_B} \right] \Delta X_B + \left[\frac{\delta d}{\delta Y_B} \right] \Delta Y_B + \left[\frac{\delta d}{\delta X_C} \right] \Delta X_C + \left[\frac{\delta d}{\delta Y_C} \right] \Delta Y_C$$

$$\frac{\delta d}{\delta X_B} = - \left[\frac{X_C^{\circ} - X_B^{\circ}}{d_{BC}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_B} = - \left[\frac{Y_C^{\circ} - Y_B^{\circ}}{d_{BC}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_C} = + \left[\frac{X_C^{\circ} - X_B^{\circ}}{d_{BC}^{\circ}} \right]$$

$$\frac{\delta d}{\delta Y_C} = + \left[\frac{Y_C^{\circ} - Y_B^{\circ}}{d_{BC}^{\circ}} \right]$$

$$d_{C-n} = d_{C-n}^{\circ} + \left[\frac{\delta d}{\delta X_C} \right] \Delta X_C + \left[\frac{\delta d}{\delta Y_C} \right] \Delta Y_C + \left[\frac{\delta d}{\delta X_n} \right] \Delta X_n + \left[\frac{\delta d}{\delta Y_n} \right] \Delta Y_n$$

$$\frac{\delta d}{\delta X_C} = - \left[\frac{X_n^{\circ} - X_C^{\circ}}{d_{C-n}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_C} = - \left[\frac{Y_n^{\circ} - Y_C^{\circ}}{d_{C-n}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_n} = + \left[\frac{X_n^{\circ} - X_C^{\circ}}{d_{C-n}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_n} = + \left[\frac{Y_n^{\circ} - Y_C^{\circ}}{d_{C-n}^{\circ}} \right]$$

$$d_{n-BM3} = d_{n-BM3}^{\circ} + \left[\frac{\delta d}{\delta X_n} \right] \Delta X_n + \left[\frac{\delta d}{\delta Y_n} \right] \Delta Y_n + \left[\frac{\delta d}{\delta X_{BM3}} \right] \Delta X_{BM3} +$$

$$\left[\frac{\delta d}{\delta Y_{BM3}} \right] \Delta Y_{BM3}$$

$$\frac{\delta d}{\delta X_n} = - \left[\frac{X_{BM3} - X_n^{\circ}}{d_{n-BM3}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_n} = - \left[\frac{Y_{BM3} - Y_n^{\circ}}{d_{n-BM3}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_{BM3}} = + \left[\frac{X_{BM3} - X_n^{\circ}}{d_{n-BM3}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_{BM3}} = + \left[\frac{Y_{BM3} - Y_n^{\circ}}{d_{n-BM3}^{\circ}} \right]$$

..... (2.21)

➤ Linierisasi Sudut

$$\beta_1 + V_1 = \left[\frac{Y_{BM2} - Y_A^{\circ}}{(d_{BM2-A}^{\circ})^2} - \frac{Y_{BM2} - Y_{BM1}}{(d_{BM1-BM2}^{\circ})^2} \right] dX_{BM2} +$$

$$\left[\frac{X_A^{\circ} - X_{BM2}}{(d_{BM2-A}^{\circ})^2} - \frac{X_{BM1} - X_{BM2}}{(d_{BM1-BM2}^{\circ})^2} \right] dY_{BM2} + \left[\frac{Y_{BM2} - Y_{BM1}}{(d_{BM1-BM2}^{\circ})^2} \right] dX_{BM1} +$$

$$\left[\frac{X_{BM1} - X_{BM2}}{(d_{BM1-BM2}^{\circ})^2} \right] dY_{BM1} + \left[\frac{Y_A^{\circ} - Y_{BM2}}{(d_{BM2-A}^{\circ})^2} \right] dX_A + \left[\frac{X_{BM2} - X_A^{\circ}}{(d_{BM2-A}^{\circ})^2} \right] dY_A$$

$$\begin{aligned}
 & \left[\frac{p^{(c-n)}(z)}{X^c - X^n} \right]_{Y^n} \\
 & + \left[\frac{p^{(c-n)}(z)}{Y^n - Y^c} \right]_{X^n} + \left[\frac{p^{(bc)}(z)}{X^b - X^c} \right]_{Y^b} + \left[\frac{p^{(bc)}(z)}{Y^c - Y^b} \right]_{X^b} \\
 & \left[\frac{p^{(bc)}(z)}{Y^c - Y^n} - \frac{p^{(c-n)}(z)}{X^c - X^b} \right]_{Y^c} + \left[\frac{p^{(bc)}(z)}{Y^c - Y^b} - \frac{p^{(c-n)}(z)}{Y^n - Y^c} \right]_{X^c} = \beta_4 + \nu_4 \\
 & \left[\frac{p^{(bc)}(z)}{X^b - X^c} \right]_{Y^c} \\
 & + \left[\frac{p^{(bc)}(z)}{Y^c - Y^b} \right]_{X^c} + \left[\frac{p^{(ab)}(z)}{X^a - X^b} \right]_{Y^a} + \left[\frac{p^{(ab)}(z)}{Y^b - Y^a} \right]_{X^a} \\
 & \left[\frac{p^{(ab)}(z)}{Y^b - Y^c} - \frac{p^{(bc)}(z)}{X^c - X^a} \right]_{Y^b} + \left[\frac{p^{(ab)}(z)}{Y^c - Y^a} - \frac{p^{(bc)}(z)}{Y^b - X^b} \right]_{X^b} = \beta_3 + \nu_3 \\
 & \left[\frac{p^{(ab)}(z)}{X^a - X^b} \right]_{Y^b} \\
 & + \left[\frac{p^{(BM2-a)}(z)}{Y^a - Y^{BM2}} \right]_{X^{BM2}} + \left[\frac{p^{(BM2-a)}(z)}{X^{BM2} - X^a} \right]_{Y^{BM2}} + \left[\frac{p^{(BM2-a)}(z)}{Y^{BM2} - Y^a} \right]_{X^a} \\
 & \left[\frac{p^{(ab)}(z)}{Y^a - Y^b} - \frac{p^{(BM2-a)}(z)}{Y^b - X^{BM2}} \right]_{Y^a} + \left[\frac{p^{(ab)}(z)}{Y^b - X^a} - \frac{p^{(BM2-a)}(z)}{X^{BM2} - Y^a} \right]_{X^a} = \beta_2 + \nu_2
 \end{aligned}$$

5. Menghitung elemen matriks konstanta (F) yang didapatkan dari selisih hitungan antara data ukuran dengan data hitungan pendekatan.

elemen matriks F :

Untuk jarak :

$$d_{BM2-A} - d_{BM2-A}^{\circ}$$

$$d_{AB} - d_{AB}^{\circ}$$

$$d_{BC} - d_{BC}^{\circ}$$

$$d_{C-n} - d_{C-n}^{\circ}$$

$$d_{n-BM3} - d_{n-BM3}^{\circ}$$

Untuk sudut :

$$\beta_1 - \beta_1^{\circ}$$

$$\beta_2 - \beta_2^{\circ}$$

$$\beta_3 - \beta_3^{\circ}$$

$$\beta_4 - \beta_4^{\circ}$$

$$\beta_n - \beta_n^{\circ}$$

$$\beta_{n+1} - \beta_{n+1}^{\circ}$$

$$\dots\dots\dots (2.23)$$

Matriks F

$$F = \begin{matrix} m \times n \\ \left| \begin{array}{c} d_{BM2-A} - d_{BM2-A}^{\circ} \\ d_{AB} - d_{AB}^{\circ} \\ d_{BC} - d_{BC}^{\circ} \\ d_{C-n} - d_{C-n}^{\circ} \\ d_{n-BM3} - d_{n-BM3}^{\circ} \\ \beta_1 - \beta_1^{\circ} \\ \beta_2 - \beta_2^{\circ} \\ \beta_3 - \beta_3^{\circ} \\ \beta_4 - \beta_4^{\circ} \\ \beta_n - \beta_n^{\circ} \\ \beta_{n+1} - \beta_{n+1}^{\circ} \end{array} \right| \end{matrix}$$

6. Menghitung elemen matriks bobot (P).

- Apabila bobot pengukuran (P) dihitung, maka bobot adalah berbanding terbalik dengan standard deviasinya (S^2) yaitu :

a. Untuk jarak (d)

$$P_{dij} = \frac{1}{(S_{dij})^2}$$

b. Untuk sudut (β)

$$P_{\beta_{jik}} = \frac{1}{(S_{\beta_{jik}})^2}$$

Sehingga matriks P adalah :

$P =$
 $m \times m$

d_{BM2-A}	d_{AB}	d_{BC}	d_{C-a}	d_{n-BM3}	β_1	β_2	β_3	β_4	β_n	β_{n+1}
$\frac{1}{(S_{dBM2-A})^2}$	0	0	0	0	0	0	0	0	0	0
	...									
		...								
										
										
0	0	0	0	0	$\frac{1}{(S_{\beta_1})^2}$	0	0	0	0	0
										
										
										
										
										

- Apabila bobot pengukuran (P) diasumsikan sama, maka matriks bobot (P) dianggap 1

7. Menghitung elemen matriks N dengan rumus sebagai berikut :

$$N = A^T P A \quad (\text{weight / diberi bobot})$$

$$N = A^T A \quad (\text{unweight / tidak diberi bobot}) \dots\dots\dots (2.24)$$

8. Menghitung matriks Q (matriks variance dan covariance)

- Matriks variance adalah kolom matriks yang berhubungan antara 1 titik dengan titik yang lain
- Matriks covariance adalah kolom matriks yang berhubungan antara 1 titik dengan titik itu sendiri.

$$Q = N^{-1}$$

$$= (A^T P A)^{-1} \rightarrow \text{weight / diberi bobot}$$

atau

$$Q = N^{-1}$$

$$= (A^T A)^{-1} \rightarrow \text{unweight / tidak diberi bobot} \dots\dots\dots (2.25)$$

9. Menghitung elemen matriks parameter (X)

$$X = Q \cdot (A^T P F)$$

$$= (A^T P A)^{-1} (A^T P F)$$

atau

$$X = Q \cdot (A^T F)$$

$$= (A^T A)^{-1} (A^T F) \dots\dots\dots (2.26)$$

10. Menghitung koordinat titik – titik poligon dengan rumus :

➤ Koordinat Titik A

$$X_A = X_A^\circ + \Delta X_A \quad , \quad Y_A = Y_A^\circ + \Delta Y_A$$

➤ Koordinat Titik B

$$X_B = X_B^\circ + \Delta X_B \quad , \quad Y_B = Y_B^\circ + \Delta Y_B$$

➤ Koordinat Titik C

$$X_C = X_C^\circ + \Delta X_C \quad , \quad Y_C = Y_C^\circ + \Delta Y_C$$

➤ Koordinat Titik n

$$X_n = X_n^\circ + \Delta X_n \quad , \quad Y_n = Y_n^\circ + \Delta Y_n \quad \dots\dots (2.27)$$

11. Proses iterasi (pengulangan sampai didapatkan nilai elemen matriks X yaitu

$$\Delta X \text{ dan } \Delta Y = 10^{-5} \text{ m})$$

12. Selanjutnya mencari elemen matriks residu (V) dengan rumus :

$$V = A.X - F \dots\dots\dots (2.28)$$

13. Menghitung standard deviasi (S_o) dengan rumus :

$$S_o = \sqrt{\frac{V^T \cdot V}{m-n}} \dots\dots\dots (2.29)$$

14. Dari standard deviasi (S_o) tersebut, maka standard deviasi untuk koordinat X

dan Y adalah sebagai berikut :

$$S_X = \sqrt{S_o^2 \cdot Q_{XX}}$$

$$S_{XA} = \sqrt{S_o^2 \cdot Q_{XA}}$$

$$S_{XB} = \sqrt{S_o^2 \cdot Q_{XB}}$$

$$S_{XC} = \sqrt{S_o^2 \cdot Q_{XC}}$$

$$S_{XD} = \sqrt{S_o^2 \cdot Q_{XD}}$$

$$\begin{aligned}
 S_Y &= \sqrt{S_o.^2 \cdot Q_{YY}} \\
 S_{YA} &= \sqrt{S_o.^2 \cdot Q_{YYA}} \\
 S_{YB} &= \sqrt{S_o.^2 \cdot Q_{YYB}} \\
 S_{YC} &= \sqrt{S_o.^2 \cdot Q_{YYC}} \\
 S_{YD} &= \sqrt{S_o.^2 \cdot Q_{YYD}} \dots\dots\dots (2.30)
 \end{aligned}$$

15. Menghitung Ketelitian Linier Jarak, dengan rumus :

$$\begin{aligned}
 cd &= \sqrt{(dX)^2 + (dY)^2} \\
 KL &= \frac{cd}{\sum d} \dots\dots\dots (2.31)
 \end{aligned}$$

Keterangan rumus :

KL = Ketelitian Linier

cd = Kesalahan Jarak

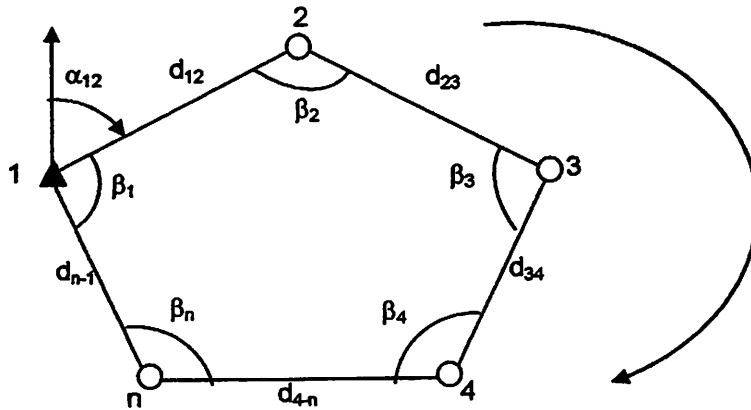
Σd = Total Jarak Terukur

II.2.3.2. Poligon Tertutup

Tahapan hitung perataan parameter untuk poligon tertutup adalah sebagai berikut :

1. Menghitung azimuth pendekatan pada poligon tertutup dibedakan antara menggunakan sudut dalam dan sudut luar. Apabila menggunakan sudut dalam,

maka azimuth pendekatan suatu titik didapatkan dari $\alpha_{awal} + 180^\circ - \beta$. (arah perhitungan searah dengan jarum jam)



Gambar 2.7. Poligon Tertutup dengan Sudut Dalam

Keterangan gambar :

α_{12} = azimuth awal

$\beta_1, \beta_2, \beta_3, \beta_4, \beta_n$ = sudut horisontal (sudut dalam)

$d_{12}, d_{23}, d_{34}, d_{4-n}, \dots$ = jarak horisontal

Rumus azimuth pendekatan :

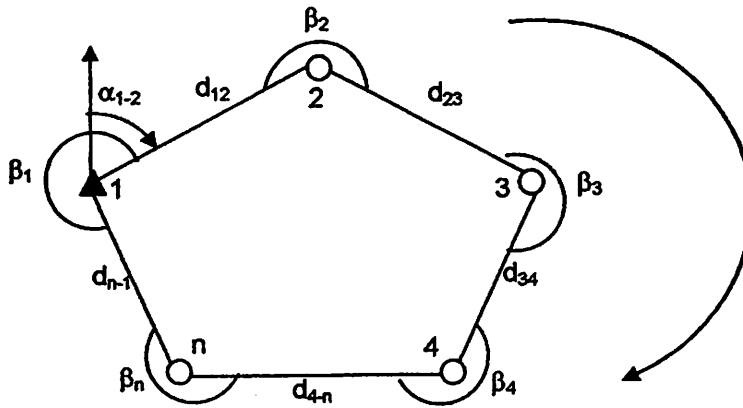
$$\alpha_{23}^\circ = \alpha_{12}^\circ + 180^\circ - \beta_2$$

$$\alpha_{34}^\circ = \alpha_{23}^\circ + 180^\circ - \beta_3$$

$$\alpha_{4-n}^\circ = \alpha_{34}^\circ + 180^\circ - \beta_4$$

$$\alpha_{n-1}^\circ = \alpha_{4-n}^\circ + 180^\circ - \beta_n \dots \dots \dots (2.31)$$

Apabila menggunakan sudut luar, maka azimuth pendekatan suatu titik didapatkan dari $\alpha_{awal} + \beta - 180^\circ$. (arah perhitungan searah dengan jarum jam)



Gambar 2.8. Poligon Tertutup dengan Sudut Luar

Keterangan gambar :

α_{12} = azimuth awal

$\beta_1, \beta_2, \beta_3, \beta_4, \beta_n$ = sudut horisontal (sudut luar)

$d_{12}, d_{23}, d_{34}, d_{4-n}, \dots$ = jarak horisontal

Rumus azimuth pendekatan :

$$\alpha_{23}^\circ = \alpha_{12} + \beta_2 - 180^\circ$$

$$\alpha_{34}^\circ = \alpha_{23}^\circ + \beta_3 - 180^\circ$$

$$\alpha_{4-n}^\circ = \alpha_{34}^\circ + \beta_4 - 180^\circ$$

$$\alpha_{n-1}^\circ = \alpha_{4-n}^\circ + \beta_n - 180^\circ \dots\dots\dots (2.32)$$

2. Menghitung koordinat pendekatan titik poligon dimana koordinat titik 1 sebagai koordinat awal (X_a, Y_a), dengan rumus sebagai berikut :

$$X_{a+1}^{\circ} = X_a + d_{a-(a+1)} \sin \alpha_{a-(a+1)}^{\circ}$$

$$Y_{a+1}^{\circ} = Y_a + d_{a-(a+1)} \cos \alpha_{a-(a+1)}^{\circ}$$

Sehingga : $X_2^{\circ} = X_1 + d_{12} \sin \alpha_{12}$

$$Y_2^{\circ} = Y_1 + d_{12} \cos \alpha_{12}$$

$$X_3^{\circ} = X_2^{\circ} + d_{23} \sin \alpha_{23}^{\circ}$$

$$Y_3^{\circ} = Y_2^{\circ} + d_{23} \cos \alpha_{23}^{\circ}$$

$$X_4^{\circ} = X_3^{\circ} + d_{34} \sin \alpha_{34}^{\circ}$$

$$Y_4^{\circ} = Y_3^{\circ} + d_{34} \cos \alpha_{34}^{\circ}$$

$$X_n^{\circ} = X_4^{\circ} + d_{4-n} \sin \alpha_{4-n}^{\circ}$$

$$Y_n^{\circ} = Y_4^{\circ} + d_{4-n} \cos \alpha_{4-n}^{\circ} \dots\dots\dots (2.33)$$

3. Menghitung jarak pendekatan (d°) dengan rumus sebagai berikut :

Jarak pendekatan (d°)

$$d_{a-(a+1)}^{\circ} = \sqrt{(X_{a+1}^{\circ} - X_a)^2 + (Y_{a+1}^{\circ} - Y_a)^2}$$

$$d_{12}^{\circ} = \sqrt{(X_2^{\circ} - X_1)^2 + (Y_2^{\circ} - Y_1)^2}$$

$$d_{23}^{\circ} = \sqrt{(X_3^{\circ} - X_2^{\circ})^2 + (Y_3^{\circ} - Y_2^{\circ})^2}$$

$$d_{34}^{\circ} = \sqrt{(X_4^{\circ} - X_3^{\circ})^2 + (Y_4^{\circ} - Y_3^{\circ})^2}$$

$$d_{4-n}^{\circ} = \sqrt{(X_n^{\circ} - X_4^{\circ})^2 + (Y_n^{\circ} - Y_4^{\circ})^2}$$

$$d_{n-1}^{\circ} = \sqrt{(X_1 - X_n^{\circ})^2 + (Y_1 - Y_n^{\circ})^2} \dots\dots\dots (2.34)$$

4. Menghitung sudut pendekatan (β°) dengan rumus sebagai berikut :

Sudut dalam = $180^\circ + \text{azimuth awal} - \text{azimuth suatu titik}$

$$\beta_2^\circ = 180^\circ + \arctg \left[\frac{X_2^\circ - X_1^\circ}{Y_2^\circ - Y_1^\circ} \right] - \arctg \left[\frac{X_3^\circ - X_2^\circ}{Y_3^\circ - Y_2^\circ} \right]$$

$$\beta_3^\circ = 180^\circ + \arctg \left[\frac{X_3^\circ - X_2^\circ}{Y_3^\circ - Y_2^\circ} \right] - \arctg \left[\frac{X_4^\circ - X_3^\circ}{Y_4^\circ - Y_3^\circ} \right]$$

$$\beta_4^\circ = 180^\circ + \arctg \left[\frac{X_4^\circ - X_3^\circ}{Y_4^\circ - Y_3^\circ} \right] - \arctg \left[\frac{X_5^\circ - X_4^\circ}{Y_5^\circ - Y_4^\circ} \right]$$

$$\beta_n^\circ = 180^\circ + \arctg \left[\frac{X_n^\circ - X_4^\circ}{Y_n^\circ - Y_4^\circ} \right] - \arctg \left[\frac{X_1 - X_n^\circ}{Y_1 - Y_n^\circ} \right]$$

$$\beta_1^\circ = 180^\circ + \arctg \left[\frac{X_1 - X_n^\circ}{Y_1 - Y_n^\circ} \right] - \arctg \left[\frac{X_2^\circ - X_1^\circ}{Y_2^\circ - Y_1^\circ} \right] \dots\dots\dots (2.35)$$

Sudut luar = $180^\circ + \text{azimuth suatu titik} - \text{azimuth awal}$

$$\beta_2^\circ = 180^\circ + \arctg \left[\frac{X_3^\circ - X_2^\circ}{Y_3^\circ - Y_2^\circ} \right] - \arctg \left[\frac{X_2^\circ - X_1^\circ}{Y_2^\circ - Y_1^\circ} \right]$$

$$\beta_3^\circ = 180^\circ + \arctg \left[\frac{X_4^\circ - X_3^\circ}{Y_4^\circ - Y_3^\circ} \right] - \arctg \left[\frac{X_3^\circ - X_2^\circ}{Y_3^\circ - Y_2^\circ} \right]$$

$$\beta_4^\circ = 180^\circ + \arctg \left[\frac{X_n^\circ - X_4^\circ}{Y_n^\circ - Y_4^\circ} \right] - \arctg \left[\frac{X_4^\circ - X_3^\circ}{Y_4^\circ - Y_3^\circ} \right]$$

$$\beta_n^\circ = 180^\circ + \arctg \left[\frac{X_1 - X_n^\circ}{Y_1 - Y_n^\circ} \right] - \arctg \left[\frac{X_n^\circ - X_4^\circ}{Y_n^\circ - Y_4^\circ} \right]$$

$$\beta_1^\circ = 180^\circ + \arctg \left[\frac{X_2^\circ - X_1^\circ}{Y_2^\circ - Y_1^\circ} \right] - \arctg \left[\frac{X_1 - X_n^\circ}{Y_1 - Y_n^\circ} \right] \dots\dots\dots (2.36)$$

5. Menghitung elemen matriks disain (A) yang didapatkan dari hasil linierisasi jarak dan sudut. Untuk linierisasi sudut dikalikan dengan nilai rho (ρ) = 206265"/rad.

➤ Linierisasi jarak

$$d_{12} = d_{12}^{\circ} + \left[\frac{\delta d}{\delta X_1} \right] \Delta X_1 + \left[\frac{\delta d}{\delta Y_1} \right] \Delta Y_1 + \left[\frac{\delta d}{\delta X_2} \right] \Delta X_2 + \left[\frac{\delta d}{\delta Y_2} \right] \Delta Y_2$$

$$\frac{\delta d}{\delta X_1} = - \left[\frac{X_2^{\circ} - X_1^{\circ}}{d_{12}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_1} = - \left[\frac{Y_2^{\circ} - Y_1^{\circ}}{d_{12}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_2} = + \left[\frac{X_2^{\circ} - X_1^{\circ}}{d_{12}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_2} = + \left[\frac{Y_2^{\circ} - Y_1^{\circ}}{d_{12}^{\circ}} \right]$$

$$d_{23} = d_{23}^{\circ} + \left[\frac{\delta d}{\delta X_2} \right] \Delta X_2 + \left[\frac{\delta d}{\delta Y_2} \right] \Delta Y_2 + \left[\frac{\delta d}{\delta X_3} \right] \Delta X_3 + \left[\frac{\delta d}{\delta Y_3} \right] \Delta Y_3$$

$$\frac{\delta d}{\delta X_2} = - \left[\frac{X_3^{\circ} - X_2^{\circ}}{d_{23}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_2} = - \left[\frac{Y_3^{\circ} - Y_2^{\circ}}{d_{23}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_3} = + \left[\frac{X_3^{\circ} - X_2^{\circ}}{d_{23}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_3} = + \left[\frac{Y_3^{\circ} - Y_2^{\circ}}{d_{23}^{\circ}} \right]$$

$$d_{34} = d_{34}^{\circ} + \left[\frac{\delta d}{\delta X_3} \right] \Delta X_3 + \left[\frac{\delta d}{\delta Y_3} \right] \Delta Y_3 + \left[\frac{\delta d}{\delta X_4} \right] \Delta X_4 + \left[\frac{\delta d}{\delta Y_4} \right] \Delta Y_4$$

$$\frac{\delta d}{\delta X_3} = - \left[\frac{X_4^{\circ} - X_3^{\circ}}{d_{34}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_3} = - \left[\frac{Y_4^{\circ} - Y_3^{\circ}}{d_{34}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_4} = + \left[\frac{X_4^{\circ} - X_3^{\circ}}{d_{34}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_4} = + \left[\frac{Y_4^{\circ} - Y_3^{\circ}}{d_{34}^{\circ}} \right]$$

$$d_{4-n} = d_{4-n}^{\circ} + \left[\frac{\delta d}{\delta X_4} \right] \Delta X_4 + \left[\frac{\delta d}{\delta Y_4} \right] \Delta Y_4 + \left[\frac{\delta d}{\delta X_n} \right] \Delta X_n + \left[\frac{\delta d}{\delta Y_n} \right] \Delta Y_n$$

$$\frac{\delta d}{\delta X_4} = - \left[\frac{X_n^{\circ} - X_4^{\circ}}{d_{4-n}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_4} = - \left[\frac{Y_n^{\circ} - Y_4^{\circ}}{d_{4-n}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_n} = + \left[\frac{X_n^{\circ} - X_4^{\circ}}{d_{4-n}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_n} = + \left[\frac{Y_n^{\circ} - Y_4^{\circ}}{d_{4-n}^{\circ}} \right]$$

$$d_{n-1} = d_{n-1}^{\circ} + \left[\frac{\delta d}{\delta X_n} \right] \Delta X_n + \left[\frac{\delta d}{\delta Y_n} \right] \Delta Y_n + \left[\frac{\delta d}{\delta X_1} \right] \Delta X_1 + \left[\frac{\delta d}{\delta Y_1} \right] \Delta Y_1$$

$$\frac{\delta d}{\delta X_n} = - \left[\frac{X_1 - X_n^{\circ}}{d_{n-1}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_n} = - \left[\frac{Y_1 - Y_n^{\circ}}{d_{n-1}^{\circ}} \right]$$

$$\frac{\delta d}{\delta X_1} = + \left[\frac{X_1 - X_n^{\circ}}{d_{n-1}^{\circ}} \right] \quad \frac{\delta d}{\delta Y_1} = + \left[\frac{Y_1 - Y_n^{\circ}}{d_{n-1}^{\circ}} \right]$$

..... (2.37)

➤ Linierisasi Sudut

❖ Sudut dalam

$$\beta_2 + V_2 = \left[\frac{Y_1 - Y_2^{\circ}}{(d_{12}^{\circ})^2} - \frac{Y_3^{\circ} - Y_2^{\circ}}{(d_{23}^{\circ})^2} \right] dX_2 + \left[\frac{X_2^{\circ} - X_1}{(d_{AB}^{\circ})^2} - \frac{X_2^{\circ} - X_3^{\circ}}{(d_{23}^{\circ})^2} \right] dY_2$$

$$+ \left[\frac{Y_2^{\circ} - Y_1}{(d_{12}^{\circ})^2} \right] dX_1 + \left[\frac{X_1 - X_2^{\circ}}{(d_{12}^{\circ})^2} \right] dY_1 + \left[\frac{Y_3^{\circ} - Y_2^{\circ}}{(d_{23}^{\circ})^2} \right] dX_3 + \left[\frac{X_2^{\circ} - X_3^{\circ}}{(d_{23}^{\circ})^2} \right] dY_3$$

$$\beta_3 + V_3 = \left[\frac{Y_2^{\circ} - Y_3^{\circ}}{(d_{23}^{\circ})^2} - \frac{Y_4^{\circ} - Y_3^{\circ}}{(d_{34}^{\circ})^2} \right] dX_3 + \left[\frac{X_3^{\circ} - X_2^{\circ}}{(d_{23}^{\circ})^2} - \frac{X_3^{\circ} - X_4^{\circ}}{(d_{34}^{\circ})^2} \right] dY_3$$

$$+ \left[\frac{Y_3^{\circ} - Y_2^{\circ}}{(d_{23}^{\circ})^2} \right] dX_2 + \left[\frac{X_2^{\circ} - X_3^{\circ}}{(d_{23}^{\circ})^2} \right] dY_2 + \left[\frac{Y_4^{\circ} - Y_3^{\circ}}{(d_{34}^{\circ})^2} \right] dX_4 + \left[\frac{X_3^{\circ} - X_4^{\circ}}{(d_{34}^{\circ})^2} \right] dY_4$$

$$\begin{aligned}
 & \left[\frac{p(z)}{Y_3^o - Y_2^o} \right] + \left[\frac{p(z)}{Y_2^o - X_3^o} \right] + \left[\frac{p(z)}{Y_4^o - Y_3^o} \right] + \left[\frac{p(z)}{X_3^o - X_4^o} \right] \\
 & \left[\frac{p(z)}{Y_3^o - Y_2^o} - \frac{p(z)}{Y_4^o - Y_3^o} \right] + \left[\frac{p(z)}{X_3^o - X_4^o} - \frac{p(z)}{X_2^o - X_3^o} \right] = \beta_3 + \nu_3 \\
 & \left[\frac{p(z)}{Y_2^o - Y_1^o} \right] + \left[\frac{p(z)}{X_1^o - X_2^o} \right] + \left[\frac{p(z)}{Y_3^o - Y_2^o} \right] + \left[\frac{p(z)}{X_2^o - X_3^o} \right] \\
 & \left[\frac{p(z)}{Y_2^o - Y_1^o} - \frac{p(z)}{Y_3^o - Y_2^o} \right] + \left[\frac{p(z)}{X_1^o - X_2^o} - \frac{p(z)}{X_2^o - X_3^o} \right] = \beta_2 + \nu_2
 \end{aligned}$$

❖ Sudut luar

(2.38)

$$\begin{aligned}
 & \left[\frac{p(z)}{Y_1^n - Y_2^n} \right] + \left[\frac{p(z)}{X_1^n - X_2^n} \right] + \left[\frac{p(z)}{Y_2^n - Y_3^n} \right] + \left[\frac{p(z)}{X_2^n - X_3^n} \right] \\
 & \left[\frac{p(z)}{Y_1^n - Y_2^n} - \frac{p(z)}{Y_2^n - Y_3^n} \right] + \left[\frac{p(z)}{X_1^n - X_2^n} - \frac{p(z)}{X_2^n - X_3^n} \right] = \beta_1 + \nu_1 \\
 & \left[\frac{p(z)}{Y_2^n - Y_3^n} \right] + \left[\frac{p(z)}{X_2^n - X_3^n} \right] + \left[\frac{p(z)}{Y_3^n - Y_4^n} \right] + \left[\frac{p(z)}{X_3^n - X_4^n} \right] \\
 & \left[\frac{p(z)}{Y_2^n - Y_3^n} - \frac{p(z)}{Y_3^n - Y_4^n} \right] + \left[\frac{p(z)}{X_2^n - X_3^n} - \frac{p(z)}{X_3^n - X_4^n} \right] = \beta_n + \nu_n \\
 & \left[\frac{p(z)}{Y_3^n - Y_4^n} \right] + \left[\frac{p(z)}{X_3^n - X_4^n} \right] + \left[\frac{p(z)}{Y_4^n - Y_5^n} \right] + \left[\frac{p(z)}{X_4^n - X_5^n} \right] \\
 & \left[\frac{p(z)}{Y_3^n - Y_4^n} - \frac{p(z)}{Y_4^n - Y_5^n} \right] + \left[\frac{p(z)}{X_3^n - X_4^n} - \frac{p(z)}{X_4^n - X_5^n} \right] = \beta_4 + \nu_4
 \end{aligned}$$

$$\begin{aligned}
\beta_4 + V_4 &= \left[\frac{Y_4^\circ - Y_n^\circ}{(d_{4-n}^\circ)^2} - \frac{Y_4^\circ - Y_3^\circ}{(d_{34}^\circ)^2} \right] dX_4 + \left[\frac{X_n^\circ - X_4^\circ}{(d_{4-n}^\circ)^2} - \frac{X_3^\circ - X_4^\circ}{(d_{34}^\circ)^2} \right] dY_4 \\
&+ \left[\frac{Y_4^\circ - Y_3^\circ}{(d_{34}^\circ)^2} \right] dX_3 + \left[\frac{X_3^\circ - X_4^\circ}{(d_{34}^\circ)^2} \right] dY_3 + \left[\frac{Y_n^\circ - Y_4^\circ}{(d_{4-n}^\circ)^2} \right] dX_n + \left[\frac{X_4^\circ - X_n^\circ}{(d_{4-n}^\circ)^2} \right] dY_n \\
\beta_n + V_n &= \left[\frac{Y_n^\circ - Y_1^\circ}{(d_{n-1}^\circ)^2} - \frac{Y_n^\circ - Y_4^\circ}{(d_{4-n}^\circ)^2} \right] dX_n + \left[\frac{X_1^\circ - X_n^\circ}{(d_{n-1}^\circ)^2} - \frac{X_4^\circ - X_n^\circ}{(d_{4-n}^\circ)^2} \right] dY_n \\
&+ \left[\frac{Y_n^\circ - Y_4^\circ}{(d_{4-n}^\circ)^2} \right] dX_4 + \left[\frac{X_4^\circ - X_n^\circ}{(d_{4-n}^\circ)^2} \right] dY_4 + \left[\frac{Y_1^\circ - Y_n^\circ}{(d_{n-1}^\circ)^2} \right] dX_1 + \left[\frac{X_n^\circ - X_1^\circ}{(d_{n-1}^\circ)^2} \right] dY_1 \\
\beta_1 + V_1 &= \left[\frac{Y_1^\circ - Y_2^\circ}{(d_{12}^\circ)^2} - \frac{Y_1^\circ - Y_n^\circ}{(d_{n-1}^\circ)^2} \right] dX_1 + \left[\frac{X_2^\circ - X_1^\circ}{(d_{12}^\circ)^2} - \frac{X_n^\circ - X_1^\circ}{(d_{n-1}^\circ)^2} \right] dY_1 \\
&+ \left[\frac{Y_1^\circ - Y_n^\circ}{(d_{n-1}^\circ)^2} \right] dX_n + \left[\frac{X_n^\circ - X_1^\circ}{(d_{n-1}^\circ)^2} \right] dY_n + \left[\frac{Y_2^\circ - Y_1^\circ}{(d_{12}^\circ)^2} \right] dX_2 + \left[\frac{X_1^\circ - X_2^\circ}{(d_{12}^\circ)^2} \right] dY_2 \\
&\dots\dots\dots (2.39)
\end{aligned}$$

6. Menghitung elemen matriks konstanta (F) yang didapatkan dari selisih antara data ukuran dengan data hitungan pendekatan.

elemen matriks F :

Untuk jarak :

$$d_{12} - d_{12}^\circ$$

$$d_{23} - d_{23}^\circ$$

$$d_{34} - d_{34}^\circ$$

$$d_{4-n} - d_{4-n}^\circ$$

$$d_{n-1} - d_{n-1}^\circ$$

Untuk sudut :

$$\begin{array}{cc}
 \beta_1 - \beta_1^\circ & \beta_2 - \beta_2^\circ \\
 \beta_3 - \beta_3^\circ & \beta_4 - \beta_4^\circ \\
 \beta_n - \beta_n^\circ & \\
 & \dots\dots\dots
 \end{array} \quad (2.40)$$

Matriks F

$$F = \begin{array}{c} m \times n \end{array} \left| \begin{array}{c} d_{12} - d_{12}^\circ \\ d_{23} - d_{23}^\circ \\ d_{34} - d_{34}^\circ \\ d_{4-n} - d_{4-n}^\circ \\ d_{n-1} - d_{n-1}^\circ \\ \beta_1 - \beta_1^\circ \\ \beta_2 - \beta_2^\circ \\ \beta_3 - \beta_3^\circ \\ \beta_4 - \beta_4^\circ \\ \beta_n - \beta_n^\circ \end{array} \right|$$

7. Menghitung elemen matriks bobot (P).

- Apabila bobot pengukuran (P) dihitung, maka bobot adalah berbanding terbalik dengan standard deviasinya (S^2) yaitu :

a. Untuk jarak (d)

$$P_{dij} = \frac{1}{(S_{dij})^2}$$

b. Untuk sudut (β)

$$P_{\beta_{ijk}} = \frac{1}{(S_{\beta_{ijk}})^2}$$

Sehingga matriks P adalah :

$$P = \begin{matrix} & d_{12} & d_{23} & d_{34} & d_{4-n} & d_{n-1} & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_n \\ \begin{matrix} m \times m \\ P = \end{matrix} & \frac{1}{(S_{dB M2-A})^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & \dots & & & & & & & & \\ & & & \dots & & & & & & & \\ & & & & \dots & & & & & & \\ & & & & & \dots & & & & & \\ & 0 & 0 & 0 & 0 & 0 & \frac{1}{(S_{\beta 1})^2} & 0 & 0 & 0 & 0 \\ & & & & & & & \dots & & & \\ & & & & & & & & \dots & & \\ & & & & & & & & & \dots & \\ & & & & & & & & & & \dots \end{matrix}$$

- Apabila bobot pengukuran (P) diasumsikan sama, maka matriks bobot (P) dianggap 1.

8. Menghitung elemen matriks N dengan rumus sebagai berikut :

$$N = A^T P A \quad (\text{weight / diberi bobot})$$

atau

$$N = A^T A \quad (\text{unweight / tidak diberi bobot}) \dots\dots\dots (2.41)$$

9. Menghitung matriks Q (matriks variance dan covariance)

- Matriks variance adalah kolom matriks yang berhubungan antara 1 titik dengan titik yang lain
- Matriks covariance adalah kolom matriks yang berhubungan antara 1 titik dengan titik itu sendiri.

$$Q = N^{-1}$$

$$= (A^T P A)^{-1} \rightarrow \text{weight / diberi bobot}$$

atau

$$\begin{aligned} Q &= N^{-1} \\ &= (A^T A)^{-1} \rightarrow \text{unweight / tidak diberi bobot} \dots\dots (2.42) \end{aligned}$$

10. Menghitung elemen matriks parameter (X)

$$\begin{aligned} X &= Q \cdot (A^T P F) \\ &= (A^T P A)^{-1} (A^T P F) \end{aligned}$$

atau

$$\begin{aligned} X &= Q \cdot (A^T F) \\ &= (A^T A)^{-1} (A^T F) \dots\dots\dots (2.43) \end{aligned}$$

11. Menghitung koordinat titik – titik poligon dengan rumus :

➤ Koordinat Titik 2

$$X_2 = X_2^\circ + \Delta X_2$$

$$Y_2 = Y_2^\circ + \Delta Y_2$$

➤ Koordinat Titik 3

$$X_3 = X_3^\circ + \Delta X_3$$

$$Y_3 = Y_3^\circ + \Delta Y_3$$

➤ Koordinat Titik 4

$$X_4 = X_4^\circ + \Delta X_4$$

$$Y_4 = Y_4^\circ + \Delta Y_4$$

➤ Koordinat Titik n

$$X_n = X_n^\circ + \Delta X_n$$

$$Y_n = Y_n^\circ + \Delta Y_n \dots\dots\dots (2.44)$$

12. Proses iterasi (pengulangan sampai didapatkan nilai elemen matriks X yaitu

ΔX dan $\Delta Y = 10^{-5}$ m)

13. Selanjutnya mencari elemen matriks residu (V) dengan rumus :

$V = A.X - F$ (2.45)

14. Menghitung standard deviasi (S_o) dengan rumus :

$S_o = \sqrt{\frac{V^T.V}{m-n}}$ (2.46)

15. Dari standard deviasi (S_o) tersebut, maka standard deviasi untuk koordinat X dan Y adalah sebagai berikut :

$S_X = \sqrt{S_o.^2.Q_{XX}}$

$S_{X2} = \sqrt{S_o.^2.Q_{XX\ 2}}$ $S_{X3} = \sqrt{S_o.^2.Q_{XX\ 3}}$

$S_{X4} = \sqrt{S_o.^2.Q_{XX\ 4}}$ $S_{X5} = \sqrt{S_o.^2.Q_{XX\ 5}}$

$S_{Xn} = \sqrt{S_o.^2.Q_{XXn}}$

$S_Y = \sqrt{S_o.^2.Q_{YY}}$

$S_{Y2} = \sqrt{S_o.^2.Q_{YY2}}$ $S_{Y3} = \sqrt{S_o.^2.Q_{YY3}}$

$S_{Y4} = \sqrt{S_o.^2.Q_{YY4}}$ $S_{Y5} = \sqrt{S_o.^2.Q_{YY5}}$

$S_{Yn} = \sqrt{S_o.^2.Q_{YYn}}$

..... (2.47)

16. Menghitung Ketelitian Linier Jarak, dengan rumus :

$$cd = \sqrt{(dX)^2 + (dY)^2}$$

$$KL = \frac{cd}{\sum d} \dots\dots\dots (2.48)$$

Keterangan rumus :

KL = Ketelitian Linier cd = Kesalahan Jarak

$\sum d$ = Total Jarak Terukur

II.2.3.3. Ellips Kesalahan

Ellips kesalahan menggambarkan korelasi dari masing-masing kesalahan (arah X dan arah Y) serta kemiringannya pada 1 titik. Sumber hitungannya adalah matriks Q (matriks variance dan covariance)

$$Q = N^{-1}$$

$$= (A^T P A)^{-1} \text{ (weight / diberi bobot)}$$

$$= (A^T A)^{-1} \text{ (unweight / tidak diberi bobot)}$$

$Q = N^{-1} \begin{vmatrix} Q_{XX} & Q_{XY} \\ Q_{XY} & Q_{YY} \end{vmatrix}$

$\nearrow$ Variance

 $\searrow$ Covariance

$\begin{vmatrix} Q_{XX} & Q_{XY} \\ Q_{XY} & Q_{YY} \end{vmatrix}$

Tahapan perhitungannya yaitu :

1. Menghitung kemiringan sumbu ellips (t)

$$\text{tg } 2t = \frac{2 \cdot Q_{XY}}{Q_{YY} - Q_{XX}} \dots\dots\dots (2.49)$$

2. Menghitung K (pemepatan/penggepengan)

$$K = [(Q_{YY} - Q_{XX})^2 + 4. (Q_{XY})^2]^{1/2} \dots\dots\dots (2.50)$$

3. Menghitung kelengkungan ellips (S_o , S_u , S_v , S_X , S_Y)

$$S_o = \sqrt{\frac{V^T \cdot V}{m-n}} \quad m = \text{jumlah data ukuran}$$

$n = \text{jumlah unknown}$

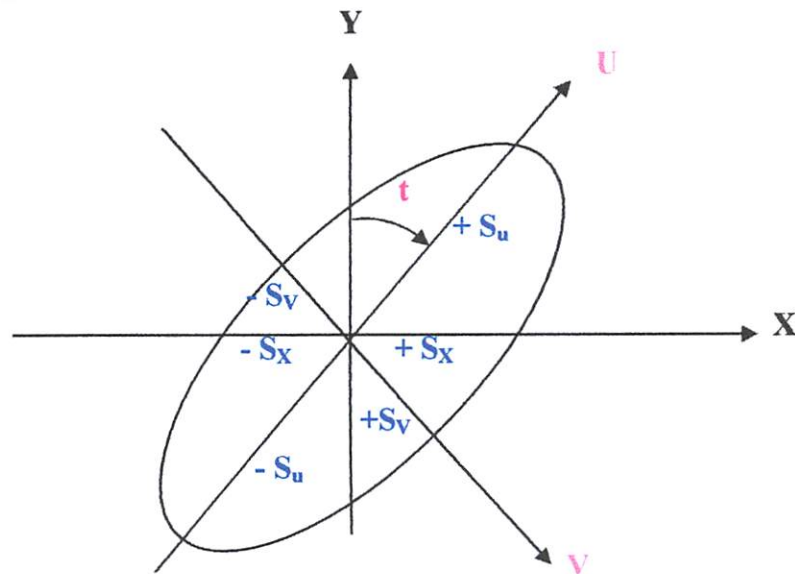
$$S_u = \pm S_o \sqrt{\frac{Q_{YY} + Q_{XX} + K}{2}}$$

$$S_v = \pm S_o \sqrt{\frac{Q_{YY} + Q_{XX} - K}{2}}$$

$$S_X = \pm S_o \sqrt{Q_{XX}}$$

$$S_Y = \pm S_o \sqrt{Q_{YY}} \dots\dots\dots (2.51)$$

Gambar Ellips Kesalahan

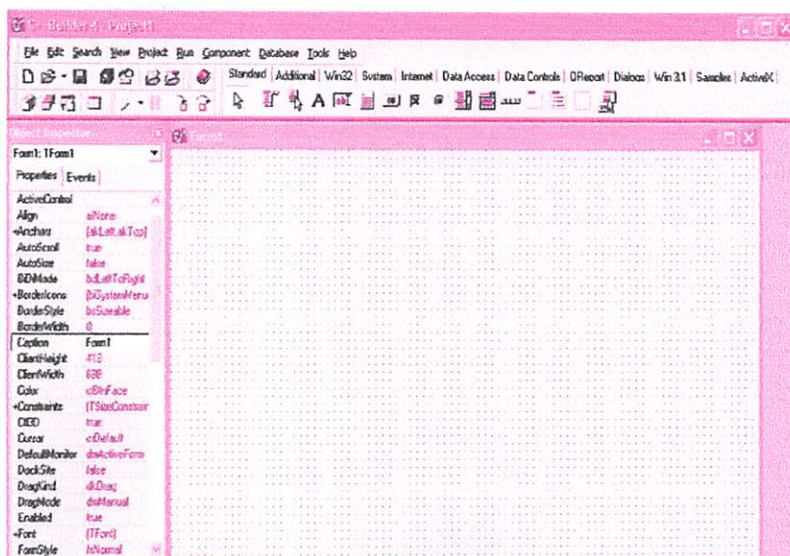


Gambar 2.9. Ellips Kesalahan

II.3. Borland C++ Builder

Borland C++ Builder adalah pemrograman berorientasi objek dengan lingkungan pemrograman visual untuk RAD (*Rapid Application Development*). Dengan C++ Builder dapat dibuat aplikasi Windows dengan efisiensi yang tinggi dengan meminimalkan pengkodean manual. C++ Builder memberikan sebuah *Library Component* yang komprehensif disertai dengan sederetan tool-tool RAD yang meliputi aplikasi, template form, dan wizard pemrograman. C++ Builder merupakan alat bantu pemrograman yang dikembangkan oleh Borland. Program ini dikembangkan karena semakin banyaknya perangkat lunak (*software*) berbasis Windows dan Linux yang dikembangkan dengan bahasa C dan C++, sehingga software yang dihasilkan lebih efisien serta meningkatkan *performance* dan *functionality*. (Wahana Komputer, 2004)

Lingkungan C++ Builder ini memberikan *tools* yang diperlukan untuk merancang, mengembangkan, menguji, dan debug program. Pada gambar 2.10 menunjukkan program C++ Builder pada saat pertama kali dijalankan.

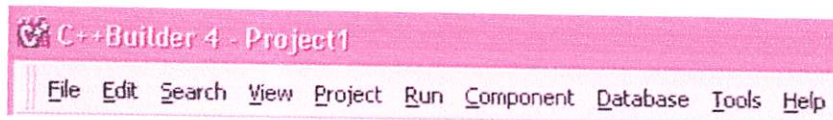


Gambar 2.10 Tampilan Program C++ Builder

Lingkungan pengembangan C++ Builder mempunyai beberapa bagian fleksibel yang dapat dipindahkan dimanapun pada layar. Lingkungan pengembangan itu terdiri dari menu utama, *toolbar*, dan *component palette*. Bagian lain yang secara otomatis akan ditampilkan pada saat memulai program C++ Builder adalah *object inspector*, *code editor*, dan sebuah *form*.

II.3.1. Menu Utama C++ Builder

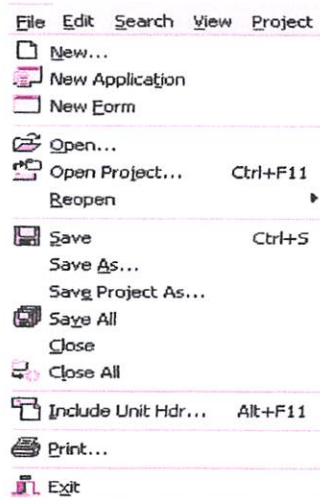
Menu-menu utama pada C++ Builder terdiri dari *File*, *Edit*, *Search*, *View*, *Project*, *Run*, *Component*, *Database*, *Tools*, *Option*, dan *Help*.



Gambar 2.11 Menu Utama

II.3.1.1. Menu File

Fungsi dari menu *File* diantaranya adalah untuk membuka (*open*), menyimpan (*save*), menutup (*exit*), dan mencetak *project* ataupun *file* (*print*).



Gambar 2.12 Menu File

Penjelasan dari tiap item pada menu File yaitu :

- **New Application** : untuk membuat project baru berupa sebuah form kosong. Perintah ini sama seperti ketika membuat program aplikasi baru melalui **File – New – Application**.
- **New Form** : untuk membuat dan menambah sebuah form kosong untuk disertakan ke dalam program aplikasi yang sedang dibuat. Perintah ini sama seperti ketika membuat dan menambah form baru melalui **File – New – Form**.
- **Open** : kotak dialog ini berisi *file project, file form, file unit*, atau *file text* yang disimpan di dalam Kode Editor.
- **Open Project** : kotak dialog ini hanya berisi file project .MAK yang merupakan file program dari program aplikasi yang sedang dibuat.
- **Reopen** : daftar dari beberapa project yang telah dibuat
- **Save** : untuk menyimpan file-file yang sedang dikembangkan pada program aplikasi yang sedang dibuat.
- **Save as** : untuk menyimpan file-file dari program aplikasi yang sedang dibuat dengan nama baru sesuai dengan keinginan. Perintah ini akan mengaktifkan kotak dialog konfirmasi penyimpanan khususnya menyimpan file-file unit.
- **Save all** : untuk menyimpan semua file-file dari program aplikasi yang sedang dibuat.

- **Close** : untuk menutup project yang sedang dibuat dan semua unit serta form yang berhubungan dengan project tersebut
- **Close all** : untuk menutup semua file yang masih aktif dalam lingkungan pemrograman C++ Builder
- **Print** : untuk mengirimkan file aktif ke printer
- **Exit** : untuk menutup project yang telah dibuka dan sekaligus keluar dari lingkungan pemrograman C++ Builder

II.3.1.2. Menu Edit

Perintah-perintah dari menu *Edit* diantaranya untuk mendisaian teks dan komponen pada waktu merancang sebuah program aplikasi. Perintah-perintah itu antara lain adalah :

- **Undo/Undelete** : perintah ini untuk membatalkan perintah terakhir yang diberikan pada C++ Builder.
- **Cut** : perintah ini untuk menghapus item-item yang telah dipilih untuk ditempatkan pada rancangan form.
- **Copy** : perintah ini untuk mengkopi item-item yang dipilih tetapi tetap meninggalkan item yang dijafikan *master*.
- **Paste** : menempatkan item-item dari hasil perintah copy ke dalam window Kode Editor maupun pada sebuah form.
- **Delete** : perintah ini berfungsi seperti halnya perintah Cut.
- **Align** : untuk mengatur komponen yang telah ditempatkan pada form rancangan



Gambar 2.13 Menu Edit

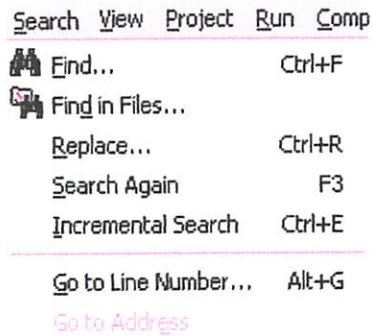
II.3.1.3. Menu Search

Perintah-perintah dari menu *Search* digunakan untuk menemukan lokasi teks, error, units, variable-variabel, dan simbol-simbol pada window Kode Editor.

Perintah-perintah yang terdapat pada menu *Search* diantaranya adalah :

- **Find** : untuk mengaktifkan kotak dialog pencarian khususnya pencarian teks. Untuk mengaktifkan kotak dialog ini pilih **Search – Find**.
- **Replace** : untuk menampilkan kotak dialog Replace, pilih **Search – Replace**
- **Search Again** : untuk mengulangi mode pencarian, dapat dengan menggunakan tombol F3 untuk mengaktifkan perintah ini.
- **Incremental Search** : untuk mencari teks seperti yang diketikkan pada kotak input
- **Go to Line Number** : untuk memindahkan cursor pada posisi yang diinginkan.

- **Go to Address** : menggunakan perintah ini untuk berpindah ke posisi alamat yang diinginkan.



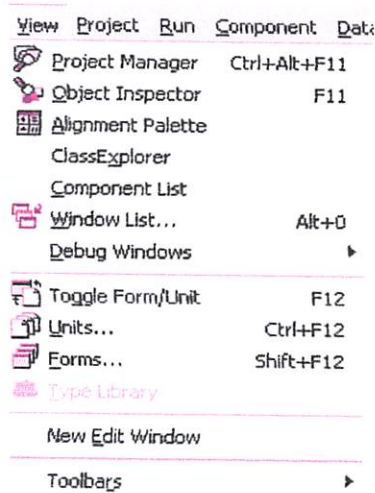
Gambar 2.14 Menu Search

II.3.1.4. Menu View

Perintah-perintah pada menu View digunakan untuk menampilkan atau menyembunyikan window atau elemen-elemen yang terdapat dalam lingkungan pemrograman C++ Builder. Perintah-perintah yang terdapat dalam menu View adalah sebagai berikut :

- **Project manager** : perintah ini untuk menampilkan window project manager.
- **Object Inspector** : untuk menampilkan window Object Inspector dapat dengan menekan tombol F11.
- **Alignment Pallette** : untuk menampilkan elemen alignment pallette.
- **Component List** : untuk menampilkan kotak dialog daftar komponen yang disediakan dalam lingkungan pemrograman C++ Builder
- **Window List** : untuk menampilkan daftar window-window yang terbuka pada saat membangun program aplikasi.

- **Units** : untuk menampilkan kotak dialog yang didalamnya terdapat daftar unit program aplikasi yang sedang dibangun.
- **Forms** : untuk menampilkan kotak dialog yang di dalamnya terdapat daftar form dari program aplikasi yang sedang dibangun.
- **Toolbar** : untuk menampilkan atau menyembunyikan toolbar pada lingkungan pemrograman C++ Builder.



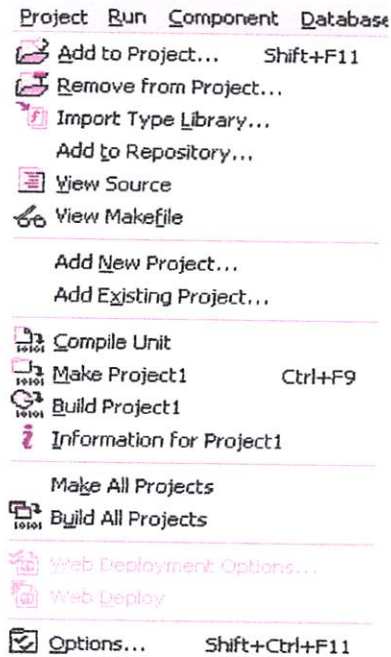
Gambar 2.15 Menu View

II.3.1.5. Menu Project

Perintah-perintah yang ada pada menu *Project* digunakan untuk mengcompile atau menjalankan program aplikasi yang sedang dibuat. Perintah-perintah yang ada pada menu *Project* diantaranya adalah :

- **Add to Project** : perintah ini untuk menambah file ke dalam project yang sedang dibuat.
- **Remove from Project** : perintah ini untuk menghapus file-file dalam project yang sedang dibuat.

- **Compile Unit** : perintah ini untuk mengcompile beberapa source kode yang telah diubah sejak pengcompilean terakhir.
- **Built All** : perintah ini untuk mengcompile segala sesuatu yang terdapat project termasuk source kode yang telah diubah.



Gambar 2.16 Menu Project

11.3.1.6. Menu Run

Perintah-perintah menu Run adalah perintah yang digunakan untuk debug program aplikasi yang sedang dibuat dalam C++ Builder. Adapun perintah-perintah penting yang biasa digunakan untuk mengeksekusi program diantaranya :

- **Run** : perintah ini untuk mengcompile dan mengeksekusi program aplikasi yang dibuat.
- **Parameters** : perintah ini untuk menentukan parameter startup untuk program aplikasi.

- **Step Over** : perintah ini untuk mengeksekusi program aplikasi baris demi baris, pengeksekusian program ini diperlakukan sebagai sebuah unit tunggal.
- **Program Pause** : perintah ini digunakan untuk menghentikan pengeksekusian program untuk sementara waktu.
- **Program Reset** : perintah ini digunakan untuk mengakhiri pengeksekusian program dan juga melepaskan pengeksekusian program aplikasi tersebut dari memory.
- **Add Breakpoint** : perintah ini untuk membuka kotak dialog edit breakpoint, yang memungkinkan untuk membuat dan memodifikasi breakpoint dari program aplikasi.



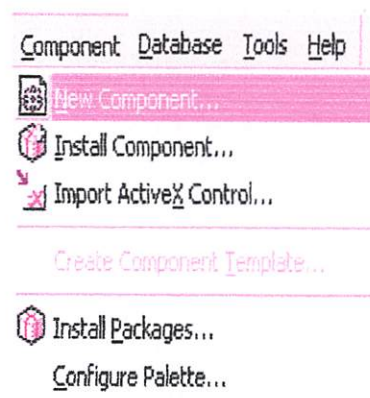
Gambar 2.17 Menu Run

II.3.1.7. Menu Component

Perintah-perintah menu *Component* digunakan untuk membuat sebuah komponen, memasukkan komponen baru, memodifikasi komponen-komponen di dalam component library, mengkonfigurasi komponen-komponen

tersebut khususnya komponen-komponen palette. Menu *Component* tersebut berisi perintah-perintah, diantaranya adalah sebagai berikut :

- **New** : perintah ini untuk membuka kotak dialog New Component. Pada kotak dialog ini dapat memberikan nama class dari komponen yang dibuat, tipe dan nama halaman pada palette yang menunjukkan komponen yang baru dimasukkan tersebut.
- **Install** : perintah ini untuk membuka kotak dialog Install Components. Pada kotak dialog ini dapat memasukkan maupun menghapus komponen-komponen dalam lingkungan pemrograman C++ Builder.
- **Configure Palette** : perintah ini untuk membuka kotak dialog konfigurasi palette. Pada kotak dialog ini dapat mengatur letak dari palette, menambahkan komponen, menghapus, ataupun mengubah nama komponen dari daftar komponen palette yang telah disediakan.

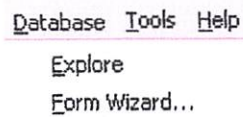


Gambar 2.18. Menu Component

II.3.1.8. Menu Database

Menu *Database* berisi perintah untuk membuat, memodifikasi, dan menampilkan database. Menu database diantaranya terdiri dari :

- **Explorer** : perintah ini sama seperti Windows Explorer sehingga pada database explorer dapat menampilkan hierarki folder-folder database.
- **From Wizard** : perintah ini untuk mengaktifkan pembuatan database wizard yaitu dengan memilih **Database – Form Wizard**. Perintah ini merupakan salah satu fasilitas yang diberikan oleh C++ Builder untuk membuat program aplikasi database secara cepat.

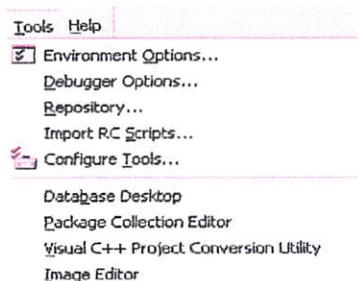


Gambar 2.19. Menu Database

II.3.1.9. Menu Tools

Perintah-perintah yang terdapat dalam menu Tools digunakan untuk mengontrol pengembangan program aplikasi. Perintah-perintah tersebut diantaranya :

- **Environment** : dengan memilih perintah **Tools – Environment**, maka akan ditampilkan kotak dialog Environment Options.
- **Repository** : dengan memilih perintah **Tools – Repository**, maka akan ditampilkan kotak dialog Object Repository. Kotak dialog ini digunakan untuk menambah, mengganti nama dari object di dalam object repository.



Gambar 2.20. Menu Tools

II.3.1.10. Menu Help

Menu *Help* digunakan untuk mengakses file-file help. Pada sistem help ini diberikan informasi tentang lingkungan pemrograman C++ Builder diantaranya :

- **Content** : perintah ini untuk menampilkan kotak dialog help topics. Kotak dialog help topic berisi menu-menu help yang bisa dijadikan petunjuk untuk menggunakan dan menyusun program aplikasi di lingkungan pemrograman C++ Builder.
- **About** : perintah ini untuk menampilkan informasi copyright dan versi dari C++ Builder.



Gambar 2.21. Menu Help

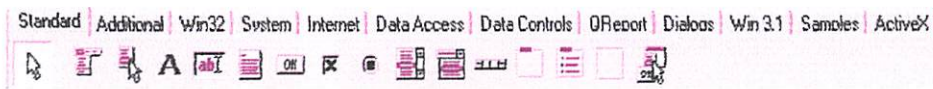
II.3.2. Menu Toolbar

Toolbar C++ Builder memberikan shortcuts untuk perintah-perintah yang berada pada menu utama. Gambar 2.22 menunjukkan default Toolbar pada C++ Builder. Susunan toolbar ini dapat diubah dengan memilih Properties dari menu Context Toolbar.



Gambar 2.22. Menu Toolbar

Pada lingkungan pemrograman C++ Builder ini, menu Toolbar menjadi satu bagian dengan Component Palette. Kedua bagian ini dipisahkan dengan garis vertical. Sedangkan component standard yang biasa digunakan dalam pemrograman C++ Builder ini adalah MainMenu, PopupMenu, Label, Edit, Memo, Button, CheckBox, RadioButton, ListBox, ScrollBar, GroupBox, RadioGroup, dan Panel.



Gambar 2.23. Komponen Standard

Penjelasan dari beberapa component standard tersebut adalah :

- MainMenu  adalah komponen yang menggabungkan menu-menu bar.
- PopupMenu  berlaku seperti MainMenu tetapi komponen ini aktif ketika pengguna melakukan klik kanan pada tempat tertentu untuk menampilkan menu-menu kontrol.
- Label  adalah komponen yang bertugas untuk menampilkan teks pada form.
- Edit  adalah komponen yang digunakan untuk menempatkan teks yang diisi oleh pengguna sebagai sebuah masukan untuk mengontrol program. Komponen edit ini hanya menerima teks dalam bentuk single line.
- Memo  adalah komponen yang digunakan untuk menempatkan teks yang berbentuk multi line, komponen ini biasa digunakan untuk membuat sebuah form yang berisi informasi.

- Button  adalah komponen yang digunakan sebagai tombol.
- CheckBox  adalah komponen yang digunakan untuk mengontrol program.
- RadioButton  adalah komponen yang digunakan untuk mengontrol program. Komponen ini berbentuk kelompok yang mempunyai opsi-opsi yang dapat menerima perintah masukan.
- GroupBox  adalah komponen yang digunakan untuk mengelompokkan kontrol yang saling berhubungan misalnya RadioButton, sehingga komponen-komponen tersebut akan menjadi satu kesatuan yang tidak dapat dipisahkan.

BAB III

PELAKSANAAN PENELITIAN

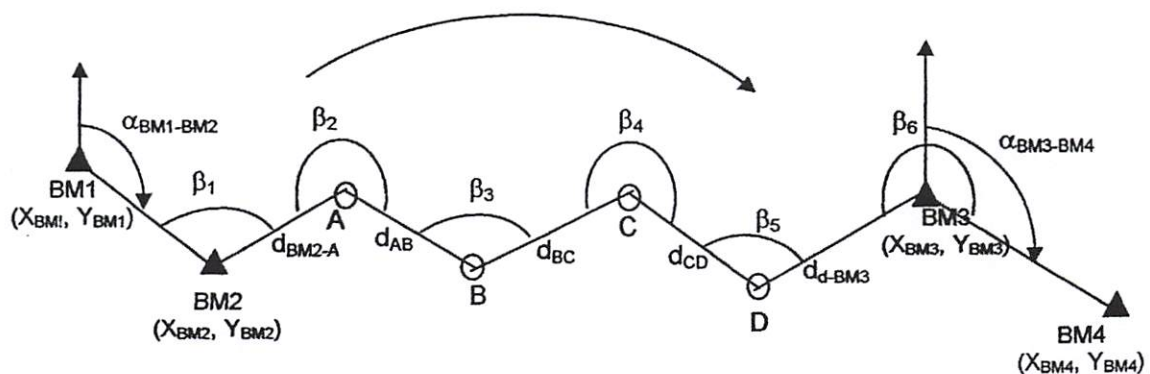


III.1. Data Penelitian

Dalam penelitian ini tidak dilakukan pengukuran secara langsung untuk memperoleh data-data poligon, melainkan hanya mengambil data yang sudah ada. Adapun data yang digunakan untuk input pada program hitung perataan parameter poligon ini adalah :

- Data pengukuran poligon meliputi sudut horisontal, jarak horisontal, azimuth, dan koordinat awal yang sudah dihitung secara manual sebagai data uji coba paket program yang akan dibuat. Bentuk poligon tersebut diantaranya adalah :

1. Poligon Terbuka Terikat Sempurna



Gambar 3.1. Poligon Terbuka Terikat Sempurna

Keterangan gambar 3.1 adalah :

BM_1, BM_2, BM_3, BM_4 : Titik ikat/tetap

A, B, C, D : Titik-titik yang akan dihitung koordinatnya

$d_{BM2-A}, d_{AB}, d_{BC}, d_{CD}, d_{D-BM3}$: Jarak horisontal

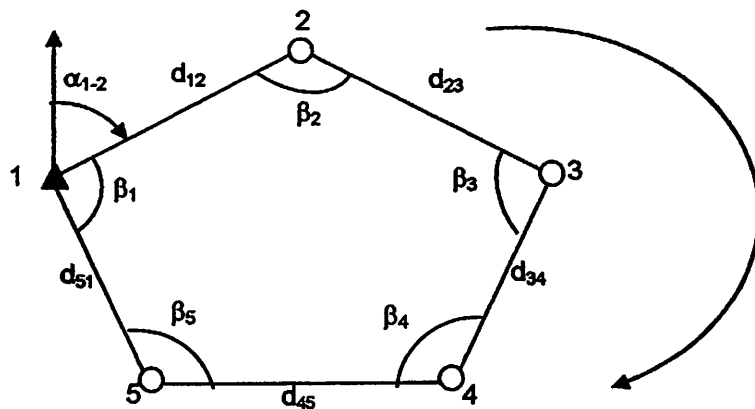
$\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6$: Sudut horisotnal

$\alpha_{BM1-BM2}, \alpha_{BM3-BM4}$: Azimuth awal dan akhir poligon

2. Poligon Tertutup

Untuk data poligon tertutup dibedakan menjadi dua yaitu untuk data poligon tertutup dengan sudut dalam dan poligon tertutup dengan sudut luar.

a. Poligon Tertutup dengan Sudut Dalam

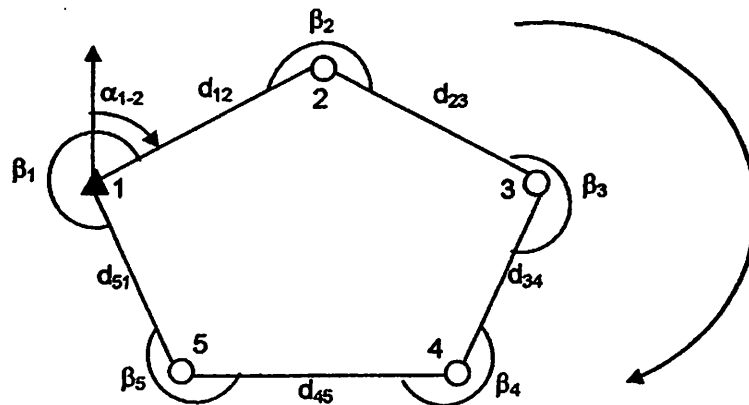


Gambar 3.2. Poligon tertutup dengan sudut dalam

Keterangan gambar 3.2 adalah :

- Titik 1 : Titik ikat/tetap
 Titik 2, 3, 4, 5 : Titik-titik poligon
 $d_{12}, d_{23}, d_{34}, d_{45}, d_{51}$: Jarak horisontal
 $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$: Sudut horizontal (sudut dalam)

b. Poligon Tertutup dengan Sudut Luar



Gambar 3.3. Poligon tertutup dengan sudut luar

Keterangan gambar 3.3 adalah :

- Titik 1 : Titik ikat/tetap
 Titik 2, 3, 4, 5 : Titik-titik poligon
 $d_{12}, d_{23}, d_{34}, d_{45}, d_{51}$: Jarak horisontal
 $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$: Sudut horizontal (sudut luar)

III.2. Peralatan Penelitian

Adapun peralatan yang digunakan dalam pelaksanaan penelitian ini adalah sebagai berikut :

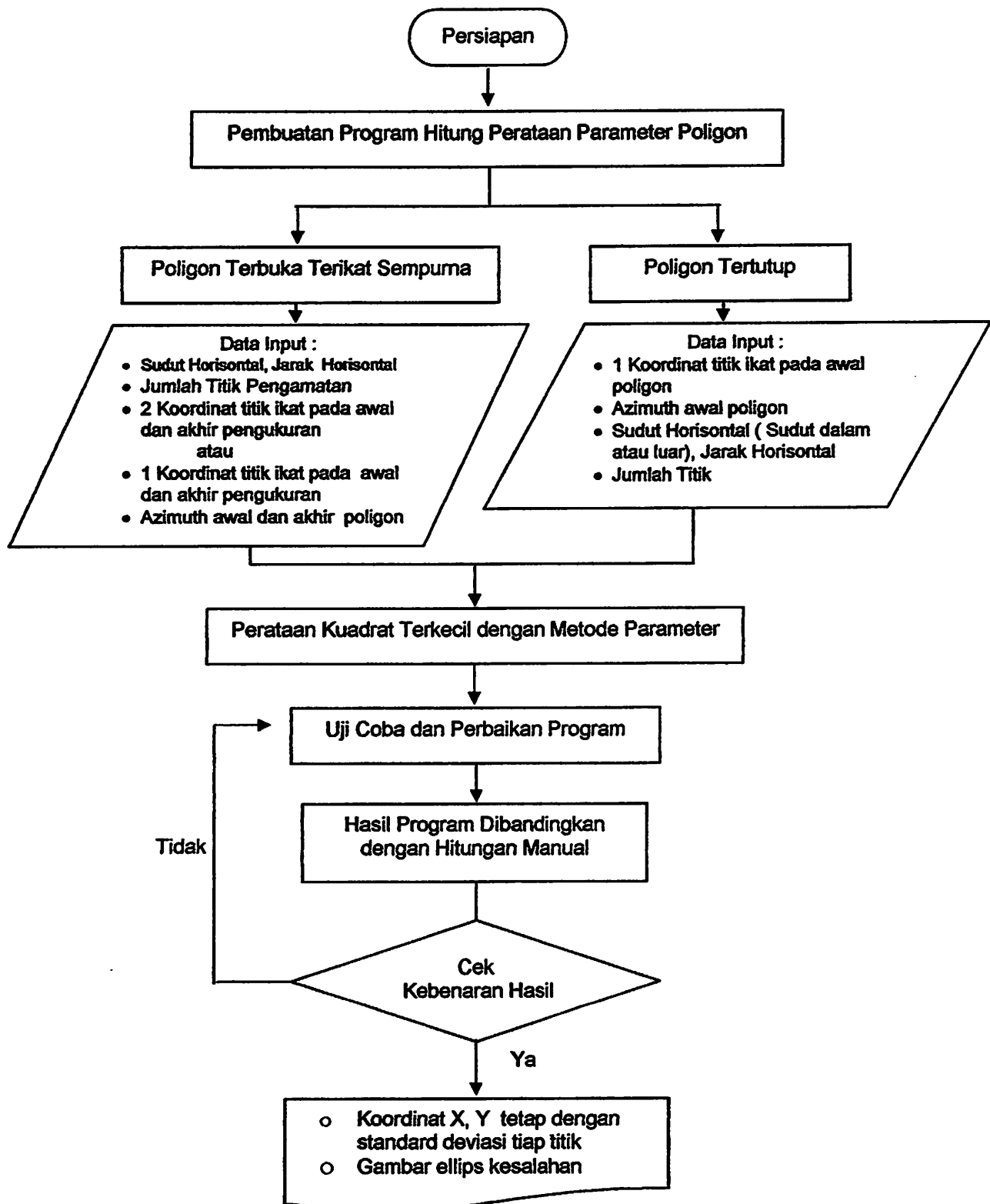
➤ **Perangkat Keras (Hardware) dengan spesifikasi :**

1. Processor Intel Pentium IV
2. Hard disk 40 GB
3. Memory 256 MB
4. Monitor
5. Keyboard
6. Mouse
7. Printer

➤ **Perangkat Lunak (Software)**

1. MS Word
2. MS Excell
3. Borland C++ Builder

III.3. Diagram Alir Penelitian



Gambar 3.4. Diagram Alir Penelitian

Penjelasan Diagram Alir Penelitian :**1. Persiapan**

Dalam tahap persiapan penelitian dilakukan persiapan-persiapan sebelum penelitian yaitu persiapan materi/teori dengan pembacaan literatur yang berhubungan dengan topik penelitian. Selain itu juga dilakukan pengumpulan data-data yang diperlukan dalam penelitian baik data poligon terbuka terikat sempurna maupun poligon tertutup, untuk selanjutnya akan diproses dalam penelitian ini.

2. Pembuatan Program Hitung Perataan Parameter Poligon

Pada tahap ini dilakukan pembuatan program aplikasi perhitungan poligon dengan menggunakan metode perataan parameter yang terdiri dari program hitung perataan parameter poligon terbuka terikat sempurna dan program hitung perataan parameter poligon tertutup

3. Data input (data masukan)

Data input (data masukan) yang akan digunakan dalam program hitung perataan parameter poligon terdiri dari data input untuk poligon terbuka terikat sempurna dan data input untuk poligon tertutup. Data input untuk Poligon Terbuka Terikat Sempurna adalah dua koordinat titik ikat pada awal pengukuran, dua koordinat titik ikat pada akhir pengukuran, sudut horisontal, jarak horizontal, jumlah titik pengamatan (titik yang dicari). Apabila diketahui 1 koordinat awal dan akhir poligon, maka data yang harus diinputkan adalah azimuth awal dan akhir poligon, sudut horisontal, jarak horisotnal, dan jumlah titik pengamatan (titik yang dicari). Sedangkan data input untuk Poligon

Tertutup adalah 1 koordinat titik ikat pada awal poligon, azimuth awal, sudut horisontal (sudut dalam atau sudut luar), jarak horisontal, dan jumlah titik.

4. Perataan Kuadrat Terkecil dengan Metode Parameter

Proses ini digunakan untuk melakukan hitung perataan parameter pada poligon baik untuk poligon terbuka terikat sempurna maupun poligon tertutup dari data input yang ada.

5. Uji coba dan perbaikan program

Tahap ini dilakukan untuk mengetahui apakah program hitung perataan parameter tersebut *Running* atau tidak, maka perlu dilakukan uji coba dengan menggunakan data-data yang sudah valid dan apabila diperlukan dilakukan perbaikan.

6. Hasil program dibandingkan dengan hitungan manual

Apabila program hitung perataan parameter poligon sudah *Running*, maka perlu dilakukan uji hasil program tersebut dengan membandingkan hasil hitung program dengan hitungan manual (dalam hal ini menggunakan *Microsoft Excel*)

7. Cek kebenaran hasil

Cek kebenaran hasil merupakan pengecekan uji coba hasil hitungan dengan menggunakan program hitung perataan parameter poligon. Apabila hasil hitungan telah sesuai dengan hasil hitungan manual, maka program tersebut sudah benar. Akan tetapi apabila selisih antara hasil hitungan dengan program hitung perataan parameter poligon dibandingkan dengan hitungan

manual berbeda jauh, maka perlu dilakukan perbaikan kembali program hitung perataan parameter poligon tersebut.

8. Hasil Program

Hasil dari program hitung perataan parameter untuk poligon (poligon terbuka terikat sempurna dan poligon tertutup) adalah koordinat X, Y tetap dengan standard deviasi tiap titik dan gambar ellips kesalahan.

III.4. Pembuatan Program

Dalam pembuatan suatu paket program, ada tahapan-tahapan yang harus dilakukan diantaranya adalah :

1. Rancangan Program

Pada tahap ini dilakukan analisa terhadap suatu permasalahan yang akan dibuat paket programnya. Dari analisa tersebut akan dapat dibuat rancangan programnya mulai dari menentukan data *input* (data yang dimasukkan / akan diproses), bagaimana memprosesnya (proses), dan apa hasil dari program tersebut (*output*). Hasil keseluruhan dari tahap ini adalah rancangan program yang berupa diagram alir (*flowchart*)

2. Rancangan Tampilan

Tahap rancangan tampilan adalah penerapan dari tahap pertama (tahap rancangan program) yaitu bagaimana merancang tampilan. Hasil dari tahapan ini adalah tampilan program berupa *form-form*.

3. Pemrograman

Tahap pemrograman sering disebut juga sebagai tahapan implementasi perangkat lunak (*coding*). Dengan kata lain, pada tahap ini dilakukan implementasi hasil analisa ke dalam baris-baris kode program yang dapat dimengerti oleh komputer.

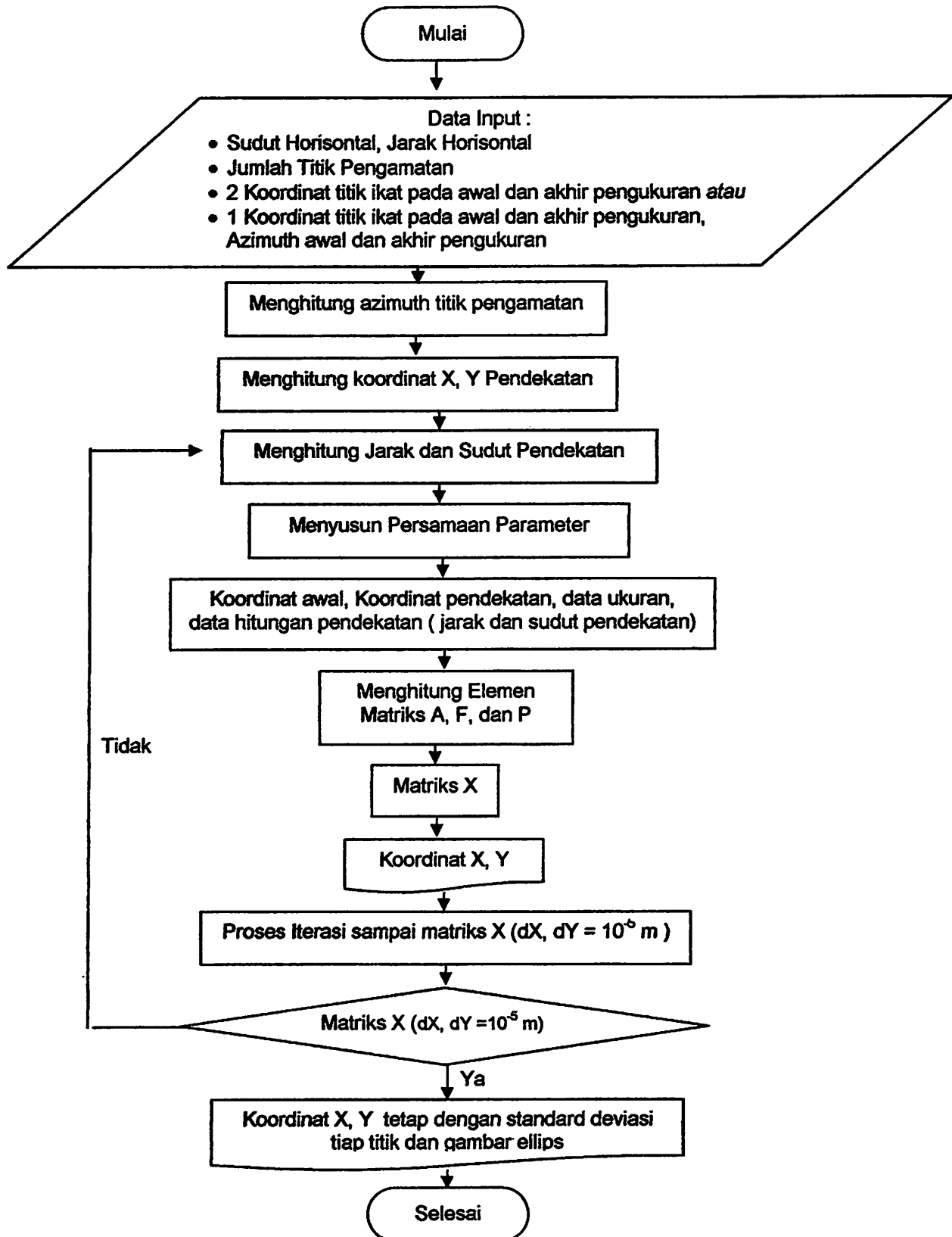
4. Pengujian

Setelah suatu paket program selesai dibuat, maka perlu dilakukan pengujian (uji coba) paket program tersebut. Pada awalnya sebaiknya menggunakan data yang sederhana baru kemudian data yang kompleks.

III.4.1. Rancangan Program

Pembuatan rancangan program ini dimaksudkan untuk memudahkan pada saat pembuatan program pada Borland C++ Builder. Dalam merancang program dilakukan dengan cara membuat flowchart-flowchart yang akan diimplementasikan ke dalam bahasa pemrograman. Paket program yang akan dibuat adalah paket program hitung perataan parameter untuk poligon yaitu untuk poligon terbuka terikat sempurna dan poligon tertutup.

Diagram Alir Poligon Terbuka Terikat Sempurna



Gambar 3.5. Diagram Alir Poligon Terbuka Terikat Sempurna

Penjelasan Diagram Alir Poligon Terbuka Terikat Sempurna

Dalam proses hitung perataan parameter poligon terbuka terikat sempurna, langkah-langkah dalam perhitungan tersebut adalah :

1. Input Data

Input data pada Poligon Terbuka Terikat Sempurna dibagi dua yaitu diketahui 2 koordinat awal dan akhir poligon dan diketahui 1 koordinat awal dan akhir poligon. Apabila diketahui 2 koordinat awal dan akhir, maka data inputnya adalah sudut horisontal, jarak horisontal, dan jumlah titik pengamatan (jumlah koordinat titik yang dicari). Sedangkan apabila diketahui 1 koordinat awal dan akhir poligon, maka data yang harus dimasukkan adalah data azimuth awal dan akhir poligon, sudut horisontal, jarak horizontal, dan jumlah titik pengamatan.

2. Menghitung Azimuth Titik

Azimuth titik-titik pengamatan dapat dihitung setelah didapatkan nilai azimuth awal dari perhitungan dua koordinat titik awal pengukuran. Bentuk persamaannya seperti persamaan (2.17).

3. Menghitung Koordinat X, Y Pendekatan

Koordinat X, Y Pendekatan dihitung dari koordinat awal, jarak dan azimuth titik dengan rumus : $X^o = X_{awal} + d_{awal} \cdot \sin \alpha_{awal}$ untuk koordinat X dan $Y^o = Y_{awal} + d_{awal} \cdot \cos \alpha_{awal}$ untuk koordinat Y. Bentuk persamaannya seperti persamaan (2.18).

4. Menghitung Jarak dan Sudut Pendekatan

Setelah didapatkan koordinat X, Y pendekatan, maka jarak dan sudut pendekatan dapat dihitung. Untuk bentuk persamaannya seperti persamaan (2.19) dan (2.20).

5. Menyusun Persamaan Parameter

Tahap selanjutnya adalah menyusun persamaan parameter dari data koordinat awal, koordinat pendekatan, data ukuran (jarak dan sudut ukuran), serta data hitungan pendekatan (jarak dan sudut pendekatan)

6. Menghitung Elemen Matriks A, F, dan P

Setelah disusun persamaan parameternya kemudian dilakukan perhitungan elemen matriks disain (A) yang didapatkan dari linierisasi jarak dan sudut, seperti persamaan (2.210 dan (2.22). Sedangkan untuk matriks konstanta (F) didapatkan dari selisih data ukuran dengan data hitungan pendekatan seperti persamaan (2.23). Matriks bobot (P) dapat dihitung maupun dianggap 1. Matriks A, F, dan P digunakan untuk menghitung nilai matriks parameter (X).

7. Menghitung Matriks Parameter (Matriks X)

Untuk menghitung nilai matriks parameter (X) apabila menggunakan matriks bobot (P), maka menggunakan rumus $X = (A^T P A)^{-1} (A^T P F)$. Sedangkan apabila tidak menggunakan matriks bobot (P), maka dengan rumus $X = (A^T A)^{-1} (A^T F)$. Bentuk persamaannya seperti pada persamaan (2.26)

8. Koordinat X, Y

Nilai koordinat X,Y tiap titik pengamatan didapatkan dari penjumlahan antara nilai koordinat X,Y pendekatan dengan nilai matriks X (dX , dY). Untuk bentuk persamaannya seperti pada persamaan (2.27)

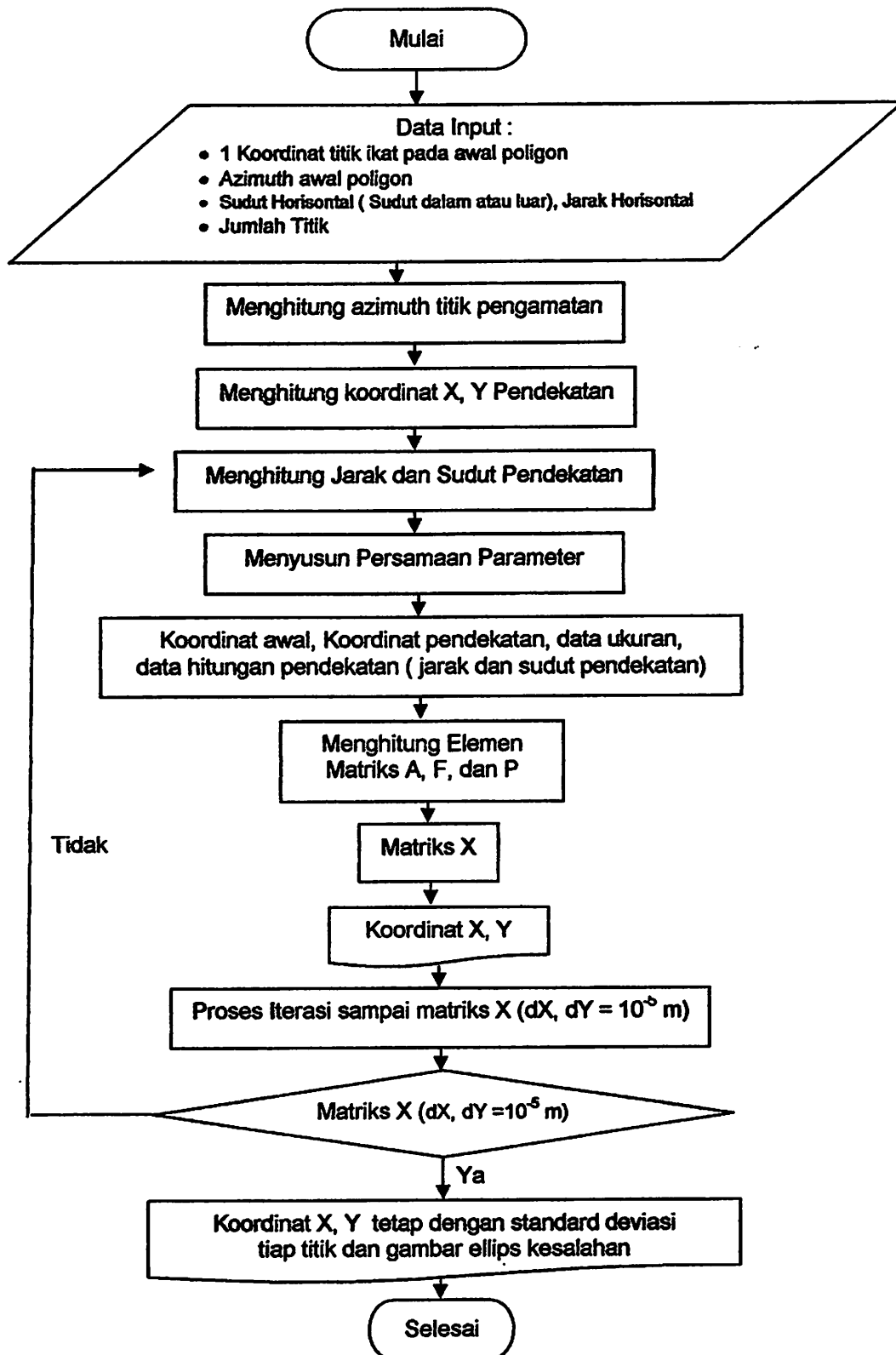
9. Proses Iterasi

Apabila nilai matriks X (dX , dY) belum sama atau mendekati 10^{-5} m, maka dilakukan proses iterasi (pengulangan) sampai didapatkan nilai dX dan $dY = 10^{-5}$ m.

10. Hasil Program

Hasil dari Program Hitung Perataan Parameter Poligon Terbuka Terikat Sempurna ini adalah nilai koordinat X, Y tetap yang merupakan penjumlahan antara koordinat X,Y dengan nilai matriks X. serta nilai standard deviasi tiap titik. Selain itu juga dilengkapi dengan gambar ellips kesalahan.

Diagram Alir Poligon Tertutup



Gambar 3.6. Diagram Alir Poligon Tertutup

Penjelasan Diagram Alir Poligon Tertutup

Dalam proses hitung perataan parameter poligon terbuka terikat sempurna, langkah-langkah dalam perhitungan tersebut adalah :

1. Input Data

Pada tahap ini dimasukkan data-data yang diperlukan yaitu satu koordinat titik ikat pada awal pengukuran, azimuth awal, sudut horisontal (sudut dalam atau sudut luar), jarak horisontal, dan jumlah titik (jumlah titik koordinat). Untuk input azimuth awal dari titik awal ke titik berikutnya searah jarum jam

2. Menghitung Azimuth Titik

Azimuth titik-titik pengamatan dapat dihitung mulai dari nilai azimuth awal ditambah sudut horisontal. Bentuk persamaannya seperti pada bentuk persamaan (2.32)

3. Menghitung Koordinat X, Y Pendekatan

Koordinat X, Y Pendekatan dihitung dari koordinat awal, jarak dan azimuth titik dengan rumus : $X^o = X_{awal} + d_{awal} \cdot \sin \alpha_{awal}$ untuk koordinat X dan $Y^o = Y_{awal} + d_{awal} \cdot \cos \alpha_{awal}$ untuk koordinat Y. Untuk bentuk persamaannya seperti pada persamaan (2.33)

4. Menghitung Jarak dan Sudut Pendekatan

Setelah didapatkan koordinat X, Y pendekatan, maka jarak dan sudut pendekatan (baik sudut dalam maupun sudut luar) dapat dihitung. Bentuk persamaannya seperti pada persamaan (2.34), (2.35), dan (2.36)

5. Menyusun Persamaan Parameter

Tahap selanjutnya adalah menyusun persamaan parameter yang didapatkan dari data koordinat awal, koordinat pendekatan, data ukuran (jarak dan sudut ukuran), serta data hitungan pendekatan (jarak dan sudut pendekatan). Proses perhitungan baik untuk poligon tertutup dengan sudut dalam dengan sudut luar adalah sama (searah dengan jarum jam)

6. Menghitung Elemen Matriks A, F, dan P

Setelah disusun persamaan parameternya kemudian dilakukan perhitungan elemen matriks disain (A) yang didapatkan dari linierisasi jarak dan sudut, seperti pada persamaan (2.37) dan (2.38). Sedangkan matriks konstanta (F) didapatkan dari selisih data ukuran dengan data hitungan pendekatan. Bentuk persamaannya seperti pada persamaan (2.40). Matriks bobot (P) dapat dihitung maupun dianggap 1. Matriks A, F, dan P digunakan untuk menghitung nilai matriks parameter (X)

7. Menghitung Matriks Parameter (Matriks X)

Untuk menghitung nilai matriks parameter (X) apabila menggunakan matriks bobot (P), maka menggunakan rumus $X = (A^T P A)^{-1} (A^T P F)$. Sedangkan apabila tidak menggunakan matriks bobot (P), maka dengan rumus $X = (A^T A)^{-1} (A^T F)$. Bentuk persamaannya seperti persamaan (2.43)

8. Koordinat X, Y

Nilai koordinat X,Y tiap titik pengamatan didapatkan dari penjumlahan antara nilai koordinat X,Y pendekatan dengan nilai matriks X (dX , dY). Bentuk persamaannya seperti pada persamaan (2.44)

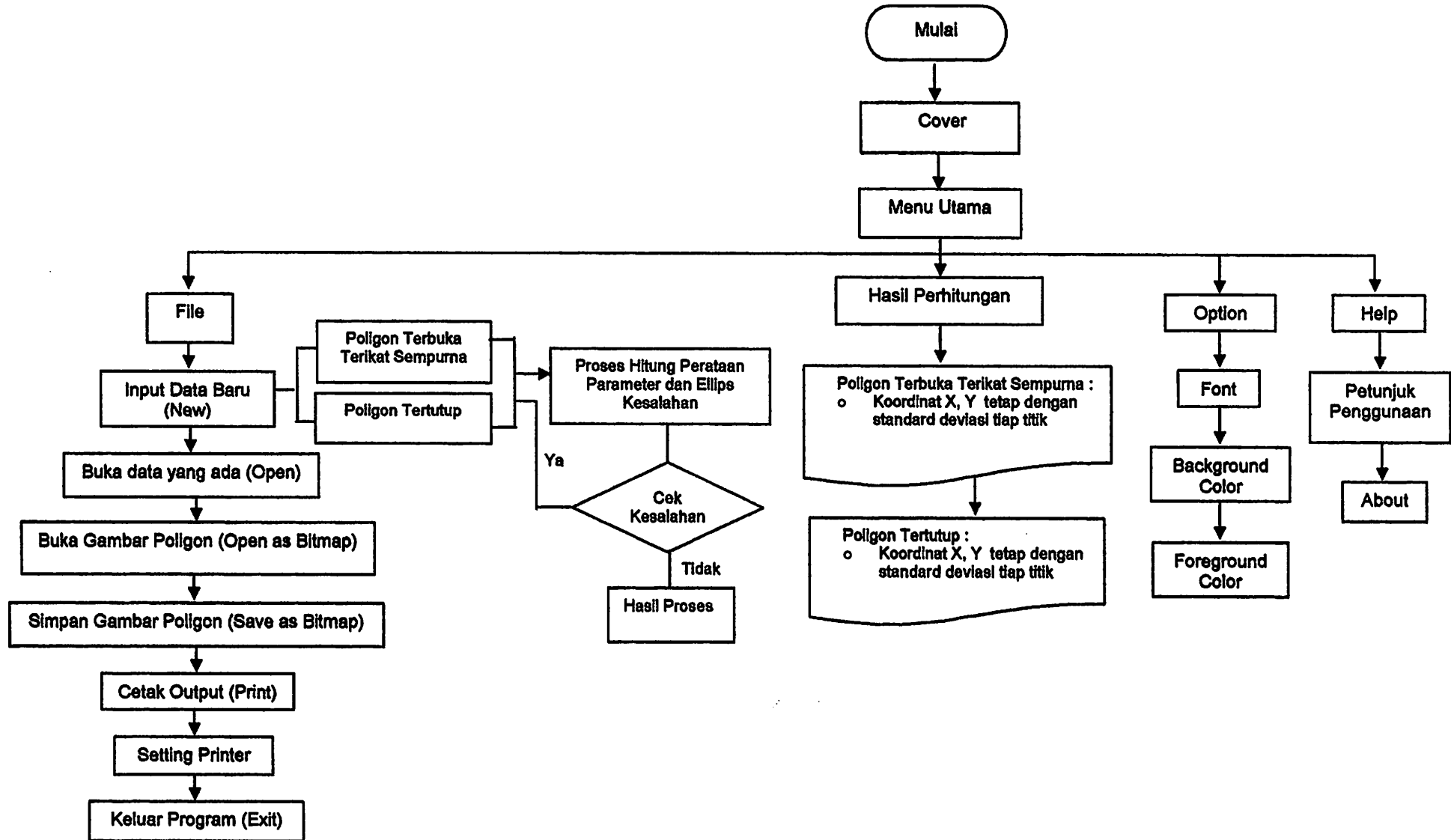
9. Proses Iterasi

Apabila nilai matriks X (dX , dY) belum sama atau mendekati 10^{-5} m, maka dilakukan proses iterasi (pengulangan) sampai didapatkan nilai dX dan $dY = 10^{-5}$ m.

10. Hasil Program

Hasil dari Program Hitung Perataan Parameter Poligon Tertutup ini adalah nilai koordinat X, Y tetap yang merupakan penjumlahan antara koordinat X, Y dengan nilai matriks X. serta standard deviasi setiap titik. Selain itu juga dilengkapi dengan gambar ellips kesalahan.

Diagram Alir Paket Program



Gambar 3.7. Diagram Alir Paket Program

Penjelasan Diagram Alir Paket Program

1. Tampilan Depan (*Cover*)

Tampilan depan (*cover*) ini adalah suatu tampilan form sebagai pembuka dari program penghitungan poligon dengan metode perataan parameter berisi pilihan apakah program akan dilanjutkan (*next*) atau keluar dari program (*exit*)

2. Menu Utama

Menu utama dari paket program ini terdiri dari Menu File, Menu Hasil Perhitungan, Menu Option, dan Menu Help.

3. File

Menu file terdiri dari new, open, open as bitmap, save as bitmap, print, setting printer, dan exit

- a. New adalah menu untuk membuat input data baru, dipilih sesuai dengan tipe file yang akan dibuka. Apabila yang dipilih adalah poligon terbuka maka akan masuk form input data poligon terbuka terikat sempurna. Kemudian dimasukkan data-datanya untuk proses hitung dan hasil penyimpanannya diakhiri dengan ekstensi *.PTA. Untuk poligon tertutup juga sama tetapi hasil penyimpanannya diakhiri dengan ekstensi *.PTB;
- b. Open adalah menu membuka file data poligon yang ada atau file yang sudah disimpan yaitu *.PTA untuk yang menggunakan poligon terbuka terikat sempurna dan *.PTB untuk poligon yang tertutup;
- c. Open as bitmap adalah menu untuk membuka file bitmap yang berisi gambar poligon yang sesuai dengan datanya;

- d. **Save as Bitmap** adalah menu untuk menyimpan gambar poligon yang ada ke dalam bidang gambar yang ada;
- e. **Print** adalah menu untuk mencetak hasil perhitungan;
- f. **Setting Printer** adalah menu untuk mensetting printer mulai dari jenis printer yang digunakan, ukuran kertas, orientasi bentuk kertas (*portrait* atau *landscape*), dan lain-lain sebelum hasil cetakan siap untuk dicetak;
- g. **Exit** adalah menu untuk keluar dari program

4. Hasil Perhitungan

Menu hasil perhitungan terdiri dari hasil perhitungan poligon terbuka terikat sempurna maupun poligon tertutup.

- a. Hasil perhitungan poligon terbuka terikat sempurna berisi hasil perhitungan dari data yang diinputkan;
- b. Hasil perhitungan poligon tertutup terdiri dari hasil perhitungan poligon tertutup menggunakan sudut dalam dan sudut luar. Hasilnya berupa data hasil perhitungan.

5. Option

Menu option (pilihan) ini terdiri dari menu font, background color, dan foreground color.

- a. **Font** adalah menu untuk memilih jenis tulisan yang digunakan;
- b. **Background Color** adalah menu untuk memilih warna background/latar belakang dari program;
- c. **Foreground Color** adalah menu untuk memilih warna pena / tulisan

6. Help

Menu Help terdiri dari petunjuk penggunaan program penghitungan poligon dengan metode perataan parameter dan about program.

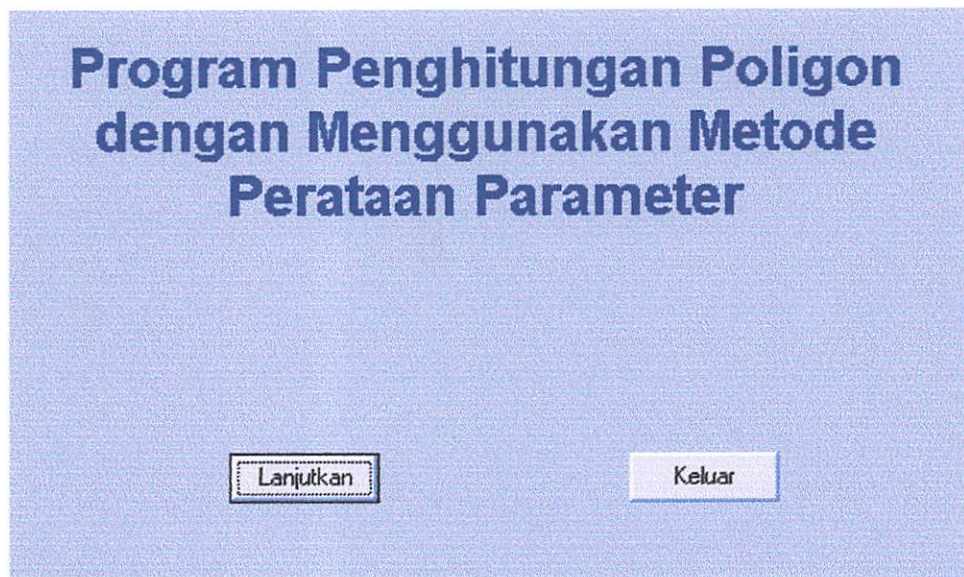
- a. Petunjuk penggunaan adalah menu yang berisi tentang bagaimana cara menggunakan program ini;
- b. About adalah menu yang berisi tentang program, keterangan nama program, versi, dan lain-lain

III.4.2. Rancangan Tampilan (*Design Interfaces*)

Rancangan tampilan berfungsi untuk mengimplementasikan rancangan program dalam bentuk tampilan, sehingga lebih mudah dalam penulisan *source code*. Berikut ini adalah contoh bentuk tampilan menu dari program hitung perataan parameter untuk poligon :

1. Form

Form adalah tempat utama yang digunakan untuk interaksi antara pengguna dengan program. Form juga merupakan tempat untuk meletakkan kontrol-kontrol lain seperti menu, toolbar, button, dan lain-lain. Pada program ini, sebelum masuk ke form menu utama, terlebih dahulu ke form depan (*cover*).

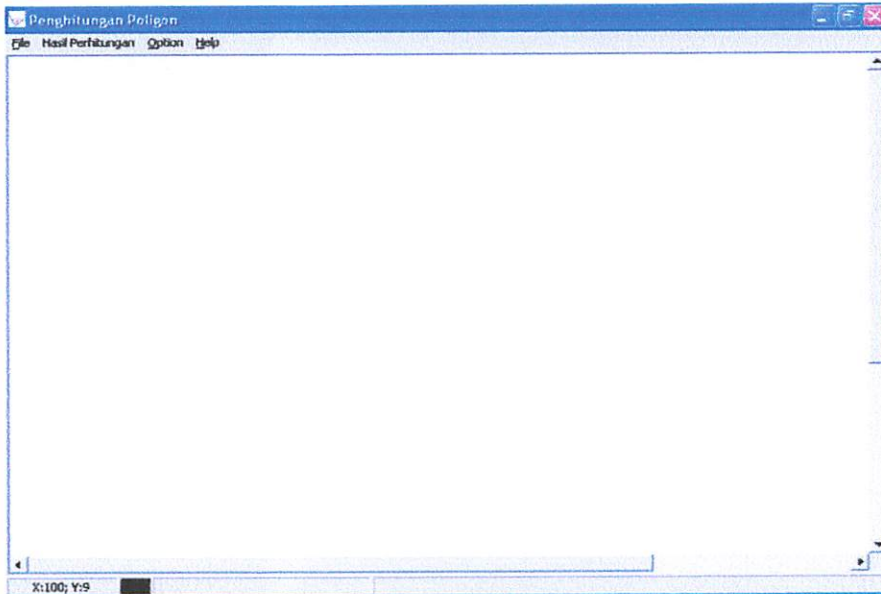
Form Depan (Cover)

Gambar 3.8. Form Depan / Cover

Komponen	Property	
	Type	Pengaturan
Label	Name	Label1
	Caption	Program Penghitungan Poligon dengan Menggunakan Perataan Parameter
	Height	97
	Left	24
	Top	16
	Width	75
Button	Name	Button1
	Caption	Lanjutkan
Button	Name	Button2
	Caption	Keluar

Form Menu Utama

Pada form menu utama terdapat empat menu yaitu menu file, hasil perhitungan, option, dan help.



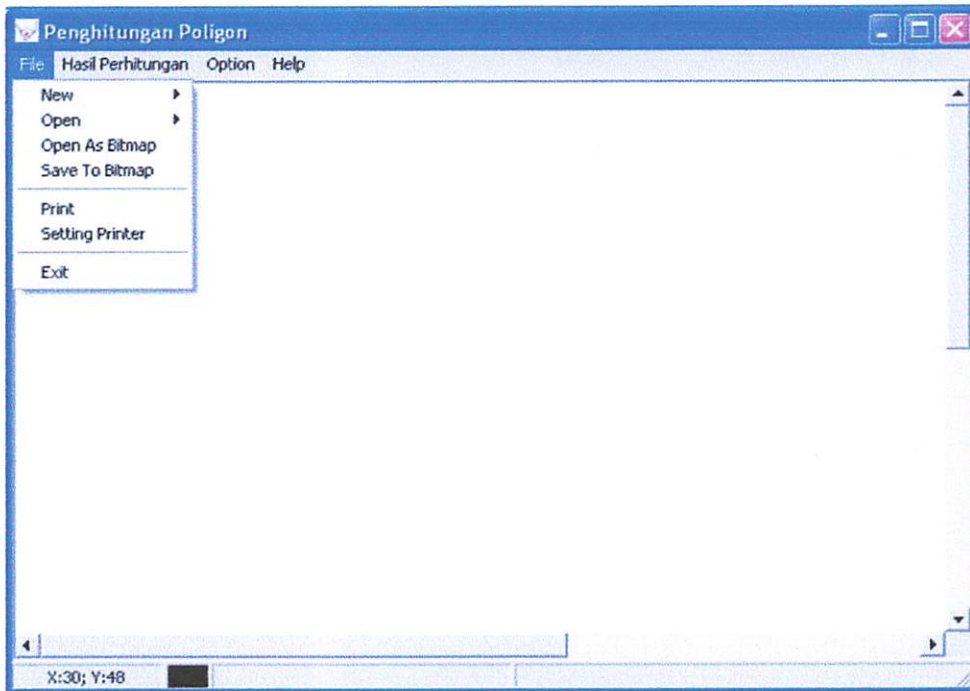
Gambar 3.9. Form Menu Utama

Komponen	Property	
	Type	Pengaturan
MainForm	Name	MainForm
	Caption	Penghitungan Poligon
	Menu	MainMenu1
MenuItem	Name	File1
	Caption	&File
MenuItem	Name	Hasil Perhitungan1
	Caption	Hasil Perhitungan
MenuItem	Name	Option1
	Caption	&Option
MenuItem	Name	Help1
	Caption	&Help

2. Menu - menu

a. Menu File

Menu file terdiri dari new, open, open as bitmap, save as bitmap, print, setting printer, dan exit



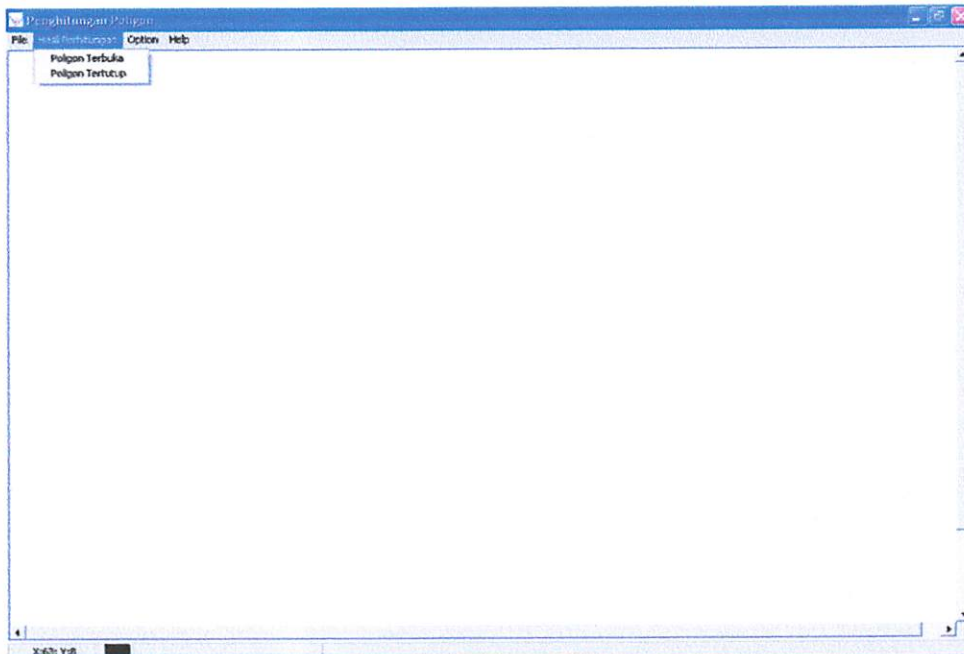
Gambar 3.10. Form Menu File

Komponen	Property	
	Type	Pengaturan
MenuItem	Name	File1
	Caption	&File
MenuItem	Name	New1
	Caption	New
MenuItem	Name	Open1
	Caption	Open
MenuItem	Name	OpenAsBitmap1
	Caption	Open as Bitmap

MenuItem	Name	Save1
	Caption	Save to Bitmap
MenuItem	Name	Print1
	Caption	Print
MenuItem	Name	SettingPrinter1
	Caption	Setting Printer
MenuItem	Name	Exit1
	Caption	&Exit

b. Menu Hasil Perhitungan

Menu hasil perhitungan terdiri dari hasil perhitungan poligon terbuka terikat sempurna dan poligon tertutup baik menggunakan sudut dalam maupun sudut luar.

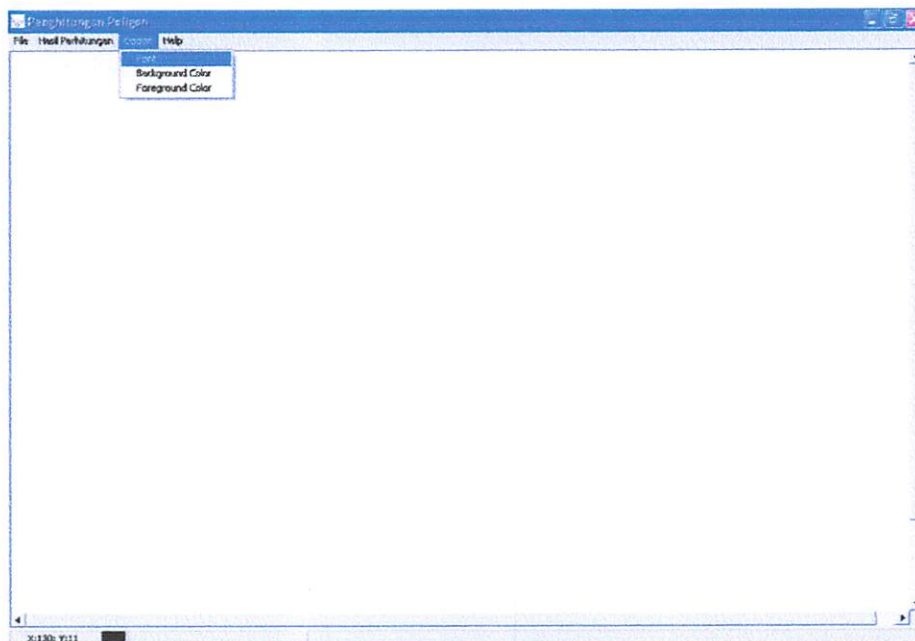


Gambar 3.11. Form Menu Hasil Perhitungan

Komponen	Property	
	Type	Pengaturan
Hasil Perhitungan	Name	Hasil Perhitungan1
	Caption	Hasil Perhitungan
Poligon Terbuka Terikat Sempurna	Name	PoligonTerbuka1
	Caption	Poligon Terbuka Terikat Sempurna
Poligon Tertutup	Name	PoligonTertutup1
	Caption	Poligon Tertutup

c. Menu Option

Menu option terdiri dari font (untuk memilih jenis tulisan), background color (untuk memilih warna latar belakang program), dan foreground color (untuk memilih warna tulisan)

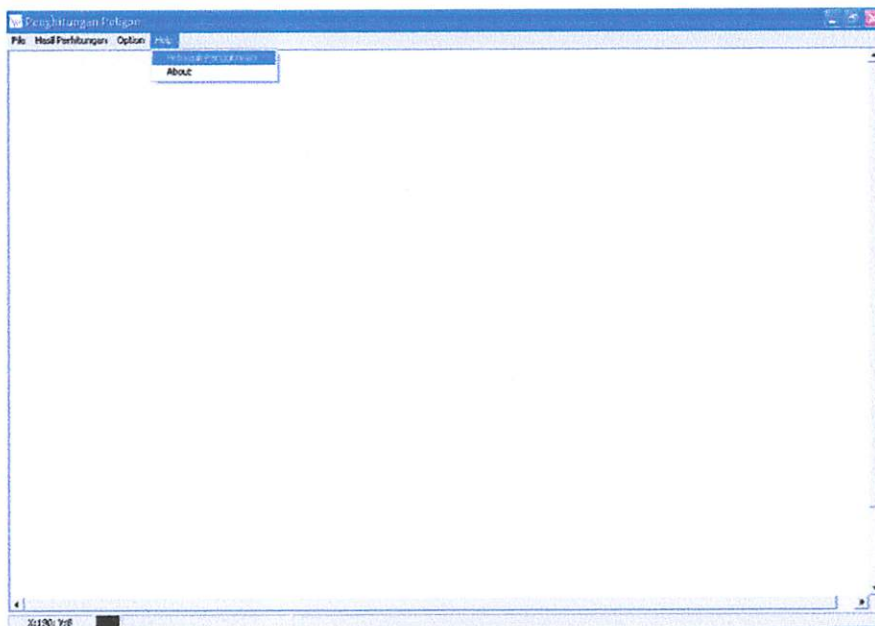


Gambar 3.12. Form Menu Option

Komponen	Property	
	Type	Pengaturan
MenuItem	Name	Option1
	Caption	&Option
MenuItem	Name	Font1
	Caption	Font
MenuItem	Name	BackColor1
	Caption	Background Color
MenuItem	Name	ForeColor1
	Caption	Foreground Color

d. Menu Help

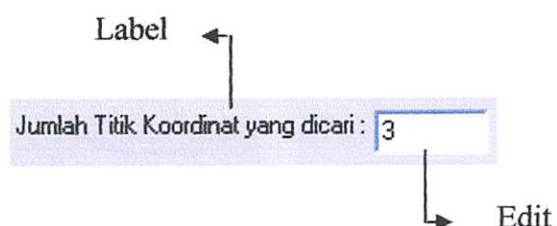
Menu help berisi tentang petunjuk penggunaan program dan about program.



Gambar 3.13. Form Menu Help

Komponen	Property	
	Type	Pengaturan
MenuItem	Name	Help1
	Caption	&Help1
MenuItem	Name	PetunjukPenggunaan1
	Caption	Petunjuk Penggunaan
MenuItem	Name	About1
	Caption	&About

3. Label dan Edit



Gambar 3.14. Label dan edit

Label berfungsi untuk memberikan keterangan pada pengguna program, oleh karena itu sangat berguna dalam tampilan. Sedangkan **edit** berfungsi untuk menginputkan data atau tampilan data yang berbentuk kotak.

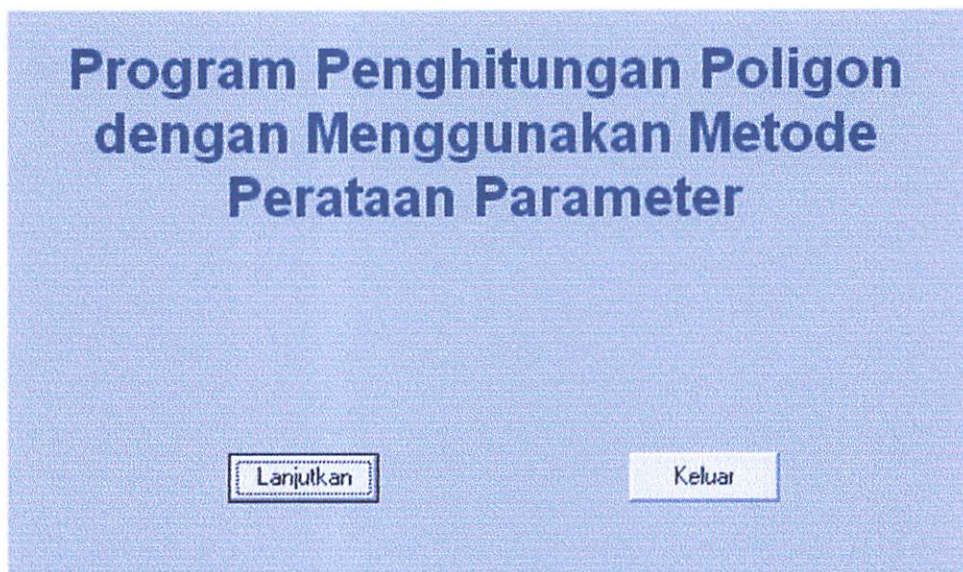
BAB IV

HASIL DAN ANALISA

IV.1. Tampilan Program Hitung Perataan Parameter Poligon

Pada penelitian ini akan dihasilkan suatu program penghitungan poligon dengan menggunakan metode perataan parameter. Program ini bertujuan untuk memudahkan pengguna di dalam melakukan proses hitung perataan parameter untuk poligon. Oleh karena itu, bentuk tampilan program ini dibuat sedemikian rupa untuk memudahkan pengguna.

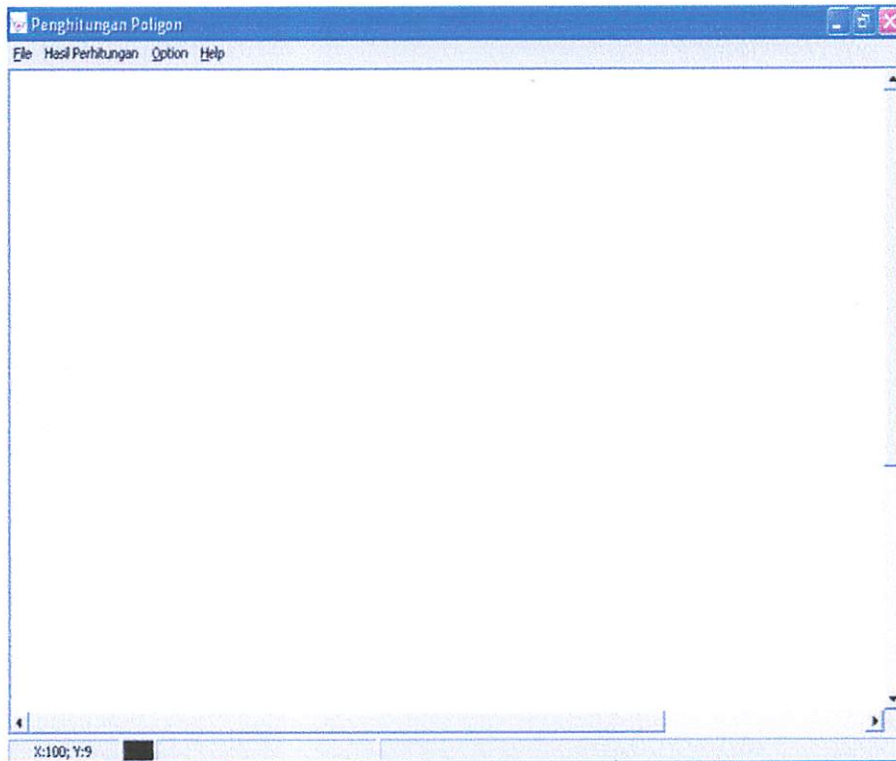
Pada waktu program dijalankan, pertama kali akan muncul Form Depan (*Cover*) yang merupakan pilihan apakah pengguna akan melanjutkan ke Form Menu Utama atau keluar dari program. Adapun bentuk tampilan dari Form Depan (*Cover*) seperti pada gambar 4.1 :



Gambar 4.1 Form Depan

Form Utama Program Penghitungan Poligon dengan Menggunakan Perataan Parameter ini terdiri dari beberapa Menu Utama yaitu Menu File, Hasil Perhitungan, Option, dan Help.

Bentuk tampilannya seperti pada gambar 4.2 :

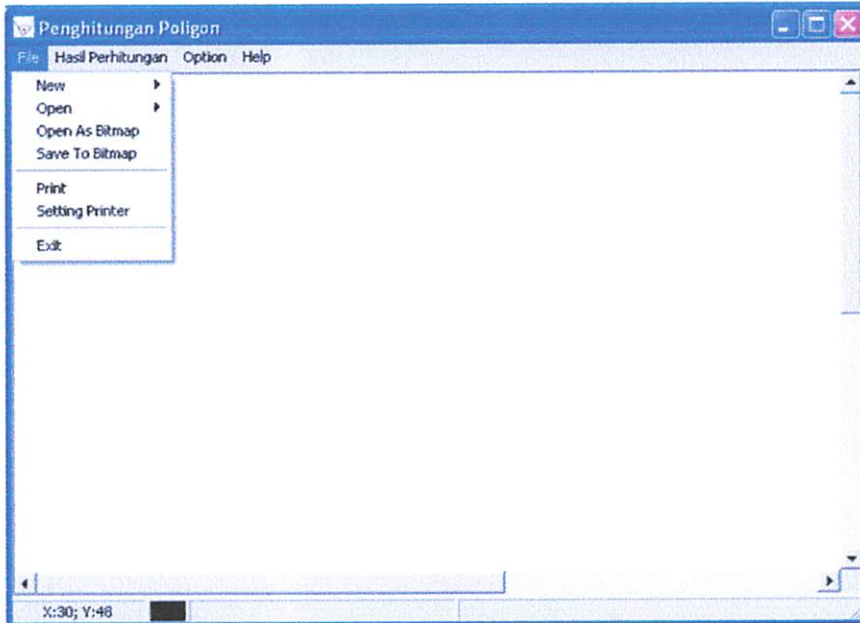


Gambar 4.2 Form Menu Utama

IV.1.1 Menu File

Pada Menu Utama Program Penghitungan Poligon dengan menggunakan Perataan Parameter *Klik* Menu File. Menu File terdiri dari beberapa Sub Menu yaitu New, Open, Open As Bitmap, Save As Bitmap, Print, Setting Printer, dan Exit.

Bentuk tampilan Menu File seperti pada gambar 4.3 :

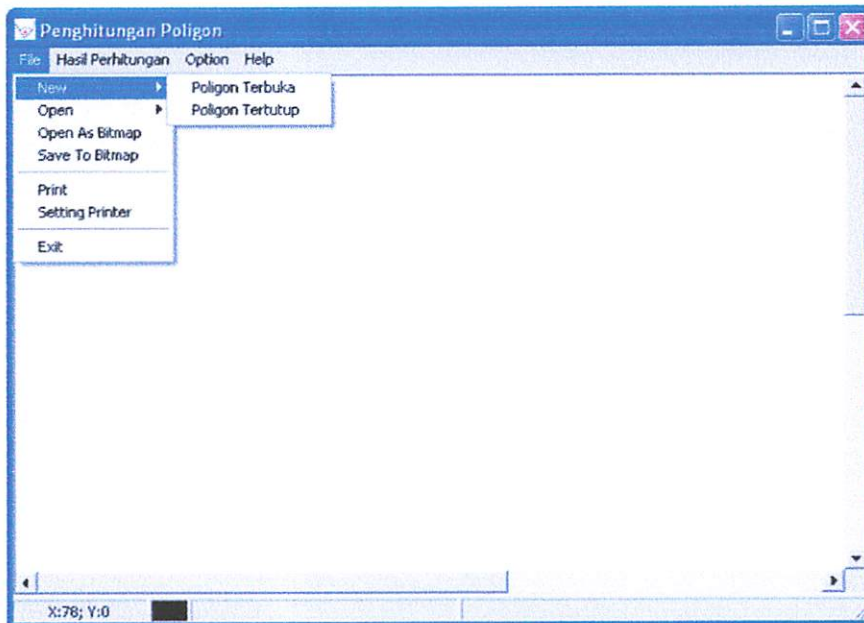


Gambar 4.3 Form Menu File

IV.1.1.1 Sub Menu New

Apabila pengguna ingin membuat input data baru maka klik New. Sub Menu New terdiri dari Poligon Terbuka Terikat Sempurna dan Poligon Tertutup. Apabila mengklik Poligon Terbuka Terikat Sempurna, maka akan keluar Form Input Data Poligon Terbuka Terikat Sempurna. Form ini digunakan untuk memasukkan koordinat BM (dua koordinat awal dan dua koordinat akhir pengukuran) atau (1 koordinat pada awal dan akhir pengukuran serta azimuth awal dan akhir), sudut mendatar, jarak mendatar, dan jumlah titik pengamatan (jumlah koordinat titik yang dicari). Sedangkan pada Form Input Data Poligon Tertutup yang dimasukkan adalah koordinat awal pengukuran, azimuth awal, sudut mendatar, jarak mendatar, jumlah titik dan tipe sudutnya (sudut dalam atau sudut luar). Setelah semua data dimasukkan, kemudian disimpan dengan ekstensi

*.PTA untuk poligon terbuka terikat sempurna dan ekstensi *.PTB untuk poligon tertutup. Bentuk tampilan Sub Menu New seperti pada gambar 4.4 :



Gambar 4.4 Form Sub Menu New

Bentuk tampilan Form Input Data Poligon Terbuka Terikat Sempurna seperti pada gambar 4.5 :

 The screenshot shows a form titled 'Input Data Poligon Terbuka Terikat Sempurna'. It is divided into several sections:

- Koordinat Awal Pengukuran:** Contains input fields for 'Titik BM1' and 'Titik BM2', each with 'X:' and 'Y:' sub-fields.
- Koordinat Akhir Pengukuran:** Contains input fields for 'Titik BM3' and 'Titik BM4', each with 'X:' and 'Y:' sub-fields.
- Buttons:** 'Hitung' (Calculate), 'Simpan' (Save), and 'Std Dev' (Standard Deviation).
- Iteration:** A checkbox for 'Iterasi' (Iteration) with a value of '0', and a checkbox for 'dengan Matrik P' (with matrix P).
- So = 0000000**: A text label.
- Jumlah Titik yang dicari:** An input field.
- Jenis Pengukuran:** A dropdown menu currently set to 'Tanpa Azimuth'.
- Entri:** A tabbed interface with tabs for 'Hasil Perhitungan', 'Pendekatan', 'Matriks F', 'Matriks A', 'Matriks AT', 'Matriks P', 'Matriks N', 'Matriks Q', 'Matriks X', 'Std Deviasi', and 'Informasi'. The 'Hasil Perhitungan' tab is active, showing two tables:

Sudut			Jarak		
No.	Derajat	Menit	Detik	No.	Jarak

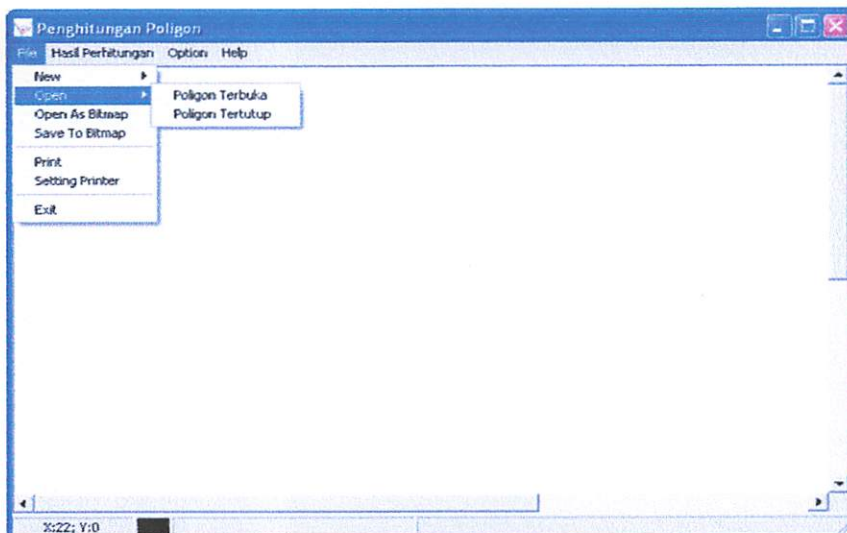
Gambar 4.5 Form Input Data Poligon Terbuka Terikat Sempurna

Bentuk tampilan Form Input Data Poligon Tertutup seperti pada gambar 4.6 :

Gambar 4.6 Form Input Data Poligon Tertutup

IV.1.1.2 Sub Menu Open

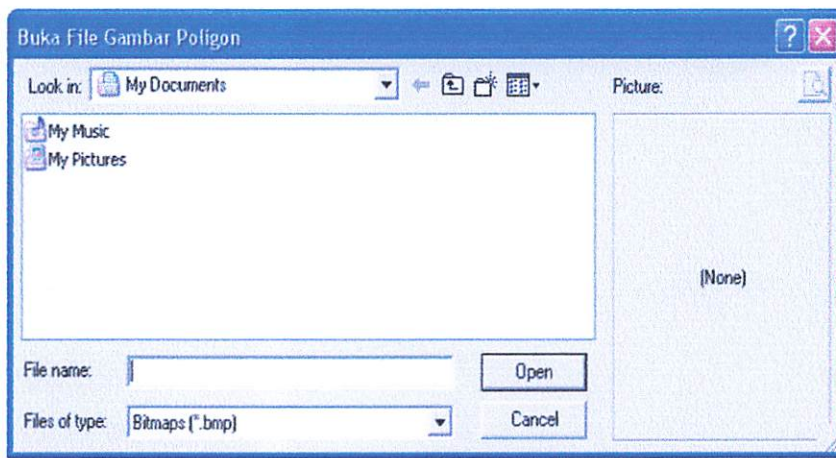
Sub Menu Open merupakan menu untuk membuka file data poligon yang sudah ada atau file yang sudah disimpan yaitu untuk poligon terbuka terikat sempurna menggunakan file dengan ekstensi *.PTA dan poligon tertutup dengan ekstensi *.PTB. Bentuk tampilannya seperti pada gambar 4.7 :



Gambar 4.7 Form Sub Menu Open

IV.1.1.3 Sub Menu Open as Bitmap

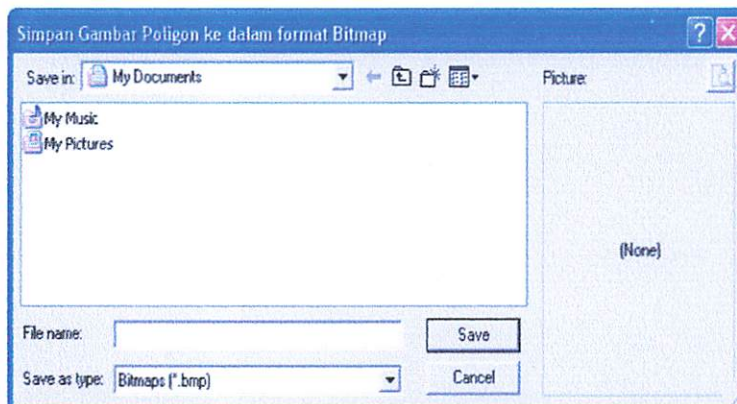
Sub Menu Open as bitmap merupakan menu untuk membuka file bitmap yang berisi gambar poligon yang sesuai dengan datanya baik untuk poligon terbuka terikat sempurna maupun poligon tertutup. Bentuk tampilannya seperti gambar 4.8 :



Gambar 4.8 Form Sub Menu Open as Bitmap

IV.1.1.4 Sub Menu Save as Bitmap

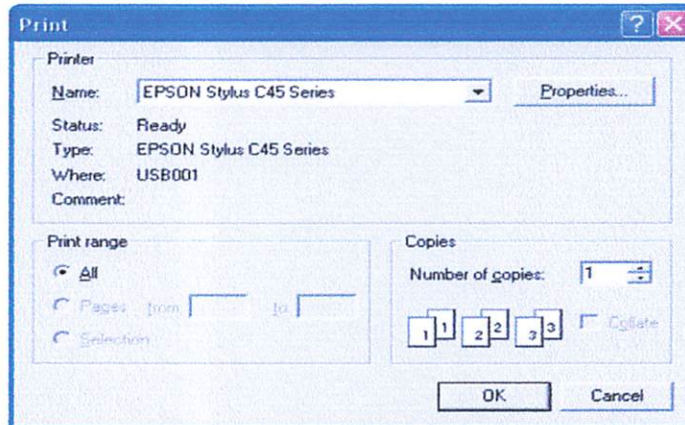
Sub Menu Save as Bitmap merupakan menu untuk menyimpan gambar poligon yang ada (poligon terbuka terikat sempurna maupun poligon tertutup) ke dalam bidang gambar yang ada. Bentuk tampilannya seperti gambar 4.9 :



Gambar 4.9 Form Sub Menu Save as Bitmap

IV.1.1.5 Sub Menu Print

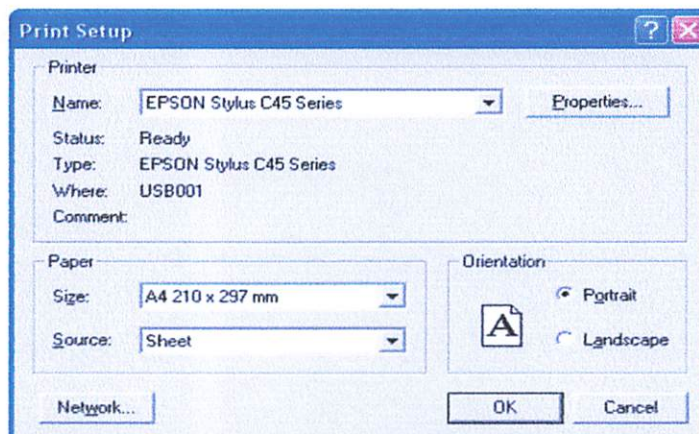
Sub Menu Print merupakan menu untuk mencetak hasil perhitungan poligon yang ada, baik untuk poligon terbuka terikat sempurna maupun poligon tertutup. Bentuk tampilannya seperti gambar 4.10 :



Gambar 4.10 Form Sub Menu Print

IV.1.1.6 Sub Menu Setting Printer

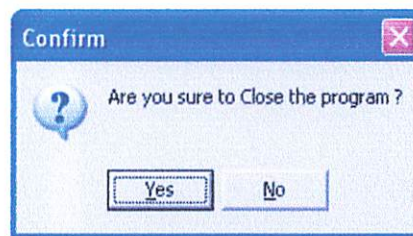
Sub Menu Setting Printer merupakan menu untuk mensetting printer yaitu mulai dari jenis printer yang digunakan, ukuran kertas, orientasi bentuk kertas (*portrait* atau *landscape*), dan lain-lain sebelum hasil cetakan siap untuk dicetak. Bentuk tampilannya seperti gambar 4.11 :



Gambar 4.11 Form Sub Menu Setting Printer

IV.1.1.7 Sub Menu Exit

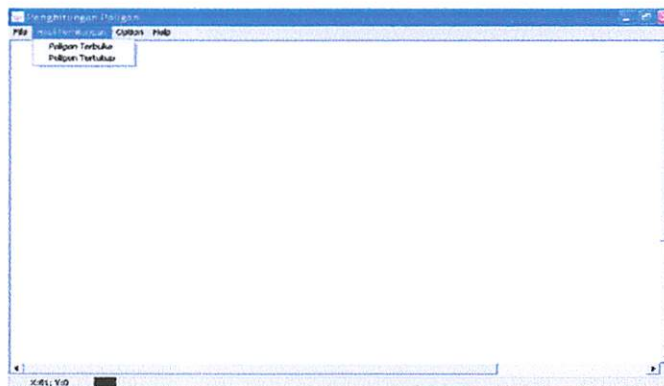
Sub Menu Exit merupakan menu untuk keluar dari Program Penghitungan Poligon dengan menggunakan Perataan Parameter. Apabila kita mengklik sub menu exit akan keluar pertanyaan apakah kita benar-benar ingin menutup program ini atau tidak (*are you sure to close the program?*). Bentuk tampilannya seperti gambar 4.12 :



Gambar 4.12 Form Sub Menu Exit

IV.1.2 Menu Hasil Perhitungan

Pada Menu Utama Program Penghitungan Poligon dengan menggunakan Perataan Parameter *Klik* Menu Hasil Perhitungan untuk melihat hasil perhitungan poligon (terbuka terikat sempurna maupun poligon tertutup) yang sudah ada atau yang sudah disimpan. Menu hasil perhitungan terdiri dari hasil perhitungan poligon terbuka terikat sempurna dan poligon tertutup. Hasil perhitungannya berupa data. Bentuk tampilan Menu Hasil Perhitungan seperti gambar 4.13 :



Gambar 4.13 Form Menu Hasil Perhitungan

Bentuk tampilan Hasil Perhitungan Poligon Terbuka Terikat Sempurna seperti pada gambar 4.14 :

The screenshot shows a web browser window with the address bar displaying "C:\TAIN\jika\program\kedi.html". The browser interface includes a menu bar (File, Edit, View, Favorites, Tools, Help) and a toolbar with icons for Back, Forward, Stop, Home, Search, Favorites, Media, and Print. The page title is "Poligon Terbuka Terikat Sempurna". The main content area displays the heading "==== Hasil Perhitungan Poligon Terbuka Terikat Sempurna =====". Below the heading is a table with 13 columns: Titik, Sudut, Hasil, Azimuth, Jarak (m), X (m), Y (m), dX (m), dY (m), X (m), Y (m), Sy, and another Y (m). The table contains data for four points: BM2, A, B, and BM3, with their respective angles, distances, and coordinates. The data is as follows:

Titik	Sudut	Hasil	Koordinat Pendekatan		Matriks X		Koordinat Definitif					
			Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Y (m)	Sy	Y (m)
BM2	270° 0' 5"	BM1-BM2	90° 0' 0,000 "	100	200,000	0,000	1,5398E-15	1,810E-22	200,00000	2,453E-15	0,00000	9,162E-19
A	09° 59' 50"	BM2-A	100° 0' 0,000 "	99,99	199,998	-99,000	1,5398E-15	1,810E-22	199,99800	2,453E-15	-99,00000	2,219E-15
B	89° 59' 59"	A-B	90° 0' 0,000 "	99,98	198,998	-99,001	1,545E-15	3,040E-15	198,99800	5,345E-15	-99,00100	5,813E-15
BM3	270° 0' 3"	B-BM3	0° 0' 2,000 "	100,02	199,999	0,999	1,545E-15	3,040E-15	198,99900	1,256E-15	0,99900	6,938E-15
		BM3-BM4	90° 0' 0,000 "	100	199,999	0,996	1,424E-14	6,162E-15				

Gambar 4.14 Form Hasil Perhitungan Poligon Terbuka Terikat Sempurna

Bentuk tampilan Hasil Perhitungan Poligon Tertutup dengan Sudut Dalam seperti pada gambar 4.15 :

The browser window shows the address bar with the URL: `D:\Taku\nta\program\hasil2.html`. The page content is as follows:

==== Hasil Perhitungan Poligon Tertutup ====

Tipe Sudut = Sudut Dalam

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy	
1.	90° 0' 1"	1-2	90° 30' 13,00 "	100,340								
2.	89° 59' 58"	2-3	180° 30' 12,00 "	68,751	1100,336	999,118	-1,2937E-05	-2,8493E-06	1100,336	1,604E-03	999,118	1,423E-05
3.	89° 59' 57"	3-4	270° 30' 14,00 "	100,350	1099,732	930,370	1,4622E-05	-6,8099E-05	1099,732	1,630E-03	930,370	3,494E-03
4.	90° 0' 0"	4-1	0° 30' 17,00 "	68,758	999,386	931,252	1,4622E-05	-6,8099E-05	999,386	2,781E-03	931,252	2,656E-03

Gambar 4.15 Form Hasil Perhitungan Poligon Tertutup Sudut Dalam

Bentuk tampilan Hasil Perhitungan Poligon Tertutup dengan sudut luar seperti pada gambar 4.16 :

Poligon Tertutup - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address: http://192.168.1.100/program/hasil2.html

Go

EPSON Web-to-Page Print Print Preview

==== Hasil Perhitungan Poligon Tertutup ====

Hasil Perhitungan Poligon Tertutup

No	Sudut	Hasil		Koordinat Perekutangan			Matriks X		Koordinat Definitif		
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Y (m)	Sy	
1.	269° 59' 59"	1-2 90° 30' 13.00"	100.340								
2.	270° 0' 2"	2-2 109° 30' 12.00"	69.751	1109.326	999.119	-1.92191-05	7.12241-04	1109.326	1.3745-03	999.119	
3.	270° 0' 8"	3-4 278° 30' 14.00"	100.350	1099.732	930.870	1.54598-05	-6.55742-05	1099.732	1.2988-03	930.870	
4.	270° 0' 0"	4-1 0° 30' 17.00"	60.756	999.306	931.252	1.54598-05	-6.55742-05	999.306	2.5062-03	931.252	

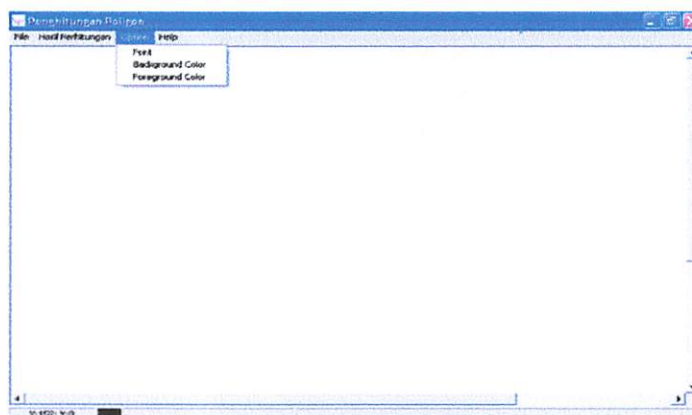
Done

My Computer

Gambar 4.16 Form Hasil Perhitungan Poligon Tertutup Sudut Luar

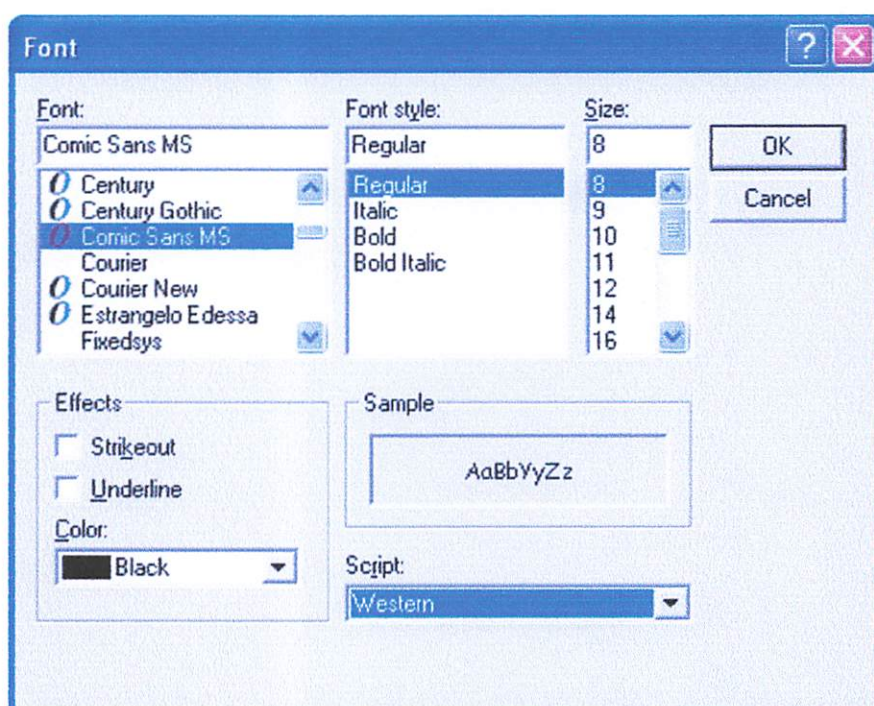
IV.1.3 Menu Option

Pada Menu Utama Program Penghitungan Poligon dengan menggunakan Perataan Parameter *Klik* Menu Option. Menu option (pilihan) terdiri dari sub menu font, background color, dan foreground color. Untuk memilih/mengganti jenis tulisan klik Sub Menu Font. Sub Menu Background Color merupakan menu untuk memilih warna background dari program sesuai keinginan pengguna. Sedangkan Foreground Color merupakan menu untuk memilih warna tulisan yang digunakan pada program. Bentuk tampilan Menu Option seperti gambar 4.17 :



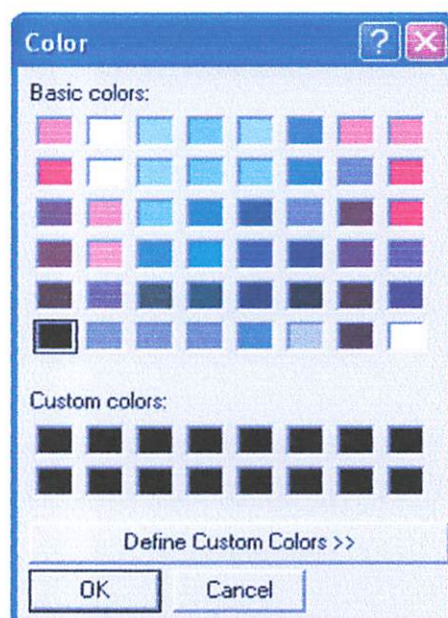
Gambar 4.17 Form Menu Option

Bentuk tampilan Sub Menu Font seperti pada gambar 4.18 :



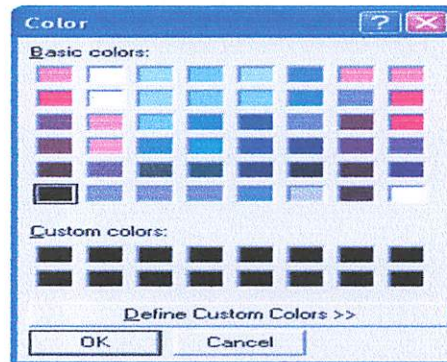
Gambar 4.18 Form Sub Menu Font

Bentuk tampilan Sub Menu Background Color seperti pada gambar 4.19 :



Gambar 4.19 Form Sub Menu Background Color

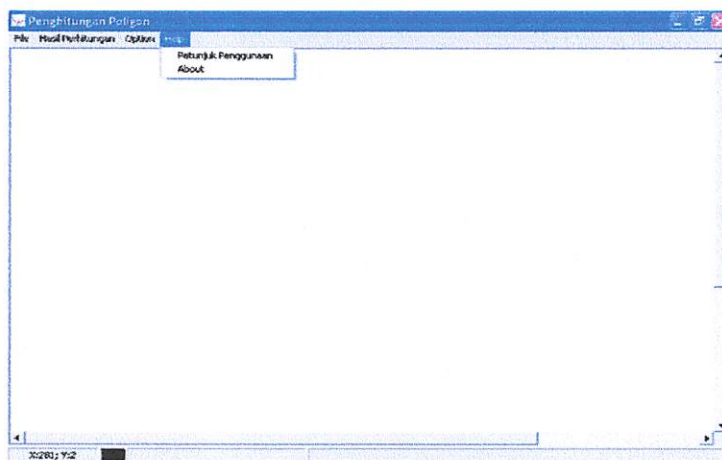
Bentuk tampilan Sub Menu Foreground Color seperti pada gambar 4.20 :



Gambar 4.20 Form Sub Menu Foreground Color

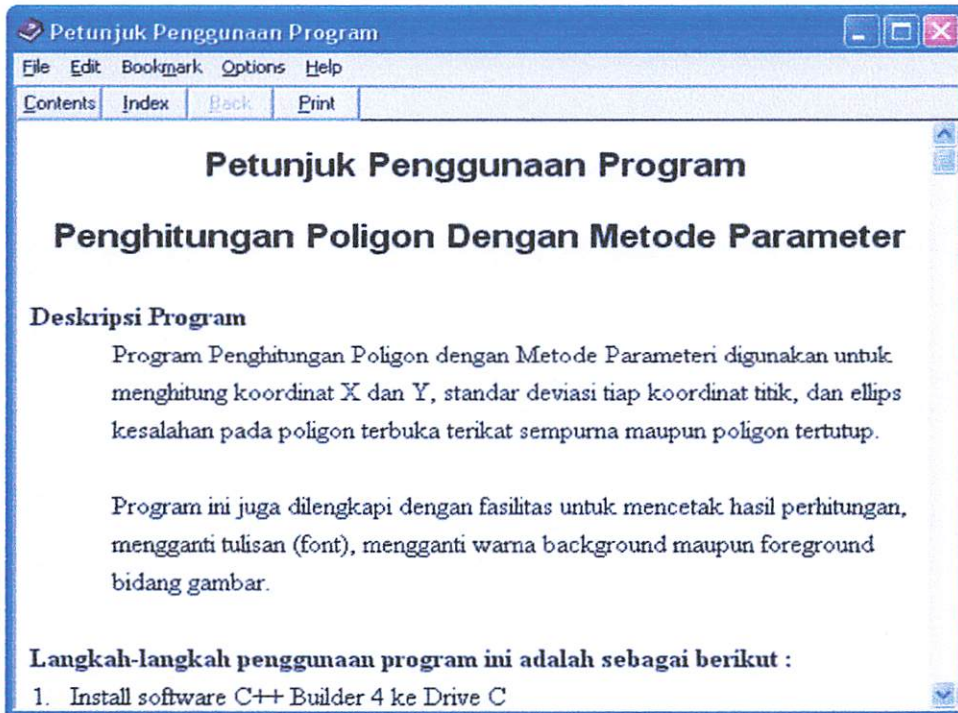
IV.1.4 Menu Help

Pada Menu Utama Program Penghitungan Poligon dengan menggunakan Perataan Parameter *Klik* Menu Help. Menu Help terdiri dari Sub Menu Petunjuk Penggunaan Program Penghitungan Poligon dengan Metode Perataan Parameter dan About Program. Apabila mengklik Sub Menu Petunjuk Penggunaan Program, maka akan dijelaskan bagaimana menggunakan Program Penghitungan Poligon dengan Metode Perataan Parameter. Sedangkan Sub Menu About Program merupakan menu yang berisi tentang program, keterangan nama program, versi, dan lain-lain. Bentuk tampilan Menu Help seperti pada gambar 4.21 :



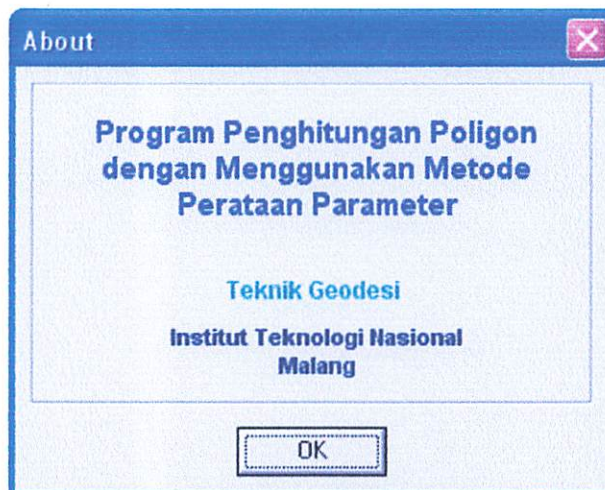
Gambar 4.21 Form Menu Help

Bentuk tampilan Sub Menu Petunjuk Penggunaan Program seperti gambar 4.22 :



Gambar 4.22 Form Sub Menu Petunjuk Penggunaan

Bentuk tampilan Sub Menu About Program seperti gambar 4.23 :



Gambar 4.23 Form Sub Menu About

IV.2 Analisa Hasil Hitung Perataan Parameter untuk Poligon

Program Penghitungan Poligon dengan Menggunakan Metode Perataan Parameter ini telah diuji coba dengan test β_1 dan β_2 . Test β_1 (*Test Laboratoris Ideal*) adalah test program dengan menggunakan data-data yang ideal definitif, bukan berasal dari data ukuran. Maksud test β_1 adalah untuk menguji “running” program apakah sudah sesuai dengan algoritma. Indikasi dari program yang sudah benar, hasil akhir berupa angka-angka koreksi (v) nol atau mendekati nol. Sedangkan test β_2 adalah test program dengan menggunakan data ukuran lapangan.

Hasil Hitung Perataan Parameter untuk Poligon yaitu poligon terbuka terikat sempurna dan poligon tertutup dengan menggunakan paket program Borland C++ Builder yaitu :

a. Poligon Terbuka Terikat Sempurna

Test β_1

- Diketahui 2 koordinat awal dan akhir pengukuran

Microsoft Internet Explorer

Address: D:\144\trial\program\test1.html

EPSON Web-To-Player Print Print Preview

==== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ====

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Y (m)	Sy	
BM2	270° 0' 0"	BM1-BM2	90° 0' 0,000"								
A	90° 0' 0"	BM2-A	180° 0' 0,000"	100	200,000	-100,000	0,000E+00	200,00000	0,000E+00	-100,00000	0,000E+00
B	90° 0' 0"	A-B	90° 0' 0,000"	100	300,000	-100,000	0,000E+00	300,00000	0,000E+00	-100,00000	0,000E+00
BM3	270° 0' 0"	B-BM3	0° 0' 0,000"	100							
		BM3-BM4	90° 0' 0,000"								

Done My Computer

Gambar 4.24 Hasil Test β_1

- Diketahui 1 koordinat awal dan akhir pengukuran

Poligon Terbuka Terikat Sempurna

Microsoft Internet Explorer

File Edit View Favorites Tools Help

Search Favorites Media

Address <D:\TAKUR\Kalsiprogram\hasil1.html> Go Link

EPSON Web-To-Page Print Print Preview

==== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ====

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy	
BM2	270 ° 0 ' 0 "	BM1-BM2	90 ° 0 ' 0,000 "	100	200,000	0,000	0,000E+00	0,000E+00	200,00000	0,000E+00	0,00000	0,000E+00
A	90 ° 0 ' 0 "	BM2-A	180 ° 0 ' 0,000 "	100	200,000	-100,000	0,000E+00	0,000E+00	200,00000	0,000E+00	-100,00000	0,000E+00
B	90 ° 0 ' 0 "	A-B	90 ° 0 ' 0,000 "	100	300,000	-100,000	0,000E+00	0,000E+00	300,00000	0,000E+00	-100,00000	0,000E+00
BM3	270 ° 0 ' 0 "	B-BM3	0 ° 0 ' 0,000 "	100	300,000	0,000	0,000E+00	0,000E+00	300,00000	0,000E+00	0,00000	0,000E+00
		BM2-BM4	90 ° 0 ' 0,000 "	100	400,000	0,000	0,000E+00	0,000E+00				

Done My Computer

Gambar 4.25 Hasil Test β_1

Test β_2

- Diketahui 2 koordinat awal dan akhir pengukuran

Dengan Matriks P

Poligon Terbuka Terikat Sempurna - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Search Favorites Media Print

Address C:\Yaku\pita\program\kadi1.html

EPSON Web-To-Page Print Print Preview

=== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ===

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy	
BM2	157° 28' 50"	BM1-BM2	25° 42' 22,979 "									
A	252° 39' 20"	BM2-A	13° 12' 13,979 "	7.20	6330,705	5642,790	-1,629E-05	-6,905E-05	6330,70498	3,095E-02	5642,78993	1,319E-02
B	112° 39' 6"	A-B	85° 31' 33,979 "	120	6450,391	5651,454	6,542E-05	-6,303E-05	6450,39107	1,701E-02	5651,45394	1,417E-02
C	110° 37' 30"	B-C	18° 30' 39,979 "	120	6488,490	5765,246	6,542E-05	-6,303E-05	6488,49009	1,137E-02	5765,24585	1,264E-02
D	171° 14' 10"	C-D	209° 28' 9,979 "	120	6395,854	5841,526	7,886E-05	-4,492E-04	6395,85396	1,898E-02	5841,52604	1,116E-02
BM3	299° 28' 30"	D-BM3	300° 42' 19,979 "	8.98								
		BM3-BM4	60° 10' 49,979 "									

Done My Computer

Gambar 4.26 Hasil Test β_2 (dengan matriks P)

Tanpa Matriks P

Poligon Terbuka Terikat Sempurna - Microsoft Internet Explorer

Address: D:\TAKU\program\hasil\hasil1.html

EPSON Web-To-Page Print Print Preview

=== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ===

Titik	Sudut		Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
			Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
BM2	157° 28' 50"	BM1-BM2	35° 43' 23,979"									
A	252° 39' 20"	BM2-A	13° 12' 13,979"	7,20	6330,705	5642,790	-1,629E-05	-6,905E-05	6330,70498	6,106E-05	5642,78993	2,602E-04
B	112° 39' 6"	A-B	85° 51' 33,979"	120	6450,391	5651,454	6,542E-05	-6,303E-05	6450,39107	3,356E-04	5651,45394	2,797E-04
C	110° 57' 30"	B-C	18° 30' 39,979"	120	6488,490	5765,246	6,542E-05	-6,303E-05	6488,49008	2,282E-04	5765,24555	2,494E-04
D	171° 14' 10"	C-D	309° 28' 9,979"	120	6395,854	5841,526	7,886E-05	-4,492E-04	6395,85296	3,743E-04	5841,52604	2,202E-04
BM3	299° 28' 30"	D-BM3	300° 42' 19,979"	8,98								
		BM3-BM4	60° 10' 49,979"									

Done My Computer

Gambar 4.27 Hasil Test β_2 (tanpa matriks P)

- Diketahui 1 koordinat awal dan akhir pengukuran

Dengan Matriks P

Poligon Terbuka Terikat Sempurna - Microsoft Internet Explorer

Address: D:\TAKU\program\hasil\hasil1.html

EPSON Web-To-Page Print Print Preview

=== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ===

Titik	Sudut		Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
			Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
BM2	157° 28' 50"	BM1-BM2	35° 43' 23,979"	22,459	6229,060	5635,780	-2,674E-04	-6,029E-04	6229,05973	1,751E-02	5625,77940	2,467E-02
A	252° 39' 20"	BM2-A	13° 12' 13,979"	7,20	6330,705	5642,790	-2,674E-04	-6,029E-04	6330,70496	1,728E-02	5642,79017	3,063E-02
B	112° 39' 6"	A-B	85° 51' 33,979"	120	6450,391	5651,455	-4,111E-05	1,724E-04	6450,38950	3,116E-02	5651,45434	4,348E-02
C	110° 57' 30"	B-C	18° 30' 39,979"	120	6488,489	5765,247	-4,111E-05	1,724E-04	6488,48937	3,384E-02	5765,24668	4,530E-02
D	171° 14' 10"	C-D	309° 28' 9,979"	120	6395,853	5841,526	-1,500E-03	-6,563E-04	6395,85339	3,899E-02	5841,52578	2,264E-02
BM3	299° 28' 30"	D-BM3	300° 42' 19,979"	8,98	6388,132	5846,112	-1,500E-03	-6,563E-04	6388,13197	2,673E-02	5846,11209	1,704E-02
		BM3-BM4	60° 10' 49,979"	15,970	6401,988	5854,053	-6,259E-04	-3,178E-04				

Done My Computer

Gambar 4.28 Hasil Test β_2 (dengan matriks P)

Tanpa Matriks P

Poligon Terbuka Terikat Sempurna - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media Mail Print Print Preview

Address D:\TAU\Program\hasil\program\hasil1.html

EPSON Web-To-Page Print Print Preview

=== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ===

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
			Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
BM2	137° 28' 50"	BM1-BM2	35° 43' 23,000 "	22,459	6329,060	5635,780	-2,674E-04	-6,029E-04	6329,05973	2,372E-05	5635,77940	3,342E-05
A	232° 39' 20"	BM2-A	13° 12' 13,000 "	7,20	6330,705	5642,790	-2,674E-04	-6,029E-04	6330,70496	2,341E-05	5642,79017	4,150E-05
B	112° 39' 6"	A-B	85° 51' 33,000 "	120	6450,391	5651,455	-4,111E-05	1,724E-04	6450,38950	4,221E-05	5651,45434	3,890E-05
C	110° 57' 30"	B-C	18° 30' 29,000 "	120	6488,489	5765,247	-4,111E-05	1,724E-04	6488,48837	4,204E-05	5765,24668	6,137E-05
D	171° 14' 10"	C-D	309° 28' 9,000 "	120	6395,853	5841,526	-1,500E-03	-6,563E-04	6395,85339	5,283E-05	5841,52578	3,070E-05
BM3	299° 28' 30"	D-BM3	300° 42' 19,000 "	8,98	6388,132	5846,112	-1,500E-03	-6,563E-04	6388,13197	3,621E-05	5846,11209	2,312E-05
		BM3-BM4	60° 10' 49,000 "	15,970	6401,988	5854,053	-6,259E-04	-3,178E-04				

Done My Computer

Gambar 4.29 Hasil Test β_2 (tanpa matriks P)

b. Poligon Tertutup

Test β_1

- Sudut Dalam

Poligon Tertutup - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Search Favorites Media Print Print Preview

Address D:\TAU\Program\hasil2.html

EPSON Web-To-Page Print Print Preview

=== Hasil Perhitungan Poligon Tertutup ===

Tipe Sudut = Sudut Dalam

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy	
1.	108° 0' 0"	1-2	45° 0' 0,00"	100								
2.	108° 0' 0"	2-3	117° 0' 0,00"	100	1070,711	1070,711	0,0000E+00	0,0000E+00	1070,711	0,0000E+00	1070,711	0,0000E+00
3.	108° 0' 0"	3-4	109° 0' 0,00"	100	1159,811	1025,312	0,0000E+00	0,0000E+00	1159,811	0,0000E+00	1025,312	0,0000E+00
4.	108° 0' 0"	4-5	261° 0' 0,00"	100	1144,168	926,543	0,0000E+00	0,0000E+00	1144,168	0,0000E+00	926,543	0,0000E+00
5.	108° 0' 0"	5-1	333° 0' 0,00"	100	1045,399	910,899	0,0000E+00	0,0000E+00	1045,399	0,0000E+00	910,899	0,0000E+00

Done

My Computer

Gambar 4.30 Hasil Test β_1

- Sudut Luar

==== Hasil Perhitungan Poligon Tertutup ====

Tipe Sudut = Sudut Luar

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
1.	252° 0' 0"	1-2 45° 0' 0.00"	100								
2.	252° 0' 0"	2-3 117° 0' 0.00"	100	1070,711	1070,711	0,0000E+00	0,0000E+00	1070,711	0,000E+00	1070,711	0,000E+00
3.	252° 0' 0"	3-4 189° 0' 0.00"	100	1159,811	1025,312	0,0000E+00	0,0000E+00	1159,811	0,000E+00	1025,312	0,000E+00
4.	252° 0' 0"	4-5 261° 0' 0.00"	100	1144,168	926,543	0,0000E+00	0,0000E+00	1144,168	0,000E+00	926,543	0,000E+00
5.	252° 0' 0"	5-1 333° 0' 0.00"	100	1045,399	910,899	0,0000E+00	0,0000E+00	1045,399	0,000E+00	910,899	0,000E+00

Gambar 4.31 Hasil Test β_1

Test β_2

- Sudut Dalam

Dengan Matriks P

==== Hasil Perhitungan Poligon Tertutup ====

Tipe Sudut = Sudut Dalam

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
1.	90° 15' 12"	1-2 45° 0' 0.00"	37,658								
2.	94° 28' 19"	2-3 134° 44' 48,00"	105,921	1040,770	1040,770	1,1644E+00	-2,3506E-02	1041,934	2,392E+00	1040,736	1,406E+00
3.	97° 10' 53"	3-4 220° 21' 29,00"	53,346	1115,898	966,205	2,3460E-01	8,5336E-01	1116,233	6,878E-01	967,058	3,034E+00
4.	89° 19' 27"	4-5 303° 10' 36,00"	72,470	1081,453	929,354	2,3460E-01	8,5336E-01	1082,500	2,342E+00	926,127	1,399E+00
5.	168° 51' 21"	5-1 33° 51' 9,00"	45,154	1020,797	965,212	1,0469E+00	5,7260E-01	1021,944	2,118E+00	964,466	2,137E+00

Gambar 4.32 Hasil Test β_2 (dengan matriks P)

Tanpa Matriks P

==== Poligon Tertutup ==== Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media Print Print Preview

Address http://D:\TAkuj\kita\program\test2.html

EPSON Web-To-Page Print Print Preview

==== Hasil Perhitungan Poligon Tertutup ====

Tipe Sudut = Sudut Dalam

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif				
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy	
1.	90 ° 15 ' 12 "	1-2	45 ° 0 ' 0.00 "	57,658								
2.	94 ° 23 ' 19 "	2-3	134 ° 44 ' 48.00 "	105,921	1040,770	1040,770	1.1644E+00	-3.3506E-02	1041,924	2.392E+00	1040,736	1.406E+00
3.	97 ° 10 ' 53 "	3-4	220 ° 21 ' 29.00 "	53,346	1115,998	966,205	2.3460E-01	8.5336E-01	1116,233	6.879E-01	967,058	3.034E+00
4.	89 ° 19 ' 27 "	4-5	303 ° 10 ' 36.00 "	72,470	1081,453	925,534	2.3460E-01	8.5336E-01	1082,500	2.342E+00	926,127	1.395E+00
5.	169 ° 51 ' 21 "	5-1	33 ° 51 ' 9.00 "	45,154	1020,797	965,212	1.0469E+00	5.7260E-01	1021,944	2.110E+00	964,466	2.137E+00

Done

My Computer

Gambar 4.33 Hasil Test β2 (tanpa matriks P)

- Sudut Luar

Dengan Matriks P

Poligon Tertutup - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <D:\TAluknits\program\hasi2.html>

Go Link

EPSON Web-To-Page - Print Print Preview

=== Hasil Perhitungan Poligon Tertutup ===

Tipe Sudut == Sudut Luar

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif			
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Sx	Y (m)	Sy
1.	269 ° 44 ' 48"	1-2	45 ° 0 ' 0.00 "	57,658							
2.	265 ° 36 ' 41"	2-3	134 ° 44 ' 48.00 "	105,921	1040,770	1040,770	1.1623E+00	-3,5449E-02	1041,922	2,401E+00	1040,735 1,410E+00
3.	262 ° 49 ' 7"	3-4	220 ° 21 ' 29.00 "	53,346	1115,998	966,205	2,2005E-01	0,5522E-01	1116,227	6,501E-01	967,060 3,044E+00
4.	270 ° 40 ' 33"	4-5	303 ° 10 ' 36.00 "	72,470	1081,453	925,534	2,2885E-01	8,5522E-01	1082,505	2,350E+00	926,126 1,400E+00
5.	191 ° 8 ' 39"	5-1	33 ° 51 ' 9.00 "	45,154	1020,797	965,212	1,0523E+00	5,7227E-01	1021,952	2,125E+00	964,465 2,144E+00

Done

My Computer

Gambar 4.34 Hasil Test β2 (dengan matriks P)

Tanpa Matriks P

Tipe Sudut == Sudut Luar

Titik	Sudut	Hasil		Koordinat Pendekatan		Matriks X		Koordinat Definitif		
		Azimuth	Jarak (m)	X (m)	Y (m)	dX (m)	dY (m)	X (m)	Y (m)	Sy
1.	269° 44' 48"	1-2 45° 0' 0,00"	57,658							
2.	265° 36' 41"	2-3 134° 44' 48,00"	105,921	1040,770	1040,770	1,1625E+00	-3,5449E-02	1041,932	2,401E+00	1,410E+00
3.	262° 49' 7"	3-4 220° 21' 29,00"	53,346	1115,998	966,205	2,2885E-01	8,5322E-01	1116,227	6,901E-01	967,060
4.	270° 40' 33"	4-5 303° 10' 36,00"	72,470	1081,453	925,554	2,2885E-01	8,5322E-01	1082,505	2,350E+00	926,126
5.	191° 8' 39"	5-1 33° 51' 9,00"	45,154	1020,797	965,212	1,0523E+00	5,7227E-01	1021,952	2,125E+00	964,465

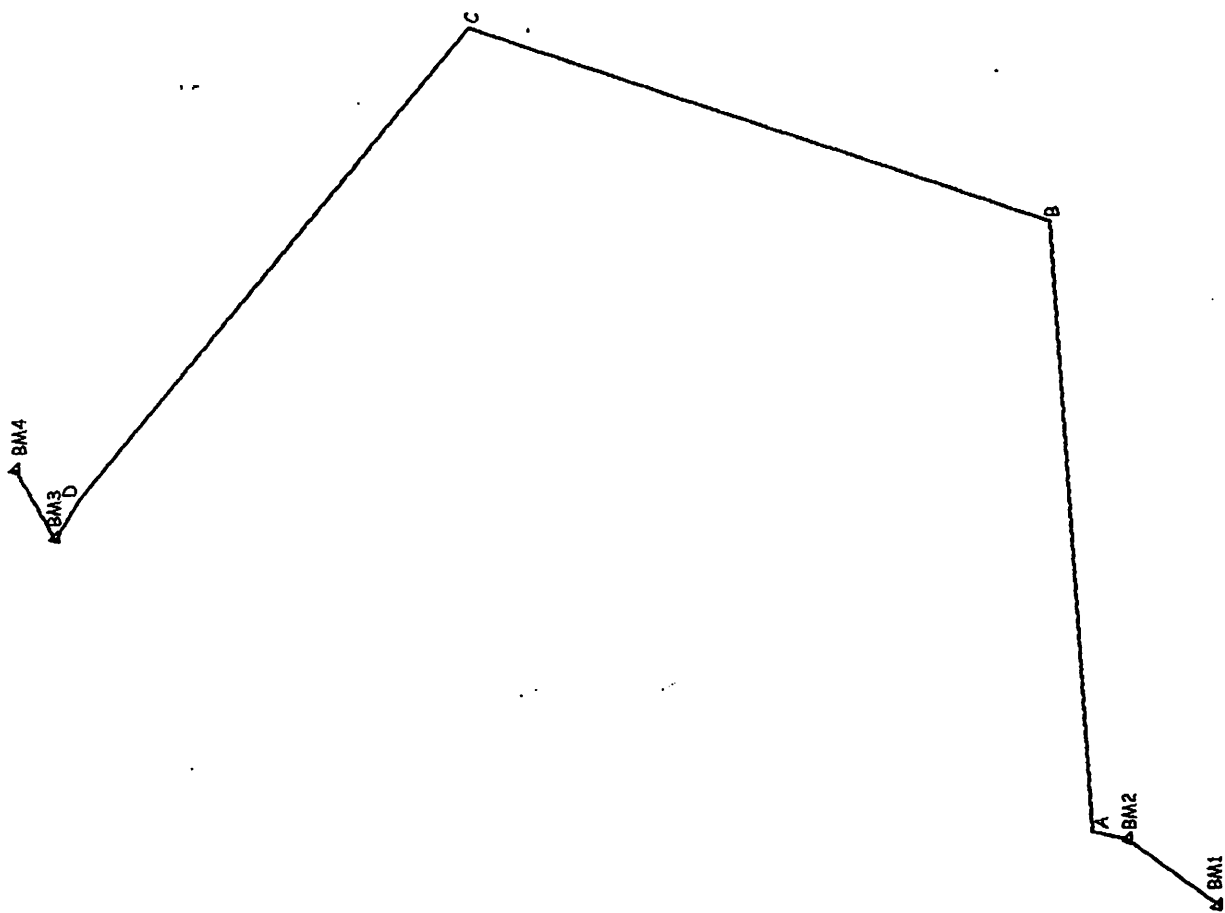
Gambar 4.35 Hasil Test β_2 (tanpa matriks P)

IV.3. Uji Ketelitian

Untuk mengetahui kebenaran dari hasil Program Penghitungan Poligon menggunakan Perataan Parameter pada Borland C++ Builder, maka hasil perhitungan dari program tersebut harus dibandingkan dengan hasil perhitungan dengan menggunakan hitungan manual (*Microsoft Excel*). Hal ini dimaksudkan untuk menunjukkan hasil dari Program Penghitungan Poligon dengan menggunakan Perataan Parameter ini benar-benar telah teruji kebenarannya.

IV.3.1 Poligon Terbuka Terikat Sempurna

Perbedaan hasil hitungan pada Poligon Terbuka Terikat Sempurna untuk mencari koordinat X dan Y, standard deviasi, standard deviasi tiap titik koordinat, ketelitian linier, serta ellips kesalahan dengan menggunakan program Borland C++ Builder dan Microsoft Excel menggunakan data ukuran lapangan (β_2) pada iterasi kedua yaitu :



P = 11 x 11										
d1	0,001	0	0	0	0	0	0	0	0	0
d2	0	0,001	0	0	0	0	0	0	0	0
d3	0	0	0,001	0	0	0	0	0	0	0
d4	0	0	0	0,001	0	0	0	0	0	0
d5	0	0	0	0	0,001	0	0	0	0	0
b1	0	0	0	0	0	0,001	0	0	0	0
b2	0	0	0	0	0	0	0,001	0	0	0
b3	0	0	0	0	0	0	0	0,001	0	0
b4	0	0	0	0	0	0	0	0	0,001	0
b5	0	0	0	0	0	0	0	0	0	0,001
b6	0	0	0	0	0	0	0	0	0	0,001

- Hasil Perhitungan dengan menggunakan Matriks P
- a. Matriks P

d1	7,20
d2	120
d3	120
d4	120
d5	8,98

JARAK HORIZONTAL (m)

	•	'	"
b1	157	28	50
b2	252	39	20
b3	112	39	6
b4	110	57	30
b5	171	14	10
b6	299	28	30

SUDUT HORIZONTAL

KOORDINAT AKHIR PENGUKURAN					
BM2	X (m)	6329,060	Y (m)	5635,780	
	Y (m)		X (m)	6401,968	5854,019

KOORDINAT AWAL PENGUKURAN					
BM1	X (m)	6315,947	Y (m)	5617,547	
	Y (m)		X (m)	6388,113	5846,077

- Data ukuran yang diketahui :

b. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
A	6330,70498	5642,78993	6330,705	5642,789
B	6450,39107	5651,45394	6450,391	5651,454
C	6488,49008	5765,24555	6488,490	5765,245
D	6395,85369	5841,52604	6395,854	5841,526

c. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

• Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0,0008012 m	0,00091889 m

• Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	Sx	Sy	Sx	Sy
A	0,003095	0,01319	0,00358369	0,01527434
B	0,01701	0,01417	0,01948438	0,01639507
C	0,01157	0,01264	0,01347686	0,0156248
D	0,01898	0,0111	0,02206324	0,01290667

d. Ellips Kesalahan

Koordinat Titik		Borland C++ Builder			Microsoft Excel
A	t	13°	13,20635 12'	22,86"	13° 12' 14",44037
	Su(±)		0,01355		0,01568912
	Sv (±)		8,844E-07		1,01E-06

B	t	50°	50,2743993 16'	27,8376295"	50° 1' 4",974786
	Su(±)		0,02205		0,0253425
	Sv (±)		0,001933		0,00248937
C	t	47°	47,64869 38'	55,284"	49° 27' 13",0698
	Su(±)		0,01695		0,02036504
	Sv (±)		0,002494		0,00332046
D	t	30°	30,4515424 27'	5,55264"	30° 19' 37",19155
	Su(±)		0,02201		0,02556108
	Sv (±)		1,103E-06		1,26E-06

o Hasil Perhitungan tanpa Matriks P (Matriks P dianggap 1)

a. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
A	6330,70498	5642,78993	6330,705	5642,789
B	6450,39107	5651,45394	6450,391	5651,454
C	6488,49008	5765,24555	6488,490	5765,245
D	6395,85369	5841,52604	6395,854	5841,526

b. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

- Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0,0004999 m	0,00091891 m

- Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Bulder		Microsoft Excel	
	Sx	Sy	Sx	Sy
A	0,00006106	0,0002602	0,00011333	0,00048303
B	0,0003356	0,0002797	0,00061616	0,00051847
C	0,0002282	0,0002494	0,00042318	0,00049411
D	0,0003745	0,0002202	0,00069771	0,00040815

c. Ketelitian Linier

Ketelitian linier jaraknya adalah 1 : 680200

d. Ellips Kesalahan

Koordinat Titik		Borland C++ Bulder	Microsoft Excel
A	t	13,20631851 13° 12' 22,7466444"	13° 12' 14",43172
	Su(±)	0,0002673	0,00049614
	Sv (±)	1,75E-08	3,21E-08
B	t	50,27408764 50° 16' 26,7155097"	50° 1' 4",974953
	Su(±)	0,0004351	0,00080141
	Sv (±)	3,83E-05	7,87E-05
C	t	47,64888054 47° 38' 55,9699444"	49° 27' 13",06969
	Su(±)	0,0003345	0,00064401
	Sv (±)	4,92E-05	0,000105
D	t	30,45155967 30° 27' 5,61480066"	30° 19' 37",19155
	Su(±)	0,000363654	0,00080833
	Sv (±)	2,18E-08	3,99E-08

Untuk Poligon Terbuka Terikat Sempurna dengan diketahui azimuth, hasil hitungannya dibandingkan dengan hasil hitungan Poligon Terbuka Terikat Sempurna yang azimuthnya dihitung pada iterasi kedua test $\beta 2$ paket program Borland C++ Builder.

Data ukuran yang diketahui adalah :

KOORDINAT							
BM1		X (m)	Y (m)		BM4	X(m)	Y (m)
		6315,947	5617,547			6401,968	5854,019

Azimuth awal ($\alpha 12$)				Azimuth akhir ($\alpha 43$)			
	35°	43'	23,979"		240°	10'	49,979"

SUDUT HORIZONTAL

		°	'	"
$\beta 1$	=	157	28	50
$\beta 2$	=	252	39	20
$\beta 3$	=	112	39	6
$\beta 4$	=	110	57	30
$\beta 5$	=	171	14	10
$\beta 6$	=	299	28	30

JARAK HORIZONTAL (m)

dBM1-BM2	=	22,459
d1	=	7,20
d2	=	120
d3	=	120
d4	=	120
d5	=	8,98
dBM3-BM4	=	15,970

- o Hasil Perhitungan dengan menggunakan Matriks P
- a. Matriks P

Matriks P

dBM2	d1	d2	d3	d4	d5	dBM3	b1	b2	b3	b4	b5	b6
0,001	0	0	0	0	0	0	0	0	0	0	0	0
0	0,001	0	0	0	0	0	0	0	0	0	0	0

P =
13 × 13

0	0	0,001	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0,001	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0,001	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0,001	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0,001	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0,001	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0,001	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0,001	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0,001	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0,001	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0,001	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0,001

b. Koordinat X dan Y

Koordinat Titik	Diketahui azimuth		Azimuth dihitung	
	X (m)	Y (m)	X (m)	Y (m)
BM2	6329,05973	5635,7794	6329,060	5635,780
A	6330,70496	5642,79017	6330,70498	5642,78993
B	6450,3895	5651,45434	6450,39107	5651,45394
C	6488,48837	5765,24668	6488,49008	5765,24555
D	6395,85339	5841,52578	6395,85369	5841,52604
BM3	6388,13197	5846,11209	6388,113	5846,077

c. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

- Standard Deviasi

So	Diketahui azimuth	Azimuth dihitung
	0,001082 m	0,0008012 m

- Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Diketahui azimuth		Azimuth dihitung	
	Sx (m)	Sy (m)	Sx (m)	Sy (m)
BM2	0,01751	0,02467		
A	0,01728	0,03063	0,003095	0,01319
B	0,03116	0,04348	0,01701	0,01417
C	0,03384	0,0453	0,01157	0,01264

D	0,03899	0,02266	0,01898	0,0111
BM3	0,02673	0,01706		

o Hasil Perhitungan tanpa Matriks P (Matriks P dianggap 1)

a. Koordinat X dan Y

Koordinat Titik	Diketahui azimuth		Azimuth dihitung	
	X (m)	Y (m)	X (m)	Y (m)
BM2	6329,05973	5635,7794	6329,060	5635,780
A	6330,70496	5642,79017	6330,70498	5642,78993
B	6450,3895	5651,45434	6450,39107	5651,45394
C	6488,48837	5765,24668	6488,49008	5765,24555
D	6395,85339	5841,52578	6395,85369	5841,52604
BM3	6388,13197	5846,11209	6388,113	5846,077

b. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

• Standard Deviasi

So	Diketahui azimuth	Azimuth dihitung
	0,00004635 m	0,0004999 m

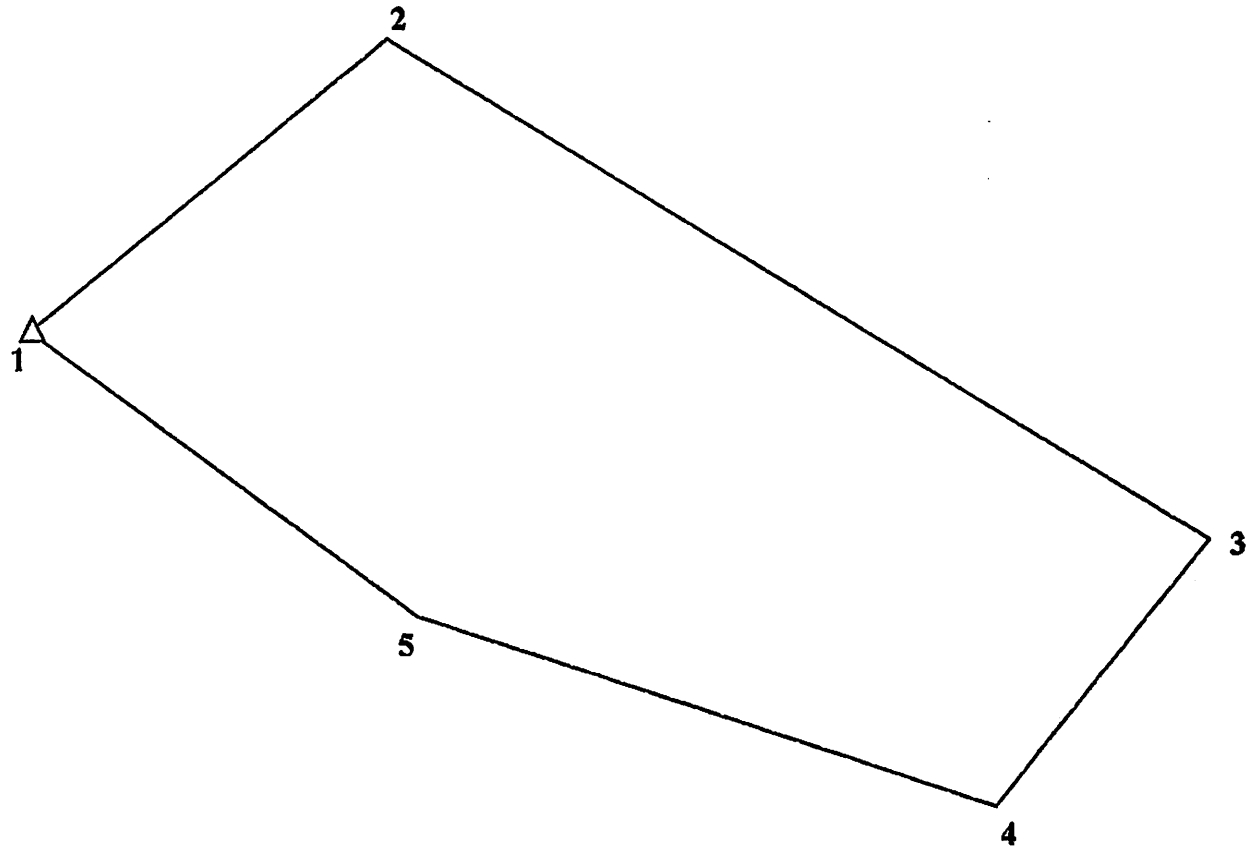
• Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Diketahui azimuth		Azimuth dihitung	
	Sx (m)	Sy (m)	Sx (m)	Sy (m)
BM2	0,00002372	0,00003342		
A	0,00002341	0,0000415	0,00006106	0,0002602
B	0,00004221	0,0000589	0,0003356	0,0002797
C	0,00004584	0,00006137	0,0002282	0,0002494
D	0,00005283	0,0000307	0,0003745	0,0002202
BM3	0,00003621	0,00002312		

• Ketelitian Linier

Ketelitian linier jaraknya adalah 1 : 141700

SKET POLIGON TERTUTUP



IV.3.2 Poligon Tertutup

Uji ketelitian pada poligon tertutup dibagi menjadi dua yaitu untuk poligon tertutup dengan menggunakan sudut dalam dan menggunakan sudut luar.

IV.3.2.1 Poligon Tertutup dengan Sudut Dalam

Perbedaan hasil hitungan Poligon Tertutup sudut dalam untuk mencari koordinat X dan Y, standard deviasi, standard deviasi tiap titik koordinat, serta ellips kesalahan pada program Borland C++ Builder dan Microsoft Excel menggunakan data data ukuran lapangan (β_2) pada iterasi pertama adalah :

- Data ukuran yang diketahui

KOORDINAT AWAL		
Titik 1	X1 (m)	Y1 (m)
	1000	1000

Azimuth awal (α_{12})			
α_{12}	45°	0'	0"

SUDUT HORIZONTAL

	°	'	"
β_1	90	15	12
β_2	94	23	19
β_3	97	10	53
β_4	89	19	27
β_5	168	51	21

JARAK HORIZONTAL (m)

d1-2	57,658
d2-3	105,921
d3-4	53,346
d4-5	72,470
d5-1	45,154

Hasil Perhitungan dengan menggunakan Matriks P

a. Matriks P

d1	d2	d3	d4	d5	β_1	β_2	β_3	β_4	β_5
0,001	0	0	0	0	0	0	0	0	0
0	0,001	0	0	0	0	0	0	0	0
0	0	0,001	0	0	0	0	0	0	0

P = 10 × 10	0	0	0	0,001	0	0	0	0	0	0
	0	0	0	0	0,001	0	0	0	0	0
	0	0	0	0	0	0,001	0	0	0	0
	0	0	0	0	0	0	0,001	0	0	0
	0	0	0	0	0	0	0	0,001	0	0
	0	0	0	0	0	0	0	0	0,001	0
	0	0	0	0	0	0	0	0	0	0,001
	0	0	0	0	0	0	0	0	0	0,001

b. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
2	1040,7706	1040,7704	1040,772	1070,771
3	1121,1793	971,8233	1121,180	971,823
4	1091,7897	927,3027	1091,791	927,304
5	1031,7879	967,9363	1031,788	967,936

c. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

- Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0.003839 m	0,003813309 m

- Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	Sx	Sy	Sx	Sy
2	0,07866	0,05539	0,078122324	0,055016555
3	0,01582	0,125	0,015712823	0,124197707
4	0,08824	0,06057	0,087639823	0,060153364
5	0,07346	0,05949	0,072962376	0,059087278

b. Ellips Kesalahan

Koordinat Titik		Borland C++ Builder			Microsoft Excel		
2	t	61.84374444			61°	50'	35",259531
		61°	50'	37,4800146"			

	Su(±)	0,08625	0,085665026
	Sv (±)	0,04262	4,23E-02
3	t	0.8973532904 0° 53' 50,47184"	0° 53' 50",100175
	Su(±)	0,1251	0,124212697
	Sv (±)	0,01570	0,01559388
4	t	58.509622662 58° 30' 34,64158379"	58° 30' 44",107137
	Su(±)	0,1013	0,100597396
	Sv (±)	0,03458	0,03434137
5	t	36,594147943 36° 35' 38,93259635"	36° 35' 31",73942
	Su(±)	0,08763	0,08702921
	Sv (±)	0,03546	3,52E-02

○ Hasil Perhitungan tanpa Matriks P (Matriks P dianggap 1)

a. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
2	1040,7706	1040,7704	1040,772	1040,771
3	1121,1793	971,8233	1121,180	971,823
4	1091,7897	927,3027	1091,791	927,304
5	1031,7879	967,9363	1031,788	967,936

b. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

• Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0.003839 m	0,003813626 m

- Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	Sx	Sy	Sx	Sy
2	0,002487	0,001752	0,00247065	0,001739921
3	0,0005002	0,003954	0,000496924	0,003927803
4	0,00279	0,001915	0,002771645	0,001902374
5	0,002323	0,001881	0,002307465	0,001868659

c. Ellips Kesalahan

Koordinat Titik		Borland C++ Builder	Microsoft Excel
2	t	61.84373444 61° 50' 33,980964"	61° 50' 35",25973828
	Su(±)	0.002727	0,002709128
	Sv (±)	0.001348	9,47E-04
3	t	0,89735329112 0° 53' 50,471848"	0° 53' 50",10017828
	Su(±)	0,003955	0,003928277
	Sv (±)	0,00004965	4,93E-04
4	t	58,50962269027 58° 30' 34,641685"	58° 30' 44",10729233
	Su(±)	0,003203	0,003181433
	Sv (±)	0,001093	0,00108606
5	t	36,5941479685 36° 35' 38,9326867"	36° 35' 31",73180363
	Su(±)	0,002771	0,002752334
	Sv (±)	0,001121	1,11E-03

IV.3.2.2 Poligon Tertutup dengan Sudut Luar

Perbedaan hasil hitungan Poligon Tertutup sudut luar untuk mencari koordinat X dan Y, standard deviasi, standard deviasi tiap titik koordinat, serta ellips kesalahan pada program Borland C++ Builder dan Microsoft Excel menggunakan data yang ideal (test $\beta 1$) dan data ukuran lapangan ($\beta 2$) pada iterasi pertama adalah :

- o Data ukuran yang diketahui

KOORDINAT AWAL		
Titik 1	X1 (m)	Y1 (m)
	1000	1000

Azimuth awal ($\alpha 12$)			
$\alpha 12$	45°	0'	0"

SUDUT HORIZONTAL

	°	'	"
$\beta 1$	269	44	48
$\beta 2$	265	36	41
$\beta 3$	262	49	7
$\beta 4$	270	40	33
$\beta 5$	191	8	39

JARAK HORIZONTAL (m)

d1-2	57,658
d2-3	105,921
d3-4	53,346
d4-5	72,470
d5-1	45,154

- o Hasil Perhitungan dengan menggunakan Matriks P

a. Matriks P

P = 10 × 10	d1	d2	d3	d4	d5	$\beta 1$	$\beta 2$	$\beta 3$	$\beta 4$	$\beta 5$
	0,001	0	0	0	0	0	0	0	0	0
	0	0,001	0	0	0	0	0	0	0	0
	0	0	0,001	0	0	0	0	0	0	0
	0	0	0	0,001	0	0	0	0	0	0
	0	0	0	0	0,001	0	0	0	0	0
	0	0	0	0	0	0,001	0	0	0	0

0	0	0	0	0	0	0,001	0	0	0
0	0	0	0	0	0	0	0,001	0	0
0	0	0	0	0	0	0	0	0,001	0
0	0	0	0	0	0	0	0	0	0,001

b. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
2	1040,7706	1040,7704	1040,772	1070,771
3	1121,1793	971,8233	1121,180	971,823
4	1091,7897	927,3027	1091,791	927,304
5	1031,7879	967,9363	1031,788	967,936

c. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

- Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0,003840 m	0,003813626 m

- Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	Sx	Sy	Sx	Sy
2	0,07867	0,0554	0,078128812	0,055021124
3	0,01582	0,1251	0,015714128	0,124208023
4	0,08825	0,06058	0,087647102	0,060158359
5	0,07347	0,0595	0,072968436	0,059092185

d. Ellips Kesalahan

Koordinat Titik		Borland C++ Builder	Microsoft Excel
2	t	61,84373444 61° 50' 37,44400146"	61° 50' 35",259531
	Su(±)	0,08626	0,08567014
	Sv (±)	0,04263	0,0423

3	t	0,897353290 0° 53' 50,47184562"	0° 53' 50",100175
	Su(±)	0,1251	0,124223013
	Sv (±)	0,01570	0,015595175
4	t	58,50962266 58° 30' 34,641583"	58° 30' 44",107137
	Su(±)	0,1013	0,10060575
	Sv (±)	0,03458	0,034344223
5	t	36,59414794 36° 35' 38,93259637"	36° 35' 31",73942
	Su(±)	0,08764	0,061544055
	Sv (±)	0,03547	0,0352

○ Hasil Perhitungan tanpa Matriks P (Matriks P dianggap 1)

a. Koordinat X dan Y

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	X (m)	Y (m)	X (m)	Y (m)
2	1040,7706	1040,7704	1040,772	1040,771
3	1121,1793	971,8233	1121,180	971,823
4	1091,7897	927,3027	1091,791	927,304
5	1031,7879	967,9363	131,788	967,936

b. Standard Deviasi dan Standard Deviasi Tiap Titik Koordinat

• Standard Deviasi

So	Borland C++ Builder	Microsoft Excel
	0,003840 m	0,00381363 m

• Standard Deviasi Tiap Titik Koordinat

Koordinat Titik	Borland C++ Builder		Microsoft Excel	
	Sx	Sy	Sx	Sy
2	0,002488	0,001752	0,00247065	0,00173992

3	0,0005003	0,003955	0,00049692	0,0039278
4	0,002791	0,001916	0,00277164	0,00190237
5	0,002323	0,001882	0,00230746	0,00186866

c. Ellips Kesalahan

Koordinat Titik		Borland C++ Builder	Microsoft Excel
2	t	61,84373444 61° 50' 37,444001"	61° 50' 35",259738
	Su(±)	0,002728	0,00270913
	Sv (±)	0,001348	1,34E-03
3	t	0,897353291 0° 53' 50,471848"	0° 53' 50",100178
	Su(±)	0,003955	0,00392828
	Sv (±)	0,0004965	4,93E-04
4	t	58,50962269 58° 30' 34,641685"	58° 30' 44",107292
	Su(±)	0,003203	0,00318143
	Sv (±)	0,001094	0,00108606
5	t	36,59414796 36° 35' 38,9326867"	36° 35' 31",731804
	Su(±)	0,002771	0,00194619
	Sv (±)	0,001122	1,11E-03

Ketelitian linier jarak untuk Poligon Tertutup baik menggunakan sudut dalam maupun sudut luar adalah sama yaitu 1 : 69330.

Program Penghitungan Poligon dengan Menggunakan Metode Perataan Parameter ini telah diuji coba dan lolos pada test β_1 dan test β_2 . Test β_1 (*Test Laboratoris Ideal*) adalah test program dengan menggunakan data-data yang ideal

definitif, bukan berasal dari data ukuran. Maksud test β_1 adalah untuk menguji “*running*” program apakah sudah sesuai dengan algoritma. Indikasi dari program yang sudah benar, hasil akhir berupa angka-angka koreksi (v) nol atau mendekati nol. Sedangkan test β_2 adalah test program dengan menggunakan data ukuran lapangan.

Analisa hasil hitung perataan parameter poligon pada poligon terbuka terikat sempurna dan poligon tertutup (sudut dalam dan sudut luar) dengan atau tanpa matriks bobot (matriks P) menggunakan program Borland C++ Builder dan Microsoft Excel adalah sebagai berikut :

- a. Hasil hitung perataan parameter pada poligon menggunakan matriks bobot (matriks P) akan tergantung pada harga relatif dari bobot semua data ukuran yang dimasukkan pada matriks P. Data ukuran yang lebih teliti akan mempunyai bobot yang lebih besar dibandingkan dengan data ukuran yang kurang teliti. Sehingga hasil perhitungan yang nilainya matriks P besar (mendekati 1) akan relatif sama dengan hasil perhitungan yang tidak memasukkan matriks P (matriks P dianggap 1). Namun demikian, pada dasarnya perbedaan antara hasil perhitungan yang menggunakan matriks P dan yang tidak menggunakan matriks P relatif kecil sehingga tidak terlalu mempengaruhi hasil perhitungan.
- b. Hasil perhitungan dengan menggunakan Borland C++ Builder lebih teliti dibandingkan dengan hasil perhitungan menggunakan Microsoft Excel. Hal ini karena jumlah desimal pada Borland C++ Builder lebih besar dibandingkan Microsoft Excel.

BAB V

KESIMPULAN DAN SARAN

V.1. KESIMPULAN

Kesimpulan yang dapat diambil dari hasil penelitian ini adalah sebagai berikut :

1. Paket Program Hitung Perataan Parameter untuk Poligon yang dibuat dengan menggunakan perangkat lunak Borland C++ Builder ini mempunyai kemampuan untuk :
 - a. Menghitung koordinat X,Y dengan standard deviasi tiap titik koordinat pada poligon terbuka terikat sempurna dan poligon tertutup dengan sudut dalam maupun sudut luar menggunakan perataan parameter;
 - b. Menghitung Ellips Kesalahan dari hasil perhitungan yang dihasilkan pada poligon terbuka terikat sempurna dan poligon tertutup dengan sudut dalam maupun sudut luar;
 - c. Menyimpan, memanggil, dan memproses kembali data yang diinputkan.
2. Ketelitian data pengukuran (sudut horisontal, jarak horisontal, dan azimuth) sangat mempengaruhi hasil hitung perataan parameter poligon baik pada poligon terbuka terikat sempurna maupun poligon tertutup.

3. Program Penghitungan Poligon dengan Menggunakan Metode Perataan Parameter ini telah diuji coba dan lolos pada test β_1 (*Test Laboratoris Ideal*) dan test β_2 (test dengan data lapangan).
4. Hasil hitung perataan parameter pada poligon menggunakan matriks bobot (matriks P) akan tergantung pada harga relatif dari bobot semua data ukuran yang dimasukkan pada matriks P.
5. Hasil perhitungan menggunakan Borland C++ Builder lebih teliti dibandingkan hasil perhitungan menggunakan Microsoft Excel karena jumlah desimal pada Borland C++ Builder lebih besar dibandingkan Microsoft Excel. Hal ini berdasarkan rata-rata perbedaan hasil hitungan poligon menggunakan data ukuran (β_2) yaitu sebagai berikut :

a. Poligon Terbuka Terikat Sempurna

- *Azimuth dihitung*

Apabila menggunakan matriks P, rata-rata perbedaan untuk hasil koordinat $X = -0,000045$ m dan $Y = 0,000365$ m sedangkan standard deviasi tiap titik koordinat $X = -0,00199$ m dan $Y = -0,00228$ m. Jika tanpa matriks P (matriks P dianggap 1), rata-rata perbedaan untuk hasil koordinat $X = -0,000045$ m dan $Y = 0,000365$ m sedangkan standard deviasi tiap koordinat $X = -0,000212755$ m dan $Y = -0,000223565$ m. Untuk ketelitian liniernya adalah 1 : 680200

- *Diketahui azimuth*

Apabila menggunakan matriks P, maka rata-rata perbedaan untuk hasil koordinat $X = -0,00251667$ m dan $Y = 0,006$ m. Jika tanpa

menggunakan matriks P (matriks P dianggap 1), maka rata-rata perbedaan untuk hasil koordinat $X = 0,002517$ m dan $Y = 0,006$ m. Untuk ketelitian liniernya adalah 1 : 141700

b. Poligon Tertutup

- *Sudut Dalam*

Apabila menggunakan matriks P, rata-rata perbedaan untuk hasil koordinat $X = -0,000875$ m dan $Y = -0,000325$ m sedangkan standard deviasi tiap titik koordinat $X = 0,000435$ m dan $Y = 0,000498$. Jika tanpa matriks P (matriks P dianggap 1), rata-rata perbedaan untuk hasil koordinat $X = -0,000875$ m dan $Y = -0,000325$ m sedangkan standard deviasi tiap koordinat $X = 0,00013379$ m dan $Y = 0,0000158$ m.

- *Sudut Luar*

Apabila menggunakan matriks P, rata-rata perbedaan untuk hasil koordinat $X = -0,000875$ m dan $Y = -0,000325$ m sedangkan standard deviasi tiap titik koordinat $X = 0,0004378$ m dan $Y = 0,000525$ m. Jika tanpa matriks P (matriks P dianggap 1), rata-rata perbedaan untuk hasil koordinat $X = -0,000875$ m dan $Y = -0,000325$ m sedangkan standard deviasi tiap koordinat $X = 0,0000139$ m dan $Y = 0,00001656$ m.

Pada Poligon Tertutup baik menggunakan sudut dalam maupun sudut luar, ketelitian linier jaraknya adalah sama yaitu 1 : 69330.

V.2. SARAN

Saran yang dapat diberikan penulis berkaitan dengan proses pengolahan data selama penelitian berlangsung, yaitu :

1. Data ukuran (sudut dan jarak ukuran) sebaiknya dilengkapi dengan bobot pengukuran agar diketahui tingkat ketelitian data ukuran.
2. Untuk pengukuran dengan kondisi area yang diukur tidak terlalu luas dan titik-titiknya terlihat, maka penggunaan metode poligon lebih dikedepankan. Hal ini dilihat dari segi efisiensi dan ekonomis penggunaan alat.

DAFTAR PUSTAKA

1. Joyosumarto, S., 1993, *Pengantar Hitungan Geodesi*, PT. Petrakonsulindo Utama, Yogyakarta
2. Djawahir, 1993, *Kerangka Kontrol Horizontal*, PT. Petrakonsulindo Utama, Yogyakarta
3. Wolf, P.R., 1981, *Adjustment Computation*, Second Edition, PBL Publishing Co, Wisconsin
4. Aprianto, A., 1995, *Studi tentang Perataan Traverse Metode Parameter dan Metode Crandall dengan Variasi Ketelitian Jarak*, Laporan Tugas Akhir Jurusan Teknik Geodesi FTSP ITN Malang
5. Kusniah, A., 2003, *Kajian Ketelitian Hasil Perataan Jaringan Triangulasi dan Trilaterasi (Simulasi Sisi Pendek)*, Laporan Tugas Akhir Jurusan Teknik Geodesi FTSP ITN Malang
6. I Komang Ari, 2004, *Perataan Jaringan Kontrol Geodesi 3D dengan Kombinasi Pengukuran antara GPS dan EDM-Theodolite*, Laporan Tugas Akhir Jurusan Teknik Geodesi FTSP ITN Malang
7. P., Djoko, 1998, *Mudah Menguasai C++ Builder*, Jilid II, PT. Elex Media Komputindo, Jakarta
8. Wahana Komputer, 2004, *Pengembangan Perangkat Lunak Simulasi dengan C++ Builder*, Edisi I, Penerbit Andi, Yogyakarta
9. Kadir, A., 1995, *Pemrograman C++*, Edisi II, Penerbit Andi, Yogyakarta

LAMPIRAN

Petunjuk Penggunaan Program

Penghitungan Poligon Dengan Metode Parameter

Deskripsi Program

Program Penghitungan Poligon dengan Metode Parameteri digunakan untuk menghitung koordinat X dan Y, standar deviasi tiap koordinat titik, dan ellips kesalahan pada poligon terbuka terikat sempurna maupun poligon tertutup.

Program ini juga dilengkapi dengan fasilitas untuk mencetak hasil perhitungan, mengganti tulisan (font), mengganti warna background maupun foreground bidang gambar.

Langkah-langkah penggunaan program ini adalah sebagai berikut :

1. Install software C++ Builder 4 ke Drive C
2. Kemudian buka Folder Program dan klik Polygon.exe
3. Untuk melanjutkan, klik Lanjut pada Form Depan (Cover) kemudian masuk Form Menu Utama
4. Pada Menu Utama terdapat empat menu yaitu Menu File, Menu Hasil Perhitungan, Menu Option, dan Menu Help
5. Klik Menu File
 - 5.1 Pilih New, untuk memasukkan data baru dan dipilih sesuai dengan tipe file yang akan dibuka.
 - Klik Poligon Terbuka Terikat Sempurna, maka akan muncul

form input data Poligon Terbuka Terikat Sempurna. Masukkan data sudut mendatar, jarak mendatar, 2 koordinat awal pengukuran, 2 koordinat akhir pengukuran, dan jumlah titik yang dicari. Apabila diketahui azimuth, maka masukkan data koordinat BM1 dan koordinat BM4, azimuth awal dan azimuth akhir, jarak mendatar, sudut mendatar, dan jumlah titik yang dicari. Klik tombol hitung dan standard deviasi untuk proses perhitungannya. Kemudian simpan dengan ekstensi .PTA

- Klik Poligon Tertutup, maka akan muncul form input data Poligon Tertutup. Masukkan data sudut mendatar, jarak mendatar, koordinat awal, azimuth awal, jumlah titik, dan tipe sudut. Klik tombol hitung dan standard deviasi untuk proses perhitungannya. Kemudian simpan dengan ekstensi .PTB

Catatan : Input azimuth awal dari titik awal ke titik berikutnya searah dengan jarum jam

- 5.2 Pilih Open, untuk membuka file data poligon yang ada atau yang sudah disimpan yaitu .PTA untuk Poligon Terbuka Terikat Sempurna dan .PTB untuk Poligon Tertutup. Klik Poligon Terbuka Terikat Sempurna atau Poligon Tertutup, maka akan muncul Gambar Sket Poligon dan data hasil
- 5.3 Pilih Open as Bitmap, untuk membuka file bitmap gambar poligon
- 5.4 Pilih Save as Bitmap, untuk menyimpan gambar poligon yang ada ke dalam bidang gambar

5.5 Pilih Print, untuk mencetak gambar sket poligon

5.6 Pilih Setting Print, untuk mensetting printer mulai dari jenis printer yang digunakan, kertas, dan lain-lain sebelum dicetak

5.7 Pilih Exit, untuk keluar dari program

6. Klik Menu Hasil Perhitungan

6.1 Pilih Poligon Terbuka, untuk melihat hasil perhitungan dari data Poligon Terbuka Terikat Sempurna

6.2 Pilih Poligon Tertutup, untuk melihat hasil perhitungan dari data Poligon Tertutup dengan sudut dalam maupun sudut luar

7. Klik Menu Option

7.1 Pilih Font, untuk memilih jenis tulisan yang akan digunakan

7.2 Pilih Background Color, untuk memilih warna background/latar belakang dari bidang gambar

7.3 Pilih Foreground Color, untuk memilih warna pena yang digunakan

8. Klik Menu Help

8.1 Pilih Petunjuk Penggunaan, untuk mengetahui cara menggunakan program

8.2 Pilih About, untuk mengetahui keterangan-keterangan tentang program

LISTING PROGRAM

FORM SPLASH (FORM COVER)

```
//-----
#include <vcl.h>
#pragma hdrstop
#include "Splash.h"
#include "FMain.h"
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TSplashForm *SplashForm;
//-----
__fastcall TSplashForm::TSplashForm(TComponent* Owner)
    : TForm(Owner)
{
}
//-----
void __fastcall TSplashForm::Button1Click(TObject *Sender)
{
    MainForm->Show();
}
//-----
void __fastcall TSplashForm::Button2Click(TObject *Sender)
{
    Close();
}
//-----
```

MENU UTAMA

```
//-----
#include <vcl.h>
#pragma hdrstop
#include "FMain.h"
#include "stdio.h";
#include "Hasil.h"
#include "Hasil.h"
#include "About.h"
#include "stdlib.h";
#include "Math.h";
#include <printers.hpp>
//-----

#pragma package(smart_init)
#pragma resource "*.dfm"
TMainForm *MainForm;
bool Drawing;
int PointCount,MaxPoint;
TColor ForeColor,BackColor;
TFont *ActiveFont;

double MinValue(const int * Data, const int Data_Size)
{
    int i,MinVal=1000000000000;
    for(i=0;i<Data_Size;i++)
```

```

    if (Data[i]<MinVal)MinVal=Data[i];
    return MinVal;
}
double StrToSudut(char *Derajat,char *Menit,char *Detik)
{
    double hasil;
    hasil=atof(Derajat)+atof(Menit)/60+atof(Detik)/3600;
    return hasil;
}
double ConverToSudut(double Derajat,double Menit,double Detik)
{
    double hasil;
    hasil=Derajat+(Menit/60)+(Detik/3600);
    return hasil;
}

//-----
__fastcall TMainForm::TMainForm(TComponent* Owner)
    : TForm(Owner)
{

}
//-----
void __fastcall TMainForm::Image1MouseMove(TObject *Sender,
    TShiftState Shift, int X, int Y)
{
    StatusBar1->Panels->Items[0]->Text="X:"+IntToStr(X)+" Y:"+IntToStr(Y);
}
//-----
void TMainForm::ClearCanvas()
{
    Image1->Canvas->Brush->Color=BackColor;
    Image1->Canvas->FillRect(Rect(0,0,Image1->Width,Image1->Height));
}
void TMainForm::DrawPolygons(TPoint Poly[]){
    ClearCanvas();
}
void TMainForm::DrawPoints(int x,int y,int width,TColor Color)
{
    Canvas->Pen->Color=Color;
    Canvas->Pie(x-width,y-width,x+width,y+width,1,1,1,1);
}
void TMainForm::OpenOpenPolygons(char *FileName)
{
    FILE *stream;
    char Posisi[200];
    int i,lenth;
    double skala;
    int PosX,PosY,MinX,MinY,MaxX,MaxY;
    int KoorX[20],KoorY[20],XWidth,YHeight;
    TPoint Pt[20];
    if ((stream = fopen(FileName, "r+")) == NULL)

```

```

{
    MessageBox("Error:Open File 'filename'",mtError,TMsgDlgButtons() << mbOK,0);
}
i=0;
while (!feof(stream))
{
    fscanf(stream,"%s\n",Posisi);
    PosX=ceil(atof(Posisi));
    fscanf(stream,"%s\n",Posisi);
    PosY=ceil(atof(Posisi));

    Pt[i].x=PosX;
    Pt[i].y=PosY;
    KoorX[i]=PosX;
    KoorY[i]=PosY;
    i++;
}

lenth=i;
MinX=KoorX[lenth-1];
MinY=KoorY[lenth-1];
MaxX=KoorX[lenth-1];
MaxY=KoorY[lenth-1];
for (i=0;i<lenth;i++){

    if (KoorX[i]<MinX)MinX=KoorX[i];
    if (KoorY[i]<MinY)MinY=KoorY[i];
    if (KoorX[i]>MaxX)MaxX=KoorX[i];
    if (KoorY[i]>MaxY)MaxY=KoorY[i];
}

//skala=Image1->Height/(MaxY-MinY);
XWidth=(int)(MaxX-MinX);
YHeight=(int)(MaxY-MinY);

skala=2;
if (YHeight>Image1->Height)skala=0.5;
ClearCanvas();
Image1->Canvas->Pen->Color=ForeColor;
Image1->Canvas->TextOutA(10,20,"Sket Poligon Terbuka Terikat Sempurna");

for (i=0;i<lenth;i++)
{
    Pt[i].x=(Pt[i].x-MinX)*skala+20;
    Pt[i].y=Image1->Height-(Pt[i].y-MinY)*skala-20;

    if (i==0){
        Image1->Canvas->MoveTo(Pt[i].x,Pt[i].y);} else
    {Image1->Canvas->LineTo(Pt[i].x,Pt[i].y); }
    Image1->Canvas->Pie(Pt[i].x-2,Pt[i].y-2,Pt[i].x+2,Pt[i].y+2,0,0,0,0);
}

Image1->Canvas->TextOutA(Pt[0].x+10,Pt[0].y-8,"BM1");
Image1->Canvas->TextOutA(Pt[1].x+10,Pt[1].y-8,"BM2");

```

```

int huruf;
for (i=2;i<lenth-2;i++){
huruf=63+i;
Image1->Canvas->TextOutA(Pt[i].x+10,Pt[i].y-8,(char)huruf);
}
Image1->Canvas->TextOutA(Pt[lenth-1].x+10,Pt[lenth-1].y-8,"BM4");
Image1->Canvas->TextOutA(Pt[lenth-2].x+10,Pt[lenth-2].y-8,"BM3");
fclose(stream);
}
void TMainForm::OpenClosePolygons(char *FileName)
{
FILE *stream;
char Posisi[200];
int i,lenth;
double skala;
int PosX,PosY,MinX,MinY,MaxX,MaxY,XWidth,YHeight;
TTitik Pt[20];
int KoorX[20],KoorY[20];
ClearCanvas();
if ((stream = fopen(FileName, "rt")) == NULL)
{
    MessageDlg("Error:Open File 'filename'",mtError,TMsgDlgButtons() << mbOK,0);
}
i=0;
while (!feof(stream))
{
    fscanf(stream,"%s\n",Posisi);
    PosX=ceil(atof(Posisi));
    fscanf(stream,"%s\n",Posisi);
    PosY=ceil(atof(Posisi));
    Pt[i].x=PosX;
    Pt[i].y=PosY;
    KoorX[i]=PosX;
    KoorY[i]=PosY;
    i++;
}
lenth=i;
MinX=KoorX[lenth-1];
MinY=KoorY[lenth-1];
MaxX=KoorX[lenth-1];
MaxY=KoorY[lenth-1];
for (i=0;i<lenth;i++){
    if (KoorX[i]<MinX)MinX=KoorX[i];
    if (KoorY[i]<MinY)MinY=KoorY[i];
    if (KoorX[i]>MaxX)MaxX=KoorX[i];
    if (KoorY[i]>MaxY)MaxY=KoorY[i];
}
skala=2;
XWidth=(int)(MaxX-MinX);
YHeight=(int)(MaxY-MinY);
if (YHeight>Image1->Height)skala=0.5;
ClearCanvas();
Image1->Canvas->Pen->Color=ForeColor;
Image1->Canvas->TextOutA(10,20,"Sket Poligon Tertutup");
for (i=0;i<lenth;i++)

```

```

{
Pt[i].x=(Pt[i].x-MinX)*skala+20;
Pt[i].y=Image1->Height-(Pt[i].y-MinY)*skala-20;
if (i==0){
Image1->Canvas->MoveTo(Pt[i].x,Pt[i].y);} else
{Image1->Canvas->LineTo(Pt[i].x,Pt[i].y); }
Image1->Canvas->Pie(Pt[i].x-2,Pt[i].y-2,Pt[i].x+2,Pt[i].y+2,0,0,0,0);
}
Image1->Canvas->LineTo(Pt[0].x,Pt[0].y);
for (i=0;i<lenth;i++)
Image1->Canvas->TextOutA(Pt[i].x+10,Pt[i].y-8,IntToStr(i+1));
fclose(stream);
}
void __fastcall TMainForm::FormCreate(TObject *Sender)
{
Drawing=false;
PointCount=0;
Application->HelpFile=ExtractFilePath(Application->ExeName)+"helpoly.hlp";
}
//-----

void __fastcall TMainForm::Exit1Click(TObject *Sender)
{
if (MessageDlg("Are you sure to Close the program ?",
mtConfirmation,TMsgDlgButtons() << mbYes<< mbNo,0)==mrYes){
Close();
}
}
//-----

void __fastcall TMainForm::FormPaint(TObject *Sender)
{
ForeColor=clBlack;
BackColor=clWhite;
ClearCanvas();
//StatusBar1->Repaint();
}
//-----

void __fastcall TMainForm::SettingPrinter1Click(TObject *Sender)
{
PrinterSetup1->Execute();
}
//-----

void __fastcall TMainForm::BackgroundColor1Click(TObject *Sender)
{
if (ColorDialog1->Execute()){
BackColor=ColorDialog1->Color;
ClearCanvas();
StatusBar1->Refresh();
}
}
//-----

```

```

void __fastcall TMainForm::ForegroundColor1Click(TObject *Sender)
{
    if (ColorDialog1->Execute()){
        ForeColor=ColorDialog1->Color;
        ClearCanvas();
        StatusBar1->Refresh();
    }
}
//-----

```

```

void __fastcall TMainForm::Font1Click(TObject *Sender)
{
    if (FontDialog1->Execute()){
        ActiveFont=FontDialog1->Font;
    }
}
//-----

```

```

void __fastcall TMainForm::StatusBar1DrawPanel(TStatusBar *StatusBar,
    TStatusPanel *Panel, const TRect &Rect)
{
    StatusBar1->Canvas->Brush->Color = ForeColor;
    StatusBar1->Canvas->FillRect(Rect);
}

```

```

double TMainForm::DegToRad(double Degree)

```

```

{
    return Degree*PI/180;
}

```

```

double TMainForm::RadToDeg(double Radian)

```

```

{
    return Radian*180/PI;
}

```

```

// Mengenerate Titik-titik Azimuth hasil dari pengolahan Data Inputan

```

```

//-----
TAzimuthPoints TMainForm::ConvertToPointAzimuth1(TOpenPolygons &OP)
{
    TAzimuthPoints AP;
    return AP;
}

```

```

// Mengenerate Titik-titik Azimuth hasil dari pengolahan Data Inputan

```

```

//-----
TAzimuthPoints TMainForm::ConvertToPointAzimuth2(TClosePolygons &CP)
{
    TAzimuthPoints AP;
    return AP;
}

```

```

//-----

```

```

void TMainForm::CreateAzimuthPoint(int x, int y,int radius,
    double sudut1,double sudut2,int modeline)

```

```

{
    int x1,y1,x2,y2;
    x1=x+ceil(radius*cos(sudut1));
    y1=y-ceil(radius*sin(sudut1));
    x2=x+ceil(radius*cos(sudut2));
    y2=y-ceil(radius*sin(sudut2));
}

```

```

Image1->Canvas->MoveTo(x2,y2);
Image1->Canvas->LineTo(x,y);
Image1->Canvas->LineTo(x1,y1);
Image1->Canvas->Rectangle(x-2,y-2,x+2,y+2);
}
//-----
void __fastcall TMainForm::OpenAsBitmap1Click(TObject *Sender)
{
    char *CurrentFile;
    OpenPictureDialog1->Title="Buka File Gambar Poligon";
    if (OpenPictureDialog1->Execute())
    {
        CurrentFile = OpenPictureDialog1->FileName.c_str();
        Image1->Picture->LoadFromFile(CurrentFile);
    }
}
//-----
void __fastcall TMainForm::Save1Click(TObject *Sender)
{
    AnsiString file,ext;
    SavePictureDialog1->Title="Simpan Gambar Poligon ke dalam format Bitmap";
    if (SavePictureDialog1->Execute())
    {
        file=SavePictureDialog1->FileName;
        ext=ExtractFileExt(file);
        if (ext!=".bmp")file=file+".bmp";
        Image1->Picture->SaveToFile(file);
    }
}
//-----
void __fastcall TMainForm::FormClose(TObject *Sender, TCloseAction &Action)
{
    Application->Terminate();
}
//-----
void __fastcall TMainForm::Print1Click(TObject *Sender)
{
    PrintDialog1->Execute();
    Printer()->BeginDoc();
    Printer()->Canvas->Assign(Image1->Canvas);
    Printer()->EndDoc();
}
//-----
void __fastcall TMainForm::PetunjukPenggunaan1Click(TObject *Sender)
{
    Application->HelpJump("Home");
}
//-----
void __fastcall TMainForm::About1Click(TObject *Sender)
{
    AboutBox=new TAboutBox(Application);
    AboutBox->ShowModal();
    delete AboutBox;
}
//-----

```

```

void __fastcall TMainForm::PolygonTerbuka1Click(TObject *Sender)
{
    AnsiString Dir;
    Dir=ExtractFilePath(Application->ExeName);
    ShellExecuteA(this-
>Handle,PChar("open"),PChar("Hasil1.html"),"" PChar(Dir.c_str()),SW_SHOWNORMAL);
}
//-----
void __fastcall TMainForm::PolygonTertutup1Click(TObject *Sender)
{
    AnsiString Dir;
    Dir=ExtractFilePath(Application->ExeName);
    ShellExecuteA(this-
>Handle,PChar("open"),PChar("Hasil2.html"),"" PChar(Dir.c_str()),SW_SHOWNORMAL);
}
//-----
void __fastcall TMainForm::PoligonTerbuka1Click(TObject *Sender)
{
    InputTerbuka=new TInputTerbuka(this);
    InputTerbuka->EditState=false;
    InputTerbuka->ShowModal();
    delete InputTerbuka;
}
//-----
void __fastcall TMainForm::PoligonTertutup1Click(TObject *Sender)
{
    InputTertutup=new TInputTertutup(this);
    InputTertutup->EditState=false;
    InputTertutup->ShowModal();
    delete InputTertutup;
}
//-----
void __fastcall TMainForm::PoligonTerbuka2Click(TObject *Sender)
{
    AnsiString NamaFile;
    OpenFileDialog1->Filter="Polygon Terbuka (*.PTA)|*.PTA";
    if (OpenDialog1->Execute()){
        //OpenOpenPolygons(OpenDialog1->FileName.c_str());
        this->Caption=OpenDialog1->FileName.c_str();
        ClearCanvas();
        OpenOpenPolygons(OpenDialog1->FileName.c_str());
        NamaFile=this->Caption.SubString(1,strlen(this->Caption.c_str())-4);
        NamaFile+="*.txt";
        HasilForm=new THasilForm(this);
        HasilForm->Caption="Hasil Perhitungan Poligon Terbuka Terikat Sempurna";
        HasilForm->NamaFile=NamaFile.c_str();
        HasilForm->ShowModal();
        delete HasilForm;
    }
}
//-----
void __fastcall TMainForm::PoligonTertutup2Click(TObject *Sender)
{

```

```

AnsiString NamaFile;
OpenDialog1->Filter="Polygon Tertutup (*.PTB)|*.PTB";
if (OpenDialog1->Execute()){
this->Caption=OpenDialog1->FileName.c_str();
ClearCanvas();
OpenClosePolygons(OpenDialog1->FileName.c_str());
NamaFile=this->Caption.SubString(1,strlen(this->Caption.c_str()-4);
NamaFile+=" .txt";
HasilForm=new THasilForm(this);
HasilForm->Caption="Hasil Perhitungan Poligon Tertutup";
HasilForm->NamaFile=NamaFile.c_str();
HasilForm->ShowModal();
delete HasilForm;
}
}
//-----

```

POLIGON TERBUKA TERIKAT SEMPUKNA

```

//-----
#include <vcl.h>
#pragma hdrstop
#include "Input1.h"
#include "stdio.h"
#include "stdlib.h"
// #include "PolygonTypes.h"
#include "math.h"
#include "fstream.h"
#include "conio.h"
#include "DrwEllipse.h"
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TInputTerbuka *InputTerbuka;
int ino=0;
double Sudut[20];

void __fastcall TInputTerbuka::SimpanText(AnsiString NamaFile){
FILE *FSave;
int i,j;
AnsiString Baris,PathFile;
NamaFile+=" .txt";
FSave = fopen(NamaFile.c_str(), "w");
fprintf(FSave,"%s\n\n","=== Poligon Terbuka Terikat Sempurna ===");
fprintf(FSave,"%s\n","=== Sudut Datar ===");
for (i=0;i<SGSudut->RowCount;i++)
{
fprintf(FSave,"%s\t",SGSudut->Cells[0][i].c_str());
fprintf(FSave,"%s\t",SGSudut->Cells[1][i].c_str());
fprintf(FSave,"%s\t",SGSudut->Cells[2][i].c_str());
fprintf(FSave,"%s\n",SGSudut->Cells[3][i].c_str());
}
fprintf(FSave,"%s\n","=== Jarak ===");
for (i=0;i<SGJarak->RowCount;i++)
{
fprintf(FSave,"%s\t",SGJarak->Cells[0][i].c_str());

```

```

fprintf(FSave,"%s\n",SGJarak->Cells[1][i].c_str());
}
fprintf(FSave,"%s\n","=== Azimuth ===");
for (i=0;i<SGH->RowCount;i++)
{
fprintf(FSave,"%s\t",SGH->Cells[0][i].c_str());
fprintf(FSave,"%s\t",SGH->Cells[1][i].c_str());
fprintf(FSave,"%s\t",SGH->Cells[2][i].c_str());
fprintf(FSave,"%s\n",SGH->Cells[3][i].c_str());
}
fprintf(FSave,"%s\n","=== Koordinat Poligon===");
for (i=0;i<SGK->RowCount;i++)
{
fprintf(FSave,"%s\t",SGK->Cells[0][i].c_str());
fprintf(FSave,"%s\t",SGK->Cells[1][i].c_str());
fprintf(FSave,"%s\n",SGK->Cells[2][i].c_str());
}
fprintf(FSave,"%s\n","=== Sudut Pendekatan ===");
for (i=0;i<SGDK->RowCount;i++)
{
fprintf(FSave,"%s\t",SGDK->Cells[0][i].c_str());
fprintf(FSave,"%s\t",SGDK->Cells[1][i].c_str());
fprintf(FSave,"%s\t",SGDK->Cells[2][i].c_str());
fprintf(FSave,"%s\n",SGDK->Cells[3][i].c_str());
}
fprintf(FSave,"%s\n","=== Jarak Pendekatan===");
for (i=0;i<SGJK->RowCount;i++)
{
fprintf(FSave,"%s\t",SGJK->Cells[0][i].c_str());
fprintf(FSave,"%s\n",SGJK->Cells[1][i].c_str());
}
fprintf(FSave,"%s\n","=== Matrik F ===");
for (i=0;i<SGF->RowCount;i++)
{
fprintf(FSave,"%s\t",SGF->Cells[0][i].c_str());
fprintf(FSave,"%s\n",SGF->Cells[1][i].c_str());
}
fprintf(FSave,"%s\n","=== Matrik A ===");
char Angka[20];
for (i=0;i<SGL->RowCount;i++)
{
for (j=0;j<SGL->ColCount;j++){
strcpy(Angka,SGL->Cells[j][i].c_str());
if (Angka=="0"){
fprintf(FSave,"%s\t\t", "0");
} else
{
fprintf(FSave,"%s\t",Angka);
}
}
fprintf(FSave,"%s\n","");
}
fprintf(FSave,"%s\n","=== Matrik X ===");
for (i=0;i<SGX->RowCount;i++)
{

```

```

fprintf(FSave,"%s\t",SGX->Cells[0][i].c_str());
if (i>0)
fprintf(FSave,"%s m\n",SGX->Cells[1][i].c_str());
else
fprintf(FSave,"%s\n",SGX->Cells[1][i].c_str());
}
fprintf(FSave,"%s\n","=== Matrik V ===");
for (i=0;i<SGV->RowCount;i++)
{
fprintf(FSave,"%s\t",SGV->Cells[0][i].c_str());
fprintf(FSave,"%s\n",SGV->Cells[1][i].c_str());
}
fprintf(FSave,"So= %s\n",lbStd->Caption.c_str());
fprintf(FSave,"%s\n","=== Standard Deviasi ===");
for (i=0;i<SGD->RowCount;i++)
{
for (j=0;j<SGD->ColCount;j++)
fprintf(FSave,"%s\t",SGD->Cells[j][i].c_str());
fprintf(FSave,"%s\n","");
}
fclose(FSave);
}

int AmbilDerajat(double Angle)
{
if (Angle!=0.0) {return INT(Angle); } else return 0;
}
int AmbilMenit(double Angle)
{
double remin;
remin=(Angle-INT(Angle))*60;
return INT(remin);
}
double AmbilDetik(double Angle)
{
double remin,rede;
remin=(Angle-INT(Angle))*60;
rede=(remin-INT(remin))*60;
return rede;
}
double RadToDeg(double Radian)
{
return Radian*180/PI;
}
double DegToRad(double Degree)
{
return Degree*PI/180;
}

double StrToSudut(char *Derajat,char *Menit,char *Detik)
{
double hasil;
hasil=atof(Derajat)+atof(Menit)/60+StrToFloat(Detik)/60;
return hasil;
}

```

```

double Jaraknya(double X1,double Y1,double X2,double Y2)
{
    double jarak;
    double selisihX,selisihY;
    selisihX=(X2-X1);
    selisihY=(Y2-Y1);
    jarak=sqrt(selisihX*selisihX+selisihY*selisihY);
    return jarak;
}

void TInputTerbuka::UpdateButton(){
    btnSave->Enabled=savStat;
    btnHitung->Enabled=hitStat;
    btnStddev->Enabled=stdStat;
}

double TInputTerbuka::Azimuth(double Dx,double Dy)
{
    float Az;
    if ((Dx>0)&&(Dy>0)) //kw I
        Az = RadToDeg(atan(Dx/Dy));
    if ((Dx>0)&&(Dy<0)) //kw II
        Az = 180 + RadToDeg(atan(Dx/Dy));
    if ((Dx<0)&&(Dy<0)) //kw III
        Az = 270 - RadToDeg(atan(Dx/Dy));
    if ((Dx<0)&&(Dy>0)) //kw IV
        Az = 360 + RadToDeg(atan(Dx/Dy));
    if ((Dx==0)&&(Dy>0))
        Az = 0;
    if ((Dx>0)&&(Dy==0))
        Az = 90;
    if ((Dx==0)&&(Dy<0))
        Az = 180;
    if ((Dx<0)&&(Dy==0))
        Az = 0.75*180;

    return Az;
}

double Sudut2(double X1,double Y1,double X2,double Y2)
{
    double hasil,dx,dy;
    dx=(X2-X1);
    dy=(Y2-Y1);
    if (dy<0){hasil=(PI / 2) + atan(dx / dy);}
    else if(dy>0){hasil=(PI * 1.5) + atan(dx / dy);}
    else hasil= PI * 2;
    hasil=RadToDeg(hasil);

    return hasil;
}

double Sudut3(double X1,double Y1,double X2,double Y2,double X3,double Y3)
{
    double hasil,f1,f2;
    f1=Sudut2(X2,Y2,X3,Y3);
    f2=Sudut2(X1,Y1,X2,Y2);
    //if(f1<0) f1=360+f1;

```

```

hasil=f1-f2;
hasil=180+hasil;
if (hasil<0)hasil+=360;
if (hasil>360) hasil=hasil-360;
return hasil;
}
//-----
__fastcall TInputTerbuka::TInputTerbuka(TComponent* Owner)
: TForm(Owner)
{
}
void __fastcall TInputTerbuka::Linierisasi_Jrk(TMATRIX &Lj, int MAXLjd,int MAXLjs)
{
float dx2,dy2;
dx2 = ((Kx[1]-Edit3->Text.ToDouble())/dp[1]);
dy2 = ((Ky[1]-Edit4->Text.ToDouble())/dp[1]);
Lj[0][0]=dx2;
Lj[0][1]=dy2;
int j=0;
for (int i=2;i<MAXLjd;i++)
{
Lj[i][j]= -((Kx[i]-Kx[i-1])/dp[i]);
Lj[i][j+1]= -((Ky[i]-Ky[i-1])/dp[i]);
Lj[i][j+2]= +((Kx[i]-Kx[i-1])/dp[i]);
Lj[i][j+3]= +((Ky[i]-Ky[i-1])/dp[i]);
j+=2;
}

Lj[MAXLjd][0]= (((Ky[0]-Edit4->Text.ToDouble())/pow(dp[1],2))*206265);
Lj[MAXLjd][1]= (((Edit3->Text.ToDouble()-Kx[1])/pow(dp[1],2))*206265);
j=0;
if (MAXLjs>1)
{
Lj[MAXLjd+1][j]=((Ky[0]-Ky[1])/(pow(dp[1],2))-(Ky[0]-Edit4-
Text.ToDouble())/pow(dp[1],2))*206265;
Lj[MAXLjd+1][j+1]=((Kx[1]-Kx[0])/(pow(dp[1],2))-(Edit3->Text.ToDouble()-
Kx[0])/pow(dp[1],2))*206265;
// Lj[MAXLjd+1][j+2]= ((y2-Ky[0])/(pow(dp[0],2)))*206265;
// Lj[MAXLjd+1][j+3]= ((Kx[0]-x2)/(pow(dp[0],2)))*206265;
Lj[MAXLjd+1][j+2]= ((Ky[1]-Ky[0])/(pow(dp[1],2)))*206265;
Lj[MAXLjd+1][j+3]= ((Kx[0]-Kx[1])/(pow(dp[1],2)))*206265;
}
if (MAXLjs>2)
{
j=0;
int i= 1;
for (int k=MAXLjd+2;k<MAXLjd+MAXLjs;k++)
{
Lj[k][j]=((Ky[i]-Ky[i+1])/(pow(dp[i+1],2))-(Ky[i]-Ky[i-1])/(pow(dp[i],2)))*206265;
Lj[k][j+1]= ((Kx[i+1]-Kx[i])/(pow(dp[i+1],2))-(Kx[i-1]-Kx[i])/(pow(dp[i],2)))*206265;
Lj[k][j+2]= ((Ky[i]-Ky[i-1])/(pow(dp[i],2)))*206265;
Lj[k][j+3]= ((Kx[i-1]-Kx[i])/(pow(dp[i],2)))*206265;
Lj[k][j+4]= ((Ky[i+1]-Ky[i])/(pow(dp[i+1],2)))*206265;
Lj[k][j+5]= ((Kx[i]-Kx[i+1])/(pow(dp[i+1],2)))*206265;
i++;
}
}

```

```

        j+=2;
    }
}
}
//-----
void __fastcall TInputTerbuka::Button1Click(TObject *Sender)
{
    int i,j,jrow;
    jrow=EdTitik->Text.ToInt()+1;
    SGJarak->RowCount=jrow+1;
    if (cbJenis->ItemIndex!=1){
        SGSudut->RowCount=jrow;
        SGDK->RowCount=jrow;
        SGH->RowCount=jrow+2;
    } else {
        SGSudut->RowCount=jrow+2;
        SGDK->RowCount=jrow+2;
        SGH->RowCount=jrow+3;
    }
    SGK->RowCount=jrow;
    SGXD->RowCount=jrow;
    SGJK->RowCount=jrow+1;

    SGL->ColCount=StrToInt(EdTitik->Text)*2+1;
    SGL->RowCount=SGSudut->RowCount+SGJarak->RowCount-1;
    SGL->ColWidths[0]=40;
    SGXD->ColWidths[0]=40;

    for (i=1;i<SGL->RowCount;i++)
        for (j=1;j<SGL->ColCount;j++)
            SGL->Cells[j][i]="0";
    for (i=1;i<SGSudut->RowCount;i++){
        SGSudut->Cells[0][i]=IntToStr(i);
        SGDK->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGJarak->RowCount;i++){
        SGJarak->Cells[0][i]=IntToStr(i);
        SGJK->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGH->RowCount;i++){
        SGH->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGK->RowCount;i++){
        if (cbJenis->ItemIndex!=1){
            if (i==1){SGK->Cells[0][i]="BM2";}
            else if (i==SGK->RowCount-1){
                SGK->Cells[0][i]="BM3";
            } else {
                SGK->Cells[0][i]=(char)(63+i);
            }
        } else {
            SGK->Cells[0][i]=(char)(64+i);
        }
    }
    SGK->ColWidths[0]=50;
}

```

```

}

//-----
void __fastcall TInputTerbuka::FormCreate(TObject *Sender)
{
    SGsudut->ColWidths[0]=40;
    SGJarak->ColWidths[0]=40;
    SGsudut->Cells[0][0]="No.";
    SGsudut->Cells[1][0]="Derajat";
    SGsudut->Cells[2][0]="Menit";
    SGsudut->Cells[3][0]="Detik";
    SGJarak->Cells[0][0]="No.";
    SGJarak->Cells[1][0]="Jarak";
    SGH->Cells[0][0]="No.";
    SGH->Cells[1][0]="Derajat";
    SGH->Cells[2][0]="Menit";
    SGH->Cells[3][0]="Detik";
    SGK->Cells[0][0]="No.";
    SGK->Cells[1][0]="X";
    SGK->Cells[2][0]="Y";
    SGDk->Cells[0][0]="No.";
    SGDk->Cells[1][0]="Derajat";
    SGDk->Cells[2][0]="Menit";
    SGDk->Cells[3][0]="Detik";
    SGJK->Cells[0][0]="No.";
    SGJK->Cells[1][0]="Jarak";
    // DecimalSeparator=LOCALE_SDECIMAL;
    // ThousandSeparator=LOCALE_STHOUSAND;
}

//-----
void __fastcall TInputTerbuka::btnSaveClick(TObject *Sender)
{
    FILE *FSave;
    int i,j,baris;
    AnsiString Baris,PathFile,NamaFile;
    AnsiString Buff,OB[10];
    j=1;
    if (SaveDialog1->Execute()){
        //char Derajat[12],Menit[12],Detik[12];
        char PosX[15],PosY[12];
        FSave = fopen(SaveDialog1->FileName.c_str(), "w");
        PathFile=ExtractFilePath(SaveDialog1->FileName);
        NamaFile=ExtractFileName(SaveDialog1->FileName);
        NamaFile=PathFile+NamaFile.SubString(1,strlen(NamaFile.c_str())-4);
        SimpanText(NamaFile);
        fprintf(FSave,"%s\n",Edit1->Text.c_str());
        fprintf(FSave,"%s\n",Edit2->Text.c_str());
        if (cbJenis->ItemIndex==1){
            fprintf(FSave,"%s\n",Edit3->Text.c_str());
            fprintf(FSave,"%s\n",Edit4->Text.c_str());
        }
        for (i=1;i<SGK->RowCount;i++)

```

```

{
strcpy(PosX,SGK->Cells[1][i].c_str());
strcpy(PosY,SGK->Cells[2][i].c_str());
fprintf(FSave,"%s\n",PosX);
fprintf(FSave,"%s\n",PosY);
}
if (cbJenis->ItemIndex==1){
fprintf(FSave,"%s\n",Edit5->Text.c_str());
fprintf(FSave,"%s\n",Edit6->Text.c_str());
}
fprintf(FSave,"%s\n",Edit7->Text.c_str());
fprintf(FSave,"%s\n",Edit8->Text.c_str());
fclose(FSave);
PathFile=ExtractFilePath(Application->ExeName);
NamaFile=PathFile+"Hasil1.html";
FSave = fopen(NamaFile.c_str(), "w");
fprintf(FSave,"%s\n\n", "<html><head><title>===Poligon Terbuka Terikat Sempurna ===</title>");
fprintf(FSave,"%s\n", "<style type='text/css'>");
fprintf(FSave,"%s\n", "td {");
fprintf(FSave,"%s\n", "font-family: Verdana, Arial, Helvetica, sans-serif;");
fprintf(FSave,"%s\n", "font-size: 10px;");
fprintf(FSave,"%s\n", "color: #000066;");
fprintf(FSave,"%s\n", " </style></head>");
fprintf(FSave,"%s\n\n", "<body><center><h2>=== Hasil Perhitungan Poligon Terbuka Terikat Sempurna ===</h2></center><br>");
fprintf(FSave,"%s\n", "<table width='100%' border=1 cellpadding=4 cellspacing=0>");
fprintf(FSave,"%s", "<tr><td rowspan=2><b>Titik</b></td><td rowspan=2 align=center><b>Sudut</b></td><td align=center colspan=3><b>Hasil</b></td>");
fprintf(FSave,"%s", "<td colspan=2 align=center><b>Koordinat Pendekatan</b></td><td colspan=2 align=center><b>Matriks X</b></td><td colspan=4 align=center><b>Koordinat Definitif</b></td></tr><tr>");
fprintf(FSave,"%s", "<td align=center>&nbsp;</td><td align=center><b>Azimuth</b></td><td align=center><b>Jarak (m)</b></td><td align=center><b>X (m)</b></td><td align=center><b>Y (m)</b></td><td align=center><b>dX (m)</b></td><td align=center><b>dY (m)</b></td>");
fprintf(FSave,"%s", "<td align=center><b>X (m)</b></td><td align=center><b>Sx</b></td><td align=center><b>Y (m)</b></td><td align=center><b>Sy</b></td></tr>");
if (cbJenis->ItemIndex==1){
baris=SGH->RowCount; } else {
baris=SGJarak->RowCount;
}
for (i=1; i<baris; i++)
{
fprintf(FSave,"%s", "<tr>");
if (i<SGSudut->RowCount){
if (cbJenis->ItemIndex==1){
if (i==1)Buff="BM2";
else if (i>JumTitik+1)Buff="BM3";
else Buff=(char)(63+i);
} else {
Buff=SGK->Cells[0][i];
}
OB[i]=Buff;
fprintf(FSave,"%s", "<td>");
fprintf(FSave,"%s", Buff.c_str());
fprintf(FSave,"%s", "</td><td> ");

```

```

fprintf(FSave,"%s ",SGSudut->Cells[1][i].c_str());
fprintf(FSave," %s",&deg; ");
fprintf(FSave,"%s ",SGSudut->Cells[2][i].c_str());
fprintf(FSave,"%s"," ");
fprintf(FSave,"%s",SGSudut->Cells[3][i].c_str());
fprintf(FSave,"%s\n",&quot; </td>");
}else {
    fprintf(FSave,"%s",<td>");
    Baris="&nbsp;";
    fprintf(FSave,"%s",Baris.c_str());
    fprintf(FSave,"%s",</td><td>&nbsp;</td>");
}
if (i<baris){
    fprintf(FSave,"%s",<td>");
    if (i==1)Buff="BM1-BM2";
    if (i>1&&i<SGH->RowCount-1)Buff=OB[i-1]+ "-" +OB[i];
    if (i==baris-1)Buff=OB[i-1]+ "-" +BM4;
    fprintf(FSave,"%s",Buff.c_str());
    fprintf(FSave,"%s",</td><td>");
    fprintf(FSave,"%s\t",SGH->Cells[1][i].c_str());
    fprintf(FSave,"%s",&deg; ");
    fprintf(FSave,"%s\t",SGH->Cells[2][i].c_str());
    fprintf(FSave,"%s"," ");
    fprintf(FSave,"%s\n",SGH->Cells[3][i].c_str());
    fprintf(FSave,"%s",&quot;");
}else {
    fprintf(FSave,"%s",<td>&nbsp;</td><td>&nbsp;</td>");
}
if (i<=SGJarak->RowCount&&i>cbJenis->ItemIndex)
{
    fprintf(FSave,"%s",<td align=right>");
    fprintf(FSave,"%s",SGJarak->Cells[1][i-cbJenis->ItemIndex].c_str());
    fprintf(FSave,"%s\n",</td>");
}else {
    fprintf(FSave,"%s",<td>&nbsp;</td>");
}
if (i<=SGK->RowCount&&i>cbJenis->ItemIndex)
{
    fprintf(FSave,"%s",<td align=right>");
    fprintf(FSave,"%s",SGK->Cells[1][i-cbJenis->ItemIndex].c_str());
    fprintf(FSave,"%s",</td><td align=right>");
    fprintf(FSave,"%s",SGK->Cells[2][i-cbJenis->ItemIndex].c_str());
    fprintf(FSave,"%s",</td>");
}else {
    fprintf(FSave,"%s",<td>&nbsp;</td><td>&nbsp;</td>");
}
if (i<=SGXD->RowCount&&i>cbJenis->ItemIndex)
{
    fprintf(FSave,"%s",<td>");
    fprintf(FSave,"%s",SGX->Cells[1][j].c_str());
    fprintf(FSave,"%s",</td><td>");
    fprintf(FSave,"%s",SGX->Cells[1][j+1].c_str());
    fprintf(FSave,"%s",</td>");
    if (i%2==0)j+=2;
} else {

```

```

        fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td>");
    }
    if (i<=SGK->RowCount&&i>cbJenis->ItemIndex)
    {
        fprintf(FSave,"%s","<td>");
        fprintf(FSave,"%s",SGXD->Cells[1][i-cbJenis->ItemIndex].c_str());
        fprintf(FSave,"%s","</td><td>");
        fprintf(FSave,"%s",SGD->Cells[1][i-cbJenis->ItemIndex].c_str());
        fprintf(FSave,"%s","</td><td>");
        fprintf(FSave,"%s",SGXD->Cells[2][i-cbJenis->ItemIndex].c_str());
        fprintf(FSave,"%s","</td><td>");
        fprintf(FSave,"%s",SGD->Cells[2][i-cbJenis->ItemIndex].c_str());
        fprintf(FSave,"%s","</td>");
    }else {
        fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td>");
    }
    fprintf(FSave,"%s\\n"," </tr>");
}
fprintf(FSave,"%s","</table>");
fprintf(FSave,"%s\\n","</body></html>");
fclose(FSave);
}
}
//-----
void __fastcall TInputTerbuka::FormShow(TObject *Sender)
{
    FILE *stream;
    struct TOpenPolygons OP;
    TTitik TitikA[24];
    int i;
    double Jarak[20];
    if (EditState==true){
        if ((stream = fopen>NamaFile.c_str(), "r")) == NULL) /* open file NameOfFile*/
        {
            MessageDlg("Error:Open File 'filename'",mtError,TMsgDlgButtons() << mbOK,0);
        }
        fseek(stream, SEEK_SET, 0);

        fread(&OP, sizeof(OP), 1, stream);
        Edit1->Text=FormatFloat("0.0000",TitikA[0].x);
        Edit2->Text=FormatFloat("0.0000",TitikA[0].y);
        Edit3->Text=FormatFloat("0.0000",TitikA[1].x);
        Edit4->Text=FormatFloat("0.0000",TitikA[1].y);
        Edit5->Text=FormatFloat("0.0000",TitikA[2].x);
        Edit6->Text=FormatFloat("0.0000",TitikA[2].y);
        Edit7->Text=FormatFloat("0.0000",TitikA[3].x);
        Edit8->Text=FormatFloat("0.0000",TitikA[3].y);
        EdTitik->Text=IntToStr(JumTitik);
        SGsudut->RowCount=JumTitik+3;
        SGJarak->RowCount=JumTitik+2;
        for (i=0;i<SGsudut->RowCount;i++){
            SGsudut->Cells[1][i+1]=FormatFloat("0.0000",Sudut[i]);
            //SGsudut->Cells[2][i+1]=IntToStr(Sudut[i]);
            //SGsudut->Cells[3][i+1]=IntToStr(Sudut[i]);
        }
    }
}

```

```

for (i=0;i<SGJarak->RowCount;i++){
    SGJarak->Cells[1][i+1]=FloatToStr(Jarak[i]);
}
fclose(stream);
}
}
//-----

void TInputTerbuka::HitungTanpaAzimuth()
{
    int i,j;
    double PosX,PosY;
    //struct TOpenPolygons OP;
    double Selisih;

    TitikA[0].x=Edit1->Text.ToDouble();
    TitikA[0].y=Edit2->Text.ToDouble();
    TitikA[1].x=Edit3->Text.ToDouble();
    TitikA[1].y=Edit4->Text.ToDouble();
    TitikA[2].x=Edit5->Text.ToDouble();
    TitikA[2].y=Edit6->Text.ToDouble();
    TitikA[3].x=Edit7->Text.ToDouble();
    TitikA[3].y=Edit8->Text.ToDouble();

    JumTitik=EdTitik->Text.ToInt();
    //Memasukkan Data Sudut yang di entri melalui StringGrid SGsudut
    for (i=0;i<SGSudut->RowCount;i++){
        Sudut[i]=StrToSudut(SGSudut->Cells[1][i+1].c_str(),
            SGSudut->Cells[2][i+1].c_str(),
            SGSudut->Cells[3][i+1].c_str());
    }
    for (i=1;i<SGJarak->RowCount;i++){
        Jarak[i]=atof(SGJarak->Cells[1][i].c_str());
    }
    int akhir;
    akhir=SGH->RowCount-1;
    //Menghitung azimuth
    azm[0]=Azimuth(Edit3->Text.ToDouble()-Edit1->Text.ToDouble(),
        Edit4->Text.ToDouble()-Edit2->Text.ToDouble());
    azm[akhir]=Azimuth(Edit7->Text.ToDouble()-Edit5->Text.ToDouble(),
        Edit8->Text.ToDouble()-Edit6->Text.ToDouble());
    /*
    azm[0]=StrToSudut(Edit9->Text.c_str(),Edit10->Text.c_str(),Edit11->Text.c_str());
    azm[akhir]=StrToSudut(Edit12->Text.c_str(),Edit13->Text.c_str(),Edit14->Text.c_str());
    SGH->RowCount=EdTitik->Text.ToInt()+4;;
    */
    for (i=1;i<SGH->RowCount-2;i++){
        azm[i]=azm[i-1]+Sudut[i-1]-180;
        if (azm[i]<0)azm[i]+=360;
        if (azm[i]>360)azm[i]-=360;
    }
    //azm[SGH->RowCount-1]=Azimuth(Edit7->Text.ToDouble()-Edit5->Text.ToDouble(),
    //Edit8->Text.ToDouble()-Edit6->Text.ToDouble());
    //memasukkan nilai azimuth kedalam stringgrid SGH

```

```

for (i=0;i<SGH->RowCount;i++)
{
    SGH->Cells[1][i+1]=IntToStr(AmbilDerajat(azm[i]));
    SGH->Cells[2][i+1]=IntToStr(AmbilMenit(azm[i]));
    SGH->Cells[3][i+1]=FormatFloat("0.##00",AmbilDetik(azm[i]));
}
//inisialisasi awal nilai koordinat
PosX=Edit3->Text.ToDouble();
PosY=Edit4->Text.ToDouble();
Koord[0].x=PosX;Koord[0].y=PosY;

//Menghitung Titik Koordinat Poligon
if (chkIter->Checked){
    for (i=1;i<SGXD->RowCount;i++)
    {
        Koord[i].x=StrToFloat(SGXD->Cells[1][i].c_str());
        Koord[i].y=StrToFloat(SGXD->Cells[2][i].c_str());
    }
    Koord[SGXD->RowCount].x=Koord[SGXD->RowCount-1].x+Jarak[JumTitik+1]*sin(DegToRad(azm[JumTitik]));
    Koord[SGXD->RowCount].y=Koord[SGXD->RowCount-1].y+Jarak[JumTitik+1]*cos(DegToRad(azm[JumTitik]));
    ino++;
} else {

    for (i=1;i<SGH->RowCount;i++)
    {
        PosX=PosX+Jarak[i]*sin(DegToRad(azm[i]));
        Koord[i].x=PosX;
        SGK->Cells[1][i]=FormatFloat("0.##00",PosX);
        PosY=PosY+Jarak[i]*cos(DegToRad(azm[i]));
        Koord[i].y=PosY;
        SGK->Cells[2][i]=FormatFloat("0.##00",PosY);
    }
}
lbNo->Caption=IntToStr(ino);

//Menghitung Jarak Pendekatan
dp[1]=Jaraknya(TitikA[1].x,TitikA[1].y,Koord[1].x,Koord[1].y);
SGJK->Cells[1][1]=FormatFloat("0.##00",dp[1]);
for (i=2;i<SGK->RowCount;i++)
{
    dp[i]=Jaraknya(Koord[i-1].x,Koord[i-1].y,Koord[i].x,Koord[i].y);
    SGJK->Cells[1][i]=FormatFloat("0.##00",dp[i]);
}
dp[SGK->RowCount]=Jaraknya(Koord[JumTitik].x,Koord[JumTitik].y,
TitikA[2].x,TitikA[2].y);
SGJK->Cells[1][SGJarak->RowCount-1]=FormatFloat("#.##00",dp[SGK->RowCount]);
//Menghitung sudut pendekatan
sd[0]=Sudut3(Edit1->Text.ToDouble(),Edit2->Text.ToDouble(),Edit3->Text.ToDouble(),Edit4->Text.ToDouble(),Koord[1].x,Koord[1].y);
sd[1]=Sudut3(Edit3->Text.ToDouble(),Edit4->Text.ToDouble(),Koord[1].x,Koord[1].y,Koord[2].x,Koord[2].y);
for (i=2;i<JumTitik;i++)
{

```

```

sd[i]=Sudut3(Koord[i-1].x,Koord[i-1].y,Koord[i].x,Koord[i].y,Koord[i+1].x,Koord[i+1].y);
}
sd[JumTitik]=Sudut3(Koord[JumTitik-1].x,Koord[JumTitik-1].y,
    Koord[JumTitik].x,Koord[JumTitik].y,
    Edit5->Text.ToDouble(),Edit6->Text.ToDouble());
sd[JumTitik+1]=Sudut3(Koord[JumTitik].x,Koord[JumTitik].y,Edit5->Text.ToDouble(),Edit6-
>Text.ToDouble(),
    Edit7->Text.ToDouble(),Edit8->Text.ToDouble());
for (i=0;i<SGDK->RowCount;i++)
{
    SGDK->Cells[1][i+1]=IntToStr(AmbilDerajat(sd[i]));
    SGDK->Cells[2][i+1]=IntToStr(AmbilMenit(sd[i]));
    SGDK->Cells[3][i+1]=FormatFloat("0.##00",AmbilDetik(sd[i]));
}
SGF->RowCount=S6Sudut->RowCount+S6Jarak->RowCount-1;
SGF->Cells[1][0]="Selisih";
for (i=1;i<SGF->RowCount;i++)
{
    if (i<S6Jarak->RowCount){
        SGF->Cells[0][i]="Jarak -"+IntToStr(i);
        Selisih=Jarak[i]-dp[i];
    } else {
        SGF->Cells[0][i]="Sudut -"+IntToStr(i-S6Jarak->RowCount+1);
        Selisih=Sudut[i-S6Jarak->RowCount]-sd[i-S6Jarak->RowCount];
    }
    SGF->Cells[1][i]=FormatFloat("0.000E+00",Selisih);
}
j=1;
for (i=1;i<SGK->RowCount;i++)
{
    SGL->Cells[j][0]=FormatFloat("X 0",i);
    SGL->Cells[j+1][0]=FormatFloat("Y 0",i);
    j+=2;
}
double deltaX,deltaY,d1[20][50];
int k;
int km=JumTitik;
j=0;i=1;
//Linierisasi_Jrk(Lj,S6Jarak->RowCount-1,S6Sudut->RowCount-1);
for (k=1;k<S6Sudut->RowCount;k++)
{
    if (k==1){
        d1[i][k]=((Koord[1].y-Edit4->Text.ToDouble())/pow(dp[1],2))*206265;
        d1[i+1][k]=((Edit3->Text.ToDouble()-Koord[1].x)/pow(dp[1],2))*206265;
        d1[i+2][k]=0.0;
        d1[i+3][k]=0.0;
        d1[i+4][k]=0.0;
        d1[i+5][k]=0.0;
    }else if (k==2){
        d1[i][k]=(((Koord[1].y-Koord[2].y)/pow(dp[2],2))-((Koord[1].y-StrToFloat(Edit4-
>Text.c_str0))/pow(dp[1],2)))*206265;
        d1[i+1][k]=(((Koord[2].x-Koord[1].x)/pow(dp[2],2))-((StrToFloat(Edit3->Text.c_str0)-
Koord[1].x)/pow(dp[1],2)))*206265;
        d1[i+2][k]=((Koord[2].y-Koord[1].y)/pow(dp[2],2))*206265;
        d1[i+3][k]=((Koord[1].x-Koord[2].x)/pow(dp[2],2))*206265;
    }
}

```

```

d1[i+4][k]=0.0;
d1[i+5][k]=0.0;
} else if (k=3&&k<SGK->RowCount) {
d1[i+2][k]=(((Koord[k-1].y-Koord[k].y)/pow(dp[k],2))-((Koord[k-1].y-Koord[k-2].y)/pow(dp[2],2)))*206265;
d1[i+3][k]=(((Koord[k].x-Koord[k-1].x)/pow(dp[k],2))-((Koord[k-2].x-Koord[k-1].x)/pow(dp[2],2)))*206265;
d1[i][k]=((Koord[k-2].y-Koord[k-1].y)/pow(dp[k-1],2))*206265;
d1[i+1][k]=((Koord[k-1].x-Koord[k-2].x)/pow(dp[k-1],2))*206265;
d1[i+4][k]=((Koord[k].y-Koord[k-1].y)/pow(dp[k],2))*206265;
d1[i+5][k]=((Koord[k-1].x-Koord[k].x)/pow(dp[k],2))*206265;
} else
if (k==SGK->RowCount){
d1[i][k]=0.0;
d1[i+1][k]=0.0;
d1[i+2][k]=((Koord[km-1].y-Koord[km].y)/pow(dp[km-1],2))*206265;
d1[i+3][k]=((Koord[km].x-Koord[km-1].x)/pow(dp[km-1],2))*206265;
d1[i+4][k]=(((Koord[km].y-Edit6->Text.ToDouble())/pow(dp[km+1],2))-((Koord[km].y-Koord[km-1].y)/pow(dp[km],2)))*206265;
d1[i+5][k]=(((Edit5->Text.ToDouble()-Koord[km].x)/pow(dp[km+1],2))-((Koord[km-1].x-Koord[km].x)/pow(dp[km],2)))*206265;
} else
{
d1[i][k]=0.0;
d1[i+1][k]=0.0;
d1[i+2][k]=0.0;
d1[i+3][k]=0.0;
d1[i+4][k]=(Koord[km].y-StrToFloat(Edit6->Text.c_str()))*206265/pow(dp[km+1],2);
d1[i+5][k]=(StrToFloat(Edit5->Text.c_str()-Koord[km].x)*206265/pow(dp[km+1],2);
}
if (k%3==0)i+=2;
}
for (i=1;i<SGK->RowCount;i++)
{
if (i<SGK->RowCount){
deltaX=(Koord[i-1].x-Koord[i].x)/dp[i];
deltaY=(Koord[i-1].y-Koord[i].y)/dp[i];
} else {
deltaX=(Koord[i-1].x-Edit5->Text.ToDouble())/dp[i];
deltaY=(Koord[i-1].y-Edit6->Text.ToDouble())/dp[i];
}
if (i>1){
SGL->Cells[j-1][i]=FormatFloat("0.##00",deltaX);
SGL->Cells[j][i]=FormatFloat("0.##00",deltaY);
}
if (j<SGL->ColCount-1){
SGL->Cells[j+1][i]=FormatFloat("0.##000",-deltaX);
SGL->Cells[j+2][i]=FormatFloat("0.##000",-deltaY);
}
j+=2;
}
j=1;
for (i=1;i<SGSudut->RowCount;i++)
{

```

```

if (d1[j][i]!=0.0)
    SGL->Cells[j][i+S6K->RowCount]=FormatFloat("0.##000",d1[j][i]);
if (d1[j+1][i]!=0.0)
    SGL->Cells[j+1][i+S6K->RowCount]=FormatFloat("0.##000",d1[j+1][i]);
if (d1[j+2][i]!=0.0)
    SGL->Cells[j+2][i+S6K->RowCount]=FormatFloat("0.##000",d1[j+2][i]);
if (d1[j+3][i]!=0.0)
    SGL->Cells[j+3][i+S6K->RowCount]=FormatFloat("0.##000",d1[j+3][i]);
if (d1[j+4][i]!=0.0)
    SGL->Cells[j+4][i+S6K->RowCount]=FormatFloat("0.##000",d1[j+4][i]);
if (d1[j+5][i]!=0.0)
    SGL->Cells[j+5][i+S6K->RowCount]=FormatFloat("0.##000",d1[j+5][i]);
if (i%3==0)j+=2;
}
}

void TInputTerbuka::HitungDenganAzimuth()
{
    int i,j;
    double PosX,PosY;
    // struct TOpenPolygons OP;
    double Selisih;
    TitikA[0].x=Edit1->Text.ToDouble();
    TitikA[0].y=Edit2->Text.ToDouble();
    TitikA[1].x=Edit7->Text.ToDouble();
    TitikA[1].y=Edit8->Text.ToDouble();

    JumTitik=EdTitik->Text.ToInt();
    //Memasukkan Data Sudut yang di entri melalui StringGrid S6Sudut
    for (i=1;i<S6Sudut->RowCount;i++){
        Sudut[i]=StrToSudut(S6Sudut->Cells[1][i].c_str(),
            S6Sudut->Cells[2][i].c_str(),
            S6Sudut->Cells[3][i].c_str());
    }
    for (i=1;i<S6Jarak->RowCount;i++){
        Jarak[i]=StrToFloat(S6Jarak->Cells[1][i].c_str());
    }
    int akhir;
    akhir=S6H->RowCount-1;
    //Menghitung azimuth
    azm[0]=StrToSudut(Edit9->Text.c_str(),Edit10->Text.c_str(),Edit11->Text.c_str());
    for (i=1;i<S6H->RowCount-2;i++){
        azm[i]=azm[i-1]+Sudut[i]-180;
        if (azm[i]<0)azm[i]+=360;
        if (azm[i]>360)azm[i]-=360;
    }
    azm[akhir]=StrToSudut(Edit12->Text.c_str(),Edit13->Text.c_str(),Edit14->Text.c_str());
    //memasukkan nilai azimuth kedalam stringgrid S6H
    for (i=0;i<S6H->RowCount;i++){
        {
            S6H->Cells[1][i+1]=IntToStr(AmbilDerajat(azm[i]));
            S6H->Cells[2][i+1]=IntToStr(AmbilMenit(azm[i]));
            S6H->Cells[3][i+1]=FormatFloat("0.##00",AmbilDetik(azm[i]));
        }
    }

    //inisialisasi awal nilai koordinat
    Koord[0].x=Edit1->Text.ToDouble();

```

```

Koord[0].y=Edit2->Text.ToDouble();
PosX=Koord[0].x;
PosY=Koord[0].y;

//Menghitung Titik Koordinat Poligon
if (chkIter->Checked){
    for (i=1;i<SGXD->RowCount;i++)
    {
        Koord[i].x=StrToFloat(SGXD->Cells[1][i].c_str());
        Koord[i].y=StrToFloat(SGXD->Cells[2][i].c_str());
    }
    Koord[SGXD->RowCount].x=Koord[SGXD->RowCount-1].x+Jarak[JumTitik+1]*sin(DegToRad(azm[JumTitik]));
    Koord[SGXD->RowCount].y=Koord[SGXD->RowCount-1].y+Jarak[JumTitik+1]*cos(DegToRad(azm[JumTitik]));
    ino++;
} else {

    for (i=1;i<SGH->RowCount;i++)
    {
        PosX=PosX+Jarak[i]*sin(DegToRad(azm[i-1]));
        Koord[i].x=PosX;
        SGK->Cells[1][i]=FormatFloat("0.##00",PosX);
        PosY=PosY+Jarak[i]*cos(DegToRad(azm[i-1]));
        Koord[i].y=PosY;
        SGK->Cells[2][i]=FormatFloat("0.##00",PosY);
    }
    //Koord[JumTitik+1].x=PosX+Jarak[JumTitik+1]*sin(DegToRad(azm[JumTitik]));
    //Koord[JumTitik+1].y=PosY+Jarak[JumTitik+1]*cos(DegToRad(azm[JumTitik]));
}
lbNo->Caption=IntToStr(ino);

//Menghitung Jarak Pendekatan
for (i=1;i<SGK->RowCount;i++)
{
    dp[i]=Jaraknya(Koord[i-1].x,Koord[i-1].y,Koord[i].x,Koord[i].y);
    SGJK->Cells[1][i]=FormatFloat("0.##00",dp[i]);
}

//Menghitung sudut pendekatan
sd[1]=Sudut3(Edit1->Text.ToDouble(),Edit2->Text.ToDouble(),Koord[1].x,Koord[1].y,Koord[2].x,Koord[2].y);
for (i=2;i<JumTitik;i++)
{
    sd[i]=Sudut3(Koord[i-1].x,Koord[i-1].y,Koord[i].x,Koord[i].y,Koord[i+1].x,Koord[i+1].y);
}
for (i=1;i<JumTitik;i++)
{
    // ShowMessage(FormatFloat("0.##00",sd[i]));
    SGDK->Cells[1][i]=IntToStr(AmbilDerajat(sd[i]));
    SGDK->Cells[2][i]=IntToStr(AmbilMenit(sd[i]));
    SGDK->Cells[3][i]=FormatFloat("0.##00",AmbilDetik(sd[i]));
}
SGF->RowCount=SGSudut->RowCount+SGJarak->RowCount-1;
SGF->Cells[1][0]="Selisih";

```

```

for (i=1;i<SGJarak->RowCount;i++)
{
    SGF->Cells[0][i]="Jarak "+IntToStr(i);
    Selisih=Jarak[i]-dp[i];
    //ShowMessage(FloatToStr(Selisih));
    SGF->Cells[1][i]=FormatFloat("0.000E+00",Selisih);
}
for (i=1;i<JumTitik;i++){
    SGF->Cells[0][i+SGJarak->RowCount-1]="Sudut "+IntToStr(i);
    Selisih=Sudut[i]-sd[i];
    //ShowMessage(FloatToStr(Selisih));
    SGF->Cells[1][i+SGJarak->RowCount-1]=FormatFloat("0.000E+00",Selisih);
}
j=3;
AnsiString Buff;
for (i=1;i<JumTitik-2;i++)
{
    Buff=(char)(64+i);
    SGL->Cells[j][0]="X "+Buff;
    SGL->Cells[j+1][0]="Y "+Buff;
    j+=2;
}
SGL->Cells[1][0]="X BM2";
SGL->Cells[2][0]="Y BM2";
SGL->Cells[SGL->ColCount-2][0]="X BM3";
SGL->Cells[SGL->ColCount-1][0]="Y BM3";
double deltaX,deltaY,d1[20][50];
int k;
int km=JumTitik;
j=0;i=1;
//Linierisasi_Jrk(Lj,SGJarak->RowCount-1,SGSudut->RowCount-1);
for (k=1;k<SGSudut->RowCount;k++)
{
    if (k==1){
        d1[i][k]=(((Koord[1].y-Koord[2].y)/pow(dp[2],2))-((Koord[1].y-
Koord[0].y)/pow(dp[1],2)))*206265;
        d1[i+1][k]=(((Koord[2].x-Koord[1].x)/pow(dp[2],2))-((Koord[0].x-
Koord[1].x)/pow(dp[1],2)))*206265;
        d1[i+2][k]=((Koord[2].y-Koord[1].y)/pow(dp[2],2))*206265;
        d1[i+3][k]=((Koord[1].x-Koord[2].x)/pow(dp[2],2))*206265;
        d1[i+4][k]=0.0;
        d1[i+5][k]=0.0;
        d1[i+6][k]=0.0;
        d1[i+7][k]=0.0;
    } else if (k==2){
        d1[i][k]=((Koord[2].y-Koord[1].y)/pow(dp[2],2))*206265;
        d1[i+1][k]=((Koord[1].x-Koord[2].x)/pow(dp[2],2))*206265;
        d1[i+2][k]=(((Koord[2].y-Koord[3].y)/pow(dp[3],2))-((Koord[2].y-
Koord[1].y)/pow(dp[2],2)))*206265;
        d1[i+3][k]=(((Koord[3].x-Koord[2].x)/pow(dp[3],2))-((Koord[1].x-
Koord[2].x)/pow(dp[2],2)))*206265;
        d1[i+4][k]=((Koord[3].y-Koord[2].y)/pow(dp[3],2))*206265;
        d1[i+5][k]=((Koord[2].x-Koord[3].x)/pow(dp[3],2))*206265;
    } else if (k=3&&k<SGK->RowCount) {

```

```

d1[i][k]=((Koord[k].y-Koord[k-1].y)/pow(dp[k],2))*206265;
d1[i+1][k]=((Koord[k-1].x-Koord[k].x)/pow(dp[k],2))*206265;
d1[i+2][k]=(((Koord[k].y-Koord[k+1].y)/pow(dp[k+1],2))-((Koord[k].y-Koord[k-1].y)/pow(dp[k],2)))*206265;
d1[i+3][k]=(((Koord[k+1].x-Koord[k].x)/pow(dp[k+1],2))-((Koord[k-1].x-Koord[k].x)/pow(dp[k],2)))*206265;
d1[i+4][k]=((Koord[k+1].y-Koord[k].y)/pow(dp[k+1],2))*206265;
d1[i+5][k]=((Koord[k].x-Koord[k+1].x)/pow(dp[k+1],2))*206265;

/*else
if (k==SGK->RowCount){
d1[i][k]=0.0;
d1[i+1][k]=0.0;
d1[i+2][k]=((Koord[km].y-Koord[km+1].y)/pow(dp[km-1],2))*206265;
d1[i+3][k]=((Koord[km+1].x-Koord[km].x)/pow(dp[km-1],2))*206265;
d1[i+4][k]=(((Koord[km].y-Koord[km-1].y)/pow(dp[km+1],2))-((Koord[km+1].y-Koord[km].y)/pow(dp[km],2)))*206265;
d1[i+5][k]=(((Koord[km-1].x-Koord[km].x)/pow(dp[km+1],2))-((Koord[km].x-Koord[km+1].x)/pow(dp[km],2)))*206265;
}*/else
{
d1[i][k]=0.0;
d1[i+1][k]=0.0;
d1[i+2][k]=(Koord[km].y-Koord[km-1].y)*206265/pow(dp[km],2);
d1[i+3][k]=(Koord[km-1].x-Koord[km].x)*206265/pow(dp[km],2);
d1[i+4][k]=(((Koord[km].y-Koord[km+1].y)/pow(dp[km+1],2))-((Koord[km].y-Koord[km-1].y)/pow(dp[km],2)))*206265;
d1[i+5][k]=(((Koord[km+1].x-Koord[km].x)/pow(dp[km+1],2))-((Koord[km-1].x-Koord[km].x)/pow(dp[km],2)))*206265;
}
if (k==2)i+=2;
}
for (i=1;i<=SGK->RowCount;i++)
{
//if (i<SGK->RowCount){
deltaX=(Koord[i].x-Koord[i-1].x)/dp[i];
deltaY=(Koord[i].y-Koord[i-1].y)/dp[i];
/* } else {
deltaX=(Koord[i-1].x-Edit5->Text.ToDouble())/dp[i];
deltaY=(Koord[i-1].y-Edit6->Text.ToDouble())/dp[i];
}*/
if (i>1){
SGL->Cells[j-1][i]=FormatFloat("0.#00",deltaX);
SGL->Cells[j][i]=FormatFloat("0.#00",deltaY);
}
if (j<SGL->ColCount-1){
SGL->Cells[j+1][i]=FormatFloat("0.#000",-deltaX);
SGL->Cells[j+2][i]=FormatFloat("0.#000",-deltaY);
}
j+=2;
}
j=1;
for (i=1;i<=SGSudut->RowCount;i++)
{
if (d1[j][i]==0.0)

```

```

        SGL->Cells[j][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j][i]);
        if (d1[j+1][i]!=0.0)
        SGL->Cells[j+1][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j+1][i]);
        if (d1[j+2][i]!=0.0)
        SGL->Cells[j+2][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j+2][i]);
        if (d1[j+3][i]!=0.0)
        SGL->Cells[j+3][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j+3][i]);
        if (d1[j+4][i]!=0.0)
        SGL->Cells[j+4][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j+4][i]);
        if (d1[j+5][i]!=0.0)
        SGL->Cells[j+5][i+S6K->RowCount]=FormatFloat("0.#00000",d1[j+5][i]);
        if (i>=2)j+=2;
    }
}

void __fastcall TInputTerbuka::btnHitungClick(TObject *Sender)
{
    if (cbJenis->ItemIndex<0)cbJenis->ItemIndex=0;
    if (cbJenis->ItemIndex==1){
        HitungTanpaAzimuth();} else {
        HitungDenganAzimuth();
    }
    HitungN(Sender);
    HitungQ(Sender);
    HitungX(Sender);
    HitungKoordinat(Sender);
}

//-----

void __fastcall TInputTerbuka::Button2Click(TObject *Sender)
{
    int i,j;
    SGP->RowCount=EdKolBar->Text.ToInt()+1;
    SGP->ColCount=EdKolBar->Text.ToInt()+1;
    for (i=1;i<SGP->RowCount;i++){
        for (j=1;j<SGP->ColCount;j++){
            SGP->Cells[j][i]="0";
        }
    }
}

//-----

double TInputTerbuka::Eliminasi(TMATRIX C,int uk){
    int tanda,j,k,l,m,n;
    double konst,det;
    tanda=1;
    for (l=2;kuk;l++){
        for (m=l;m<kuk;m++){
            k=l-1;
            if (C[k][k]==0){
                konst=0;
                n=k-1;
                while ((konst==0)&&(n<kuk)){
                    n=n++;
                    konst=C[n][k];
                }
            }
        }
    }
}

```

```

    if (konst!=0){
        tanda=-tanda;
        for (j=k; j<uk; j++){
            konst=C[k][j];
            C[k][j]=C[n][j];
            C[n][j]=konst;
        }
    } else {
        C[k][k]=1;
        tanda=0;
    }
}
konst=C[m][k]/C[k][k];
for (n=k; n<uk; n++){
    C[m][n]=C[m][n]-konst*C[k][n];
}
}
}
det=tanda;
for (m=1; m<uk; m++) det=det*C[m][m];
return det;
}
void __fastcall TInputTerbuka::HitungN(TObject *Sender)
{
    int i,j,k;
    // double det;
    SGT->RowCount=SGL->ColCount;
    SGT->ColCount=SGL->RowCount;
    SGN->RowCount=SGL->ColCount;
    SGN->ColCount=SGL->ColCount;
    SGN->ColWidths[0]=40;
    for (i=1; i<SGL->RowCount; i++){
        for (j=1; j<SGL->ColCount; j++){
            AN[i][j]=StrToFloat(SGL->Cells[j][i].c_str());
            AT[j][i]=AN[i][j];
            SGT->Cells[i][j]=FloatToStr(AT[j][i]);
        }
    }
    if (chkP->Checked){
        for (i=1; i<SGP->RowCount; i++){
            for (j=1; j<SGP->ColCount; j++){
                P[i][j]=StrToFloat(SGP->Cells[j][i].c_str());
            }
        }
        for (i=1; i<SGT->RowCount; i++){
            for (j=1; j<SGP->ColCount; j++){
                PA[i][j]=0.0;
                for (k=1; k<SGT->ColCount; k++){
                    PA[i][j]=PA[i][j]+AT[i][k]*P[k][j];
                }
            }
        }
    }
    for (i=1; i<SGT->RowCount; i++){
        for (j=1; j<SGL->ColCount; j++){
            NN[i][j]=0.0;

```

```

    for (k=1;k<SGT->ColCount;k++){
        NN[i][j]=NN[i][j]+PA[i][k]*AN[k][j];
    }
    SGN->Cells[j][i]=FloatToStr(NN[i][j]);
}
}
} else {
for (i=1;i<SGL->ColCount;i++){
for (j=1;j<SGL->ColCount;j++){
    NN[i][j]=0.0;
    for (k=1;k<SGL->RowCount;k++){
        NN[i][j]=NN[i][j]+AT[i][k]*AN[k][j];
    }
    SGN->Cells[j][i]=FloatToStr(NN[i][j]);
}
}
}
//det=Eliminasi(&NN,SGL->ColCount);
// ShowMessage(" Determinan "+FloatToStr(det));
SGQ->RowCount=SGN->ColCount;
SGQ->ColCount=SGN->ColCount;
}
//-----
void __fastcall TInputTerbuka::HitungQ(TObject *Sender)
{
    int i,j,k,baris;
    double b;
    baris=SGN->RowCount-1;
    SGQ->ColCount=baris+1;
    for (i=1;i<SGN->RowCount;i++){
        for (j=1;j<SGN->ColCount;j++){
            NN[i][j]=StrToFloat(SGN->Cells[j][i].c_str());
        }
    }
    for (i=1;i<=baris;i++){
        for (j=1;j<=baris;j++){
            if (i==j) NN[i][baris+j]=1;else
                NN[i][baris+j]=0;
        }
    }
    for (i=1;i<=baris;i++){
        if (NN[i][i]==0){
            for (j=i+1;j<=baris;j++){
                if (NN[j][i]==0){
                    for (k=i;k<=2*baris;k++){
                        b=NN[i][k];
                        NN[i][k]=NN[j][k];
                        NN[j][k]=b;
                    }
                }
            }
        }
        b=NN[i][i];
        for (j=i;j<=2*baris;j++){NN[i][j]=NN[i][j]/b;
        for (j=1;j<=baris;j++){

```

```

    if (j!=i){
        for (k=i;k<=2*baris;k++){C[j][k]=NN[j][k]-NN[j][i]*NN[i][k];
        for (k=i;k<=2*baris;k++){NN[j][k]=C[j][k];
        }
    }
}
SGQ->ColWidths[0]=40;
SGQ->ColCount=baris+1;
for (i=1;i<SGQ->RowCount;i++){
    for (j=baris+1;j<=2*baris;j++){
        SGQ->Cells[j-baris][i]=FloatToStr(NN[i][j]);
        Q[i][j-baris]=NN[i][j];
    }
}
}
//-----
void __fastcall TInputTerbuka::HitungX(TObject *Sender)
{
    int i,j,k;
    SGX->ColWidths[0]=50;
    SGX->RowCount=SGQ->ColCount;
    SGX->ColCount=SGF->ColCount;
    for (i=1;i<SGF->RowCount;i++){
        for (j=1;j<SGF->ColCount;j++){
            F[i][j]=StrToFloat(SGF->Cells[j][i].c_str());
        }
    }
    if (chkP->Checked){
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGP->ColCount;j++){
                AP[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    AP[i][j]=AP[i][j]+AT[i][k]*P[k][j];
                }
            }
        }
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGF->ColCount;j++){
                AA[i][j]=0.0;
                for (k=1;k<SGP->ColCount;k++){
                    AA[i][j]=AA[i][j]+AP[i][k]*F[k][j];
                }
            }
        }
    }
    else {
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGF->ColCount;j++){
                AA[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    AA[i][j]=AA[i][j]+AT[i][k]*F[k][j];
                }
            }
        }
    }
    int h=1,baris;
    AnsiString Huruf;

```

```

baris=(SGQ->RowCount-1)/2;
for (i=1;i<SGQ->RowCount;i++){
    if (cbJenis->ItemIndex==1){
        Huruf=(char)(64+h);
    } else {
        if (h==1) Huruf="BM2";
        if (h==baris) Huruf="BM3";
        if (h>1&&h<baris) Huruf=(char)(63+h);
    }
    if (i%2==1) SGX->Cells[0][i]="dX"+Huruf;
    if (i%2==0) SGX->Cells[0][i]="dY"+Huruf;

    for (j=1;j<SGL->ColCount;j++){
        X[i][j]=0.0;
        for (k=1;k<SGQ->ColCount;k++){
            X[i][j]=X[i][j]+Q[i][k]*AA[k][j];
        }
        SGX->Cells[j][i]=FormatFloat("0.000E+00",X[i][j]);
    }
    if (i%2==0) h++;
}
}
for (i=1;i<SGL->RowCount;i++){
    for (j=1;j<SGX->ColCount;j++){
        AX[i][j]=0.0;
        for (k=1;k<SGL->ColCount;k++){
            AX[i][j]=AX[i][j]+AN[i][k]*X[k][j];
        }
    }
    SGV->RowCount=SGL->RowCount;
    SGV->ColWidths[0]=40;
    for (i=1;i<SGL->RowCount;i++){
        SGV->Cells[0][i]=IntToStr(i);
        for (j=1;j<SGX->ColCount;j++){
            V[i][j]=AX[i][j]-F[i][j];
            SGV->Cells[j][i]=FormatFloat("0.000E+00",V[i][j]);
        }
    }
}
//-----
void __fastcall TInputTerbuka::HitungKoordinat(TObject *Sender)
{
    int i,j;
    double X,Y;
    j=1;
    for (i=1;i<SGK->RowCount;i++){
        if (StrToFloat(SGK->Cells[1][j])>=1||StrToFloat(SGX->Cells[1][j+1])>=1){
            ShowMessage("Periksa data input kembali, kemungkinan data inputnya salah!");
        }
        X=StrToFloat(SGK->Cells[1][i].c_str())+StrToFloat(SGX->Cells[1][j]);
        Y=StrToFloat(SGK->Cells[2][i].c_str())+StrToFloat(SGX->Cells[1][j+1]);
        SGXD->Cells[1][i]=FormatFloat("#0.#0000",X);
        SGXD->Cells[2][i]=FormatFloat("#0.#0000",Y);
        j+=2;
    }
}

```

```

SGXD->Cells[1][0]="X";
SGXD->Cells[2][0]="Y";
for (i=1;i<SGK->RowCount;i++){
    SGXD->Cells[0][i]=SGK->Cells[0][i];
}
}
//-----
void __fastcall TInputTerbuka::chkIterClick(TObject *Sender)
{
    ino=0;
    chkIter->Caption="Iterasi ke-";
}
//-----
void __fastcall TInputTerbuka::btnStddevClick(TObject *Sender)
{
    int i,j,k,m,n;
    double So,stdX,stdY;

    m=SGSudut->RowCount+SGJarak->RowCount-2;
    n=2*(SGK->RowCount-1);
    for (i=1;i<SGV->RowCount;i++){
        for (j=1;j<SGV->ColCount;j++){
            VT[i][j]=V[i][j];
        }
    }
    for (i=1;i<SGV->ColCount;i++){
        for (j=1;j<SGV->ColCount;j++){
            VV[i][j]=0.0;
            for (k=1;k<SGV->RowCount;k++){
                VV[i][j]=VV[i][j]+VT[i][k]*V[k][j];
            }
        }
    }
    So=sqrt(VV[1][1]/(m-n));
    lbStd->Caption=FormatFloat("0.000E+00",So);
    SGD->RowCount=SGXD->RowCount;
    SGD->ColWidths[0]=40;
    SGD->Cells[1][0]="Sx";
    SGD->Cells[2][0]="Sy";
    SGD->Cells[3][0]="Su";
    SGD->Cells[4][0]="Sv";
    SGD->Cells[5][0]="Sudut t";
    j=1;
    double K[20],Su[20],Sv[20],QVY[20],QXX[20],QXY[20],Tg2T[20],T[20];
    for (i=1;i<SGXD->RowCount;i++){
        QXX[i]=StrToFloat(SGQ->Cells[j][i]);
        QVY[i]=StrToFloat(SGQ->Cells[j+1][i+1]);
        QXY[i]=StrToFloat(SGQ->Cells[j+1][i]);
        stdX=sqrt(pow(So,2)*QXX[i]);
        stdY=sqrt(pow(So,2)*QVY[i]);
        SGD->Cells[0][i]=SGK->Cells[0][i];
        SGD->Cells[1][i]=FormatFloat("0.000E+00",stdX);
        SGD->Cells[2][i]=FormatFloat("0.000E+00",stdY);
        K[i]=sqrt(pow(QVY[i]-QXX[i],2)+4*pow(QXY[i],2));
        Su[i]=So*sqrt(((QVY[i]+QXX[i]+K[i])/2);

```

```

Sv[i]=So*sqrt(((QVY[i]-QXX[i]-K[i])/2);
SGD->Cells[3][i]=FormatFloat("0.000E+00",Su[i]);
SGD->Cells[4][i]=FormatFloat("0.000E+00",Sv[i]);
Tg2T[i]=2*QXY[i]/(QVY[i]-QXX[i]);
T[i]=RadToDeg(atan(Tg2T[i]))/2;
if (T[i]<0) T[i]+=180;
if (T[i]>90) T[i]-=90;
SGD->Cells[5][i]=FloatToStr(T[i]);
j+=2;
}
}
//-----
void __fastcall TInputTerbuka::GambarEllipse(TObject *Sender)
{
if (Sender->InheritsFrom(__classid(TButton))){
TButton *tomb=(TButton *)Sender;
ShowMessage(tomb->Caption);
}
}
//-----
void __fastcall TInputTerbuka::SGDDblClick(TObject *Sender)
{
double Sudut,Abses,Ordinat,Multi;
Ordinat=StrToFloat(SGD->Cells[4][ActiveRow].c_str());
Abses=StrToFloat(SGD->Cells[3][ActiveRow].c_str());
if (Ordinat==0.0&&Abses==0.0){
Multi=1;
Abses=1;
} else Multi=Abses/Ordinat;
while (Abses<60)Abses*=10;
Ordinat=Abses/Multi;
Sudut=StrToFloat(SGD->Cells[5][ActiveRow].c_str());
if (Sudut<=90)Sudut=90-Sudut;
if (Sudut>90)Sudut=450-Sudut;
Form1=new TForm1(this);
Form1->Angle=Sudut;
Form1->Abses=Abses;
Form1->Ordinat=Ordinat;
Form1->Judul="Titik "+SGK->Cells[0][ActiveRow];
Form1->ShowModal();
delete Form1;
}
//-----
void __fastcall TInputTerbuka::SGDSelectCell(TObject *Sender, int ACol,
int ARow, bool &CanSelect)
{
ActiveRow=ARow;
}
//-----
void __fastcall TInputTerbuka::cbJenisChange(TObject *Sender)
{
GroupAwal->Visible = (cbJenis->ItemIndex==0);
GroupAkhir->Visible = (cbJenis->ItemIndex==0);
}
//-----

```

POLIGON TERTUTUP

```
//-----
#include <vcl.h>
#pragma hdrstop

#include "Input3.h"
#include "stdio.h"
#include "stdlib.h"
// #include "PolygonTypes.h"
#include "math.h"
#include "FMain.h"
#include "fstream.h"
#include "conio.h"
#include "DrwEllipse.h"
//-----

#pragma package(smart_init)
#pragma resource "*.dfm"
TInputTertutup *InputTertutup;
void __fastcall TInputTertutup::SimpanText(AnsiString NamaFile){
FILE *FSave;
    int i,j;
    AnsiString Baris,PathFile;
    NamaFile+="*.txt";
    FSave = fopen(NamaFile.c_str(), "w");
    fprintf(FSave,"%s\n\n","=== Poligon Tertutup ===");
    if (RGJenis->ItemIndex==0){
        fprintf(FSave,"%s\n\n","Tipe Sudut = Sudut Dalam ");
    } else {
        fprintf(FSave,"%s\n\n","Tipe Sudut == Sudut Luar ");
    }
    fprintf(FSave,"%s\n","=== Sudut Datar ===");
    for (i=0;i<SGSudut->RowCount;i++)
    {
        fprintf(FSave,"%s\t",SGSudut->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",SGSudut->Cells[1][i].c_str());
        fprintf(FSave,"%s\t",SGSudut->Cells[2][i].c_str());
        fprintf(FSave,"%s\n",SGSudut->Cells[3][i].c_str());
    }
    fprintf(FSave,"%s\n","=== Jarak ===");
    for (i=0;i<SGJarak->RowCount;i++)
    {
        fprintf(FSave,"%s\t",SGJarak->Cells[0][i].c_str());
        fprintf(FSave,"%s\n",SGJarak->Cells[1][i].c_str());
    }
    fprintf(FSave,"%s\n","=== Azimuth ===");
    for (i=0;i<SGH->RowCount;i++)
    {
        fprintf(FSave,"%s\t",SGH->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",SGH->Cells[1][i].c_str());
        fprintf(FSave,"%s\t",SGH->Cells[2][i].c_str());
        fprintf(FSave,"%s\n",SGH->Cells[3][i].c_str());
    }
    fprintf(FSave,"%s\n","=== Koordinat Poligon===");
    for (i=0;i<SGK->RowCount;i++)
    {

```

```

    fprintf(FSave,"%s\t",S6K->Cells[0][i].c_str());
    fprintf(FSave,"%s\t",S6K->Cells[1][i].c_str());
    fprintf(FSave,"%s\t",S6K->Cells[2][i].c_str());
    for (i=0;i<S6DK->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6DK->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[1][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[2][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[3][i].c_str());
    }
    fprintf(FSave,"%s\n",Jarak Pendekatan==");
    for (i=0;i<S6DK->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6DK->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[1][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[2][i].c_str());
        fprintf(FSave,"%s\t",S6DK->Cells[3][i].c_str());
    }
    fprintf(FSave,"%s\n",Jarak Pendekatan==");
    for (i=0;i<S6JK->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6JK->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",S6JK->Cells[1][i].c_str());
    }
    fprintf(FSave,"%s\n",Matrrik F ==");
    for (i=0;i<S6F->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6F->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",S6F->Cells[1][i].c_str());
    }
    fprintf(FSave,"%s\n",Matrrik A ==");
    char Angka[20];
    for (i=0;i<S6L->RowCount;i++)
    {
        for (j=0;j<S6L->ColCount;j++)
        strcpy(Angka,S6L->Cells[j][i].c_str());
        if (Angka=="0")
        fprintf(FSave,"%s\t\t","0");
        else
        {
            fprintf(FSave,"%s\t",Angka);
        }
    }
    fprintf(FSave,"%s\n",");
    fprintf(FSave,"%s\n",Matrrik X ==");
    for (i=0;i<S6X->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6X->Cells[0][i].c_str());
        if (i>0)
        fprintf(FSave,"%s m\n",S6X->Cells[1][i].c_str());
        else
        fprintf(FSave,"%s\n",S6X->Cells[1][i].c_str());
    }
    fprintf(FSave,"%s\n",Matrrik V ==");
    for (i=0;i<S6V->RowCount;i++)
    {
        fprintf(FSave,"%s\t",S6V->Cells[0][i].c_str());
        fprintf(FSave,"%s\t",S6V->Cells[1][i].c_str());
    }

```

```

fprintf(FSave,"So= %s\n",lbStd->Caption.c_str());
fprintf(FSave,"%s\n","=== Standard Deviasi ===");
for (i=0;i<SGD->RowCount;i++)
{
    for (j=0;j<SGD->ColCount;j++)
        fprintf(FSave,"%s\t",SGD->Cells[j][i].c_str());
    fprintf(FSave,"%s\n","");
}
fclose(FSave);
}
int AmbilDerajat(double Angle)
{
    if (Angle!=0.0) {return INT(Angle); } else return 0;
}
int AmbilMenit(double Angle)
{
    double remin;
    remin=(Angle-INT(Angle))*60;
    return INT(remin);
}
double AmbilDetik(double Angle)
{
    double remin,rede;
    remin=(Angle-INT(Angle))*60;
    rede=(remin-INT(remin))*60;
    return rede;
}
double RadToDeg(double Radian)
{
    return Radian*180/PI;
}
double DegToRad(double Degree)
{
    return Degree*PI/180;
}
double StrToSudut(char *Derajat,char *Menit,char *Detik)
{
    double hasil;
    hasil=atof(Derajat)+atof(Menit)/60+StrToFloat(Detik)/60;
    return hasil;
}
double DbtToSudut(double Derajat,double Menit,double Detik)
{
    double hasil;
    hasil=Derajat+(Menit/60)+(Detik/60);
    return hasil;
}
double Jaraknya(double X1,double Y1,double X2,double Y2)
{
    double jarak;
    double selisihX,selisihY;
    selisihX=(X2-X1);
    selisihY=(Y2-Y1);
    jarak=sqrt(pow(selisihX,2)+pow(selisihY,2));
    return jarak;
}

```

```

}
double Azimuth(double X1,double Y1,double X2,double Y2)
{
    double dx,dy,hasil;
    dx=(X2-X1);
    dy=(Y2-Y1);
    if (dy<0){hasil=(PI / 2) + atan(dx / dy);}
    else if(dy>0){hasil=(PI * 1.5) + atan(dx / dy);}
    else hasil= PI * 2;
    hasil=RadToDeg(hasil);
    return hasil;
}
double Sudut2(double X1,double Y1,double X2,double Y2)
{
    double Dx,Dy;
    Dx=(X2-X1);
    Dy=(Y2-Y1);
    float Az;

    if ((Dx>0)&&(Dy>0)) //kw I
        Az = RadToDeg(atan(Dx/Dy));
    if ((Dx>0)&&(Dy<0)) //kw II
        Az = 180 + RadToDeg(atan(Dx/Dy));
    if ((Dx<0)&&(Dy<0)) //kw III
        Az = 270 - RadToDeg(atan(Dx/Dy));
    if ((Dx<0)&&(Dy>0)) //kw IV
        Az = 360 + RadToDeg(atan(Dx/Dy));
    if ((Dx==0)&&(Dy>0))
        Az = 0;
    if ((Dx>0)&&(Dy==0))
        Az = 90;
    if ((Dx==0)&&(Dy<0))
        Az = 180;
    if ((Dx<0)&&(Dy==0))
        Az = 0.75*180;
    /*if (dy<0){hasil=(PI / 2) + atan(dx / dy);}
    else if(dy>0){hasil=(PI * 1.5) + atan(dx / dy);}
    else hasil= PI * 2;
    hasil=RadToDeg(hasil);
    */
    return Az;
}

double Sudut3(double X1,double Y1,double X2,double Y2,double X3,double Y3,int type)
{
    double hasil,f1,f2;
    f1=Sudut2(X1,Y1,X2,Y2);
    f2=Sudut2(X2,Y2,X3,Y3);
    if (type==1){
        hasil=180+f1-f2;
    } else {
        hasil=180+f2-f1;
    }

    if (hasil<0)hasil+=360;

```

```

if (hasil>360) hasil=hasil-360;
return hasil;
}
//-----
__fastcall TInputTertutup::TInputTertutup(TComponent* Owner)
: TForm(Owner)
{
}
//-----
void __fastcall TInputTertutup::Button1Click(TObject *Sender)
{
    int i,j,jrow;
    jrow=StrToInt(EdTitik->Text)+1;
    SGsudut->RowCount=jrow;
    SGJarak->RowCount=jrow;
    SGH->RowCount=jrow;
    SGK->RowCount=jrow-1;
    SGJK->RowCount=jrow;
    SGDK->RowCount=jrow;
    SGL->ColCount=StrToInt(EdTitik->Text)*2-1;
    SGL->RowCount=SGsudut->RowCount+SGJarak->RowCount-1;
    SGXD->ColWidths[0]=40;
    SGXD->RowCount=jrow-1;
    for (i=1;i<SGL->RowCount;i++)
        for (j=1;j<SGL->ColCount;j++)
            SGL->Cells[j][i]="0";
    for (i=1;i<SGsudut->RowCount;i++){
        SGsudut->Cells[0][i]=IntToStr(i);
        SGDK->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGJarak->RowCount;i++){
        SGJarak->Cells[0][i]=IntToStr(i);
        SGJK->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGH->RowCount;i++){
        SGH->Cells[0][i]=IntToStr(i);
    }
    for (i=1;i<SGK->RowCount;i++){
        SGK->Cells[0][i]=IntToStr(i+1);
    }
}
//-----
void __fastcall TInputTertutup::FormCreate(TObject *Sender)
{
    ino=0;
    SGsudut->ColWidths[0]=40;
    SGJarak->ColWidths[0]=40;
    SGsudut->Cells[0][0]="No.";
    SGsudut->Cells[1][0]="Derajat";
    SGsudut->Cells[2][0]="Menit";
    SGsudut->Cells[3][0]="Detik";
    SGJarak->Cells[0][0]="No.";
    SGJarak->Cells[1][0]="Jarak";
    SGH->Cells[0][0]="No.";
    SGH->Cells[1][0]="Derajat";
    SGH->Cells[2][0]="Menit";

```

```

        SGH->Cells[3][0]="Detik";
        SGK->Cells[0][0]="No.";
        SGK->Cells[1][0]="X";
        SGK->Cells[2][0]="Y";
        SDK->Cells[0][0]="No.";
        SDK->Cells[1][0]="Derajat";
        SDK->Cells[2][0]="Menit";
        SDK->Cells[3][0]="Detik";
        SJK->Cells[0][0]="No.";
        SJK->Cells[1][0]="Jarak";
        // DecimalSeparator=LOCALE_SDECIMAL;
        // ThousandSeparator=LOCALE_STHOUSAND;
    }
    //-----
void __fastcall TInputTertutup::BitBtn2Click(TObject *Sender)
{
    FILE *FSave;
    int i,j=1;
    AnsiString Baris,PathFile,NamaFile;
    AnsiString Buff;
    if (SaveDialog1->Execute()){
        //char Derajat[12],Menit[12],Detik[12];
        char PosX[15],PosY[12];
        PathFile=ExtractFilePath(SaveDialog1->FileName);
        NamaFile=ExtractFileName(SaveDialog1->FileName);
        NamaFile=PathFile+NamaFile.SubString(1,strlen(NamaFile.c_str()-4);
        SimpanText(NamaFile);
        FSave = fopen(SaveDialog1->FileName.c_str(), "w");
        fprintf(FSave,"%s\n",Edit1->Text.c_str());
        fprintf(FSave,"%s\n",Edit2->Text.c_str());
        for (i=1;i<SGK->RowCount;i++)
        {
            strcpy(PosX,SGK->Cells[1][i].c_str());
            strcpy(PosY,SGK->Cells[2][i].c_str());
            fprintf(FSave,"%s\n",PosX);
            fprintf(FSave,"%s\n",PosY);
        }
        fclose(FSave);
        PathFile=ExtractFilePath(Application->ExeName);
        NamaFile=PathFile+"Hasil2.html";
        FSave = fopen(NamaFile.c_str(), "w");
        fprintf(FSave,"%s\n\n","<html><head><title>=== Poligon Tertutup ===</title>");
        fprintf(FSave,"%s\n","<style type='text/css'>");
        fprintf(FSave,"%s\n","td {");
        fprintf(FSave,"%s\n","font-family: Verdana, Arial, Helvetica, sans-serif;");
        fprintf(FSave,"%s\n","font-size: 10px;");
        fprintf(FSave,"%s\n","color: #000066;}");
        fprintf(FSave,"%s\n","</style></head>");
        Fprintf(FSave,"%s\n\n","<body><center><h2>===HasilPerhitunganPoligonTertutup==</h2><br>");
        if (RGJenis->ItemIndex==0){
            fprintf(FSave,"%s\n","Tipe Sudut = Sudut Dalam <br>");
        } else {
            fprintf(FSave,"%s\n","Tipe Sudut == Sudut Luar <br>");
        }
        fprintf(FSave,"%s\n","</center><tablewidth='100%' border=1 cellpadding=4 cellspacing=0>");
    }
}

```

```

fprintf(FSave,"%s","<tr><td rowspan=2><b>Titik</b></td><td rowspan=2  

align=center><b>Sudut</b></td><td align=center colspan=3><b>Hasik</b></td>");
fprintf(FSave,"%s","<td colspan=2 align=center><b>Koordinat Pendekatan</b></td><td  

colspan=2 align=center><b>Matriks X</b></td><td colspan=4 align=center><b>Koordinat  

Definitif</b></td></tr><tr>");
fprintf(FSave,"%s","<td align=center>&nbsp;</td><td align=center><b>Azimuth</b></td><td  

align=center><b>Jarak (m)</b></td><td align=center><b>X (m)</b></td><td align=center><b>Y  

(m)</b></td><td align=center><b>dX (m)</b></td><td align=center><b>dY (m)</b></td>");
fprintf(FSave,"%s","<td align=center><b>X (m)</b></td><td align=center><b>Sx</b></td><td  

align=center><b>Y (m)</b></td><td align=center><b>Sy</b></td></tr>");

```

```

for (i=1;i<SGH->RowCount;i++)
{
fprintf(FSave,"%s","<tr>");
if (i<SGSudut->RowCount){
fprintf(FSave,"%s","<td>");
fprintf(FSave,"%s",SGSudut->Cells[0][i].c_str());
fprintf(FSave,"%s",".</td><td>");
fprintf(FSave,"%s",SGSudut->Cells[1][i].c_str());
fprintf(FSave,"%s","&deg; ");
fprintf(FSave,"%s",SGSudut->Cells[2][i].c_str());
fprintf(FSave,"%s","' ");
fprintf(FSave,"%s",SGSudut->Cells[3][i].c_str());
fprintf(FSave,"%s\\n","&quot;</td>");
}else {
fprintf(FSave,"%s","<td>");
Baris=IntToStr(i);
fprintf(FSave,"%s",Baris.c_str());
fprintf(FSave,"%s",".</td><td>&nbsp;</td>");
}
if (i<SGH->RowCount){
fprintf(FSave,"%s","<td>");
if (i<SGH->RowCount-1){
Buff=IntToStr(i)+"-"+IntToStr(i+1);
} else {
Buff=IntToStr(i)+"-1";
}
fprintf(FSave,"%s",Buff.c_str());
fprintf(FSave,"%s",".</td><td>");
fprintf(FSave,"%s\\t",SGH->Cells[1][i].c_str());
fprintf(FSave,"%s","&deg; ");
fprintf(FSave,"%s\\t",SGH->Cells[2][i].c_str());
fprintf(FSave,"%s","' ");
fprintf(FSave,"%s\\n",SGH->Cells[3][i].c_str());
fprintf(FSave,"%s","&quot;");
}else {
fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td>");
}
if (i<SGJarak->RowCount)
{
fprintf(FSave,"%s","<td>");
fprintf(FSave,"%s",SGJarak->Cells[1][i].c_str());
fprintf(FSave,"%s\\n",".</td>");
}else {
fprintf(FSave,"%s","<td>&nbsp;</td>");
}
}

```

```

    }
    if (i<=SGK->RowCount&&i>1)
    {
        fprintf(FSave,"%s","<td align=right>");
        fprintf(FSave,"%s",SGK->Cells[1][i-1].c_str());
        fprintf(FSave,"%s","</td><td align=right>");
        fprintf(FSave,"%s",SGK->Cells[2][i-1].c_str());
        fprintf(FSave,"%s","</td>");
    }else {
        fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td>");
    }
    if (i<=SGXD->RowCount&&i>1)
    {
        fprintf(FSave,"%s","<td align=right>");
        fprintf(FSave,"%s",SGX->Cells[1][j].c_str());
        fprintf(FSave,"%s","</td><td align=right>");
        fprintf(FSave,"%s",SGX->Cells[1][j+1].c_str());
        fprintf(FSave,"%s","</td>");
        if (i%2==0)j+=2;
    } else {
        fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td>");
    }
    if (i<=SGXD->RowCount&&i>1)
    {
        fprintf(FSave,"%s","<td align=right>");
        fprintf(FSave,"%s",SGXD->Cells[1][i-1].c_str());
        fprintf(FSave,"%s","</td><td>");
        fprintf(FSave,"%s",SGD->Cells[1][i-1].c_str());
        fprintf(FSave,"%s","</td><td align=right>");
        fprintf(FSave,"%s",SGXD->Cells[2][i-1].c_str());
        fprintf(FSave,"%s","</td><td>");
        fprintf(FSave,"%s",SGD->Cells[2][i-1].c_str());
        fprintf(FSave,"%s","</td>");
    }else {
        fprintf(FSave,"%s","<td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td>");
    }
    fprintf(FSave,"%s\n","</tr>");
}
fprintf(FSave,"%s","</table>");
fprintf(FSave,"%s\n","</body></html>");
fclose(FSave);
}
}
//-----

```

```

void __fastcall TInputTertutup::FormShow(TObject *Sender)
{
    FILE *stream;
    struct TOpenPolygons OP;
    int i;
    double Jarak[20];
    if (EditState==true){
        if ((stream = fopen>NamaFile.c_str(), "r")) == NULL) /* open file NameOfFile*/
        {
            MessageDlg("Error: Open File 'filename'",mtError,TMsgDlgButtons() << mbOK,0);
        }
    }
}

```

```

    }
    fseek(stream, SEEK_SET, 0);
    fread(&OP, sizeof(OP), 1, stream);
    Edit1->Text=FormatFloat("0.0000",TitikA[0].x);
    Edit2->Text=FormatFloat("0.0000",TitikA[0].y);
    EdTitik->Text=IntToStr(JumTitik);
    SGsudut->RowCount=JumTitik+3;
    SGJarak->RowCount=JumTitik+2;
    for (i=0;i<SGsudut->RowCount;i++){
        SGsudut->Cells[1][i+1]=FormatFloat("0.0000",Sudut[i]);
        //SGsudut->Cells[2][i+1]=IntToStr(Sudut[i]);
        //SGsudut->Cells[3][i+1]=IntToStr(Sudut[i]);
    }
    for (i=0;i<SGJarak->RowCount;i++){
        SGJarak->Cells[1][i+1]=FloatToStr(Jarak[i]);
    }
    fclose(stream);
}
}
//-----
void __fastcall TInputTertutup::BitBtn3Click(TObject *Sender)
{
    int i;
    double PosX,PosY;
    double Selisih;
    TTitik Titik;

    Titik.x=Edit1->Text.ToDouble();
    Titik.y=Edit2->Text.ToDouble();
    JumTitik=EdTitik->Text.ToInt();

    for (i=1;i<=SGsudut->RowCount;i++){
        Sudut[i]=StrToSudut(SGSudut->Cells[1][i].c_str(),
            SGSudut->Cells[2][i].c_str(),
            SGSudut->Cells[3][i].c_str());
    }
    for (i=1;i<=JumTitik;i++){
        Jarak[i]=StrToFloat(SGJarak->Cells[1][i].c_str());
    }
    azm[0]=DbfToSudut(Edit3->Text.ToDouble(),Edit4->Text.ToDouble(),Edit5-
>Text.ToDouble());
    // azm[0]=StrToSudut(Edit3->Text.c_str(),Edit4->Text.c_str(),Edit5->Text.c_str());

    for (i=1;i<SGH->RowCount;i++){
        if (RGJenis->ItemIndex==0){
            azm[i]=azm[i-1]+180-Sudut[i+1];
        } else {
            azm[i]=azm[i-1]+Sudut[i+1]-180;
        }
        if (azm[i]>360) azm[i]-=360;
        if (azm[i]<0) azm[i]+=360;
    }
    PosX=Titik.x;
    PosY=Titik.y;

```

```

Koord[1].x=Titik.x;
Koord[1].y=Titik.y;

if (chkIter->Checked){
    for (i=1;i<JumTitik;i++){
        {
            Koord[i].x=XPD[i].x;
            Koord[i].y=XPD[i].y;
        }
        ino++;
    } else {
        ino=0;
        for (i=1;i<=JumTitik+1;i++){
            {
                PosX=PosX+Jarak[i]*sin(DegToRad(azm[i-1]));
                Koord[i+1].x=PosX;
                SGK->Cells[1][i]=FormatFloat("0.##00",PosX);
                PosY=PosY+Jarak[i]*cos(DegToRad(azm[i-1]));
                Koord[i+1].y=PosY;
                SGK->Cells[2][i]=FormatFloat("0.##00",PosY) ;
            }
        }
        lbNo->Caption=IntToStr(ino);
        for (i=1;i<SGH->RowCount;i++){
            {
                SGH->Cells[1][i]=IntToStr( AmbilDerajat(azm[i-1]));
                SGH->Cells[2][i]=IntToStr(AmbilMenit(azm[i-1]));
                SGH->Cells[3][i]=FormatFloat("0.00",AmbilDetik(azm[i-1]));
            }
            d[1]=Jaraknya(Edit1->Text.ToDouble(),Edit2->Text.ToDouble(),Koord[2].x,Koord[2].y);
            SGJK->Cells[1][1]=FormatFloat("0.##00",d[1]);
            for (i=2;i<=JumTitik+1;i++){
                {
                    d[i-1]=Jaraknya(Koord[i-1].x,Koord[i-1].y,Koord[i].x,Koord[i].y);
                    SGJK->Cells[1][i-1]=FormatFloat("0.##00",d[i-1]);
                }
                //d[JumTitik]=Jaraknya(Koord[JumTitik-1].x,Koord[JumTitik-1].y,Edit1-
>Text.ToDouble(),Edit2->Text.ToDouble());
                //SGJK->Cells[1][JumTitik]=FormatFloat("0.##00",d[JumTitik]);
                // ShowMessage(FloatToStr(Koord[JumTitik-1].x));
                //sd[1]=Sudut3(Koord[JumTitik-1].x,Koord[JumTitik-
1].y,Koord[0].x,Koord[0].y,Koord[1].x,Koord[1].y,R6Jenis->ItemIndex);
                double hasil,f1,f2,X1,X2,Y1,Y2,X3,Y3;
                int derajat,menit;
                double detik;
                for (i=1;i<=JumTitik;i++){
                    {
                        if (i==1){
                            X1=Koord[JumTitik].x;
                            Y1=Koord[JumTitik].y;
                        }
                    }
                } else {
                    X1=Koord[i-1].x;
                    Y1=Koord[i-1].y;
                }
            }
        }
    }
}

```

```

X2=Koord[i].x;
Y2=Koord[i].y;
X3=Koord[i+1].x;
Y3=Koord[i+1].y;
f1=Azimuth(X1,Y1,X2,Y2);
f2=Azimuth(X2,Y2,X3,Y3);
if (RGJenis->ItemIndex==0){
    hasil=180+f1-f2;
} else {
    hasil=180+f2-f1;
}
if (hasil<0) hasil+=360;
if (hasil>360) hasil-=360;
sd[i]=hasil;
derajat=AmbilDerajat(hasil);
menit=AmbilMenit(hasil);
detik=AmbilDetik(hasil);
if (detik>59.99){
    menit++;
    detik=0.0;
}
if (menit==60){
    derajat++;
    menit=0;
}
SGDK->Cells[1][i]=IntToStr(derajat);
SGDK->Cells[2][i]=IntToStr(menit);
SGDK->Cells[3][i]=FormatFloat("0.00",detik);
//sd[i]=Sudut3(Koord[i-1].x,Koord[i-
1].y,Koord[i].x,Koord[i].y,Koord[i+1].x,Koord[i+1].y,RGJenis->ItemIndex);
}
//
sd[JumTitik]=Sudut3(Koord[JumTitik].x,Koord[JumTitik].y,Koord[JumTitik].x,Koord[JumTitik
].y,Edit1->Text.ToDouble(),Edit2->Text.ToDouble(),RGJenis->ItemIndex);

SGF->RowCount=SGSudut->RowCount+SGJarak->RowCount-1;
SGF->Cells[1][0]="Selisih";
for(i=1;i<SGJarak->RowCount;i++){
    SGF->Cells[0][i]="Jarak "+IntToStr(i);
    Selisih=Jarak[i]-d[i];
    SGF->Cells[1][i]=FormatFloat("0.#0000",Selisih);
}
for (i=1;i<SGDK->RowCount;i++)
{
    SGF->Cells[0][i+SGJarak->RowCount-1]="Sudut "+IntToStr(i);
    Selisih=Sudut[i]-sd[i];
    SGF->Cells[1][i+SGJarak->RowCount-1]=FormatFloat("0.#0000",Selisih);
}
int j;
j=1;
for (i=1;i<SGK->RowCount;i++)
{
    SGL->Cells[j][0]=FormatFloat("X 0",i+1);
    SGL->Cells[j+1][0]=FormatFloat("Y 0",i+1);
    j+=2;
}

```

```

}
double deltaX,deltaY,d1[20][50];
j=0;i=1;
int k;
int km=JumTitik;
for (k=1;k<=JumTitik;k++)
{
    if (k==1){
        //ShowMessage("X="+FloatToStr(Koord[km-1].x)+"\n"+
        //    "Y="+FloatToStr(Koord[km-1].y));
        d1[i][k]=((Koord[1].y-Koord[2].y)/pow(d[1],2))*206265;
        d1[i+1][k]=((Koord[2].x-Koord[1].x)/pow(d[1],2))*206265;
        d1[i+2][k]=0.0;
        d1[i+3][k]=0.0;
        d1[i+4][k]=0.0;
        d1[i+5][k]=0.0;
        d1[i+6][k]=((Koord[1].y-Koord[km].y)/pow(d[km],2))*206265;
        d1[i+7][k]=((Koord[km].x-Koord[1].x)/pow(d[km],2))*206265;
    } else if (k==2){
        //d1[i][k]=((Koord[k-1].y-Koord[k-2].y)/pow(d[k-1],2));
        //d1[i+1][k]=((Koord[k-2].x-Koord[k-1].x)/pow(d[k-1],2));
        d1[i][k]=(((Koord[k-1].y-Koord[k].y)/pow(d[k-1],2))-
        ((Koord[k+1].y-Koord[k].y)/pow(d[k],2)))*206265;
        d1[i+1][k]=(((Koord[k].x-Koord[k-1].x)/pow(d[k-1],2))-
        ((Koord[k].x-Koord[k+1].x)/pow(d[k],2)))*206265;
        d1[i+2][k]=((Koord[k].y-Koord[k+1].y)/pow(d[k],2))*206265;
        d1[i+3][k]=((Koord[k+1].x-Koord[k].x)/pow(d[k],2))*206265;
        d1[i+4][k]=0.0;
        d1[i+5][k]=0.0;
        d1[i+6][k]=0.0;
        d1[i+7][k]=0.0;
    } else
    if (k>2&& k<JumTitik){
        d1[i][k]=((Koord[k].y-Koord[k-1].y)/pow(d[k-1],2))*206265;
        d1[i+1][k]=((Koord[k-1].x-Koord[k].x)/pow(d[k-1],2))*206265;
        d1[i+2][k]=(((Koord[k].y-Koord[k+1].y)/pow(d[k-1],2))-((Koord[k+1].y-
        koord[k].y)/pow(d[k],2)))*206265;
        d1[i+3][k]=(((Koord[k].x-Koord[k-1].x)/pow(d[k-1],2))-((Koord[k].x-
        Koord[k+1].x)/pow(d[k],2)))*206265;
        d1[i+4][k]=((Koord[k].y-Koord[k+1].y)/pow(d[k],2))*206265;
        d1[i+5][k]=((Koord[k+1].x-Koord[k].x)/pow(d[k],2))*206265;
        d1[i+6][k]=0.0;
        d1[i+7][k]=0.0;
    } else
    if (k==JumTitik){
        d1[SGL->ColCount-8][k]=0.0;
        d1[SGL->ColCount-7][k]=0.0;
        d1[SGL->ColCount-6][k]=0.0;
        d1[SGL->ColCount-5][k]=0.0;
        d1[SGL->ColCount-4][k]=((Koord[km].y-Koord[km-1].y)/pow(d[km-1],2))*206265;
        d1[SGL->ColCount-3][k]=((Koord[km-1].x-Koord[km].x)/pow(d[km-1],2))*206265;
        d1[SGL->ColCount-2][k]=((Koord[km-1].y-Koord[km].y)/pow(d[km-1],2))-((Koord[1].y-
        Koord[km].y)/pow(d[km],2))*206265;
        d1[SGL->ColCount-1][k]=((Koord[km].x-Koord[km-1].x)/pow(d[km-1],2))-((Koord[km].x-
        Koord[1].x)/pow(d[km],2))*206265;
    }
}

```

```

// ShowMessage(IntToStr(i+4)+":"+IntToStr(k)+"="+FloatToStr(d1[i+4][k])+"\n"+
// IntToStr(i+5)+":"+IntToStr(k)+"="+FloatToStr(d1[i+5][k]));

}
if (kl=JumTitik){
if (d1[i][k]=0.0)
SGL->Cells[i][k+S6K->RowCount]=FormatFloat("0.##000",d1[i][k]);
if (d1[i+1][k]=0.0)
SGL->Cells[i+1][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+1][k]);
if (d1[i+2][k]=0.0)
SGL->Cells[i+2][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+2][k]);
if (d1[i+3][k]=0.0)
SGL->Cells[i+3][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+3][k]);
if (d1[i+4][k]=0.0)
SGL->Cells[i+4][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+4][k]);
if (d1[i+5][k]=0.0)
SGL->Cells[i+5][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+5][k]);
if (d1[i+6][k]=0.0)
SGL->Cells[i+6][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+6][k]);
if (d1[i+7][k]=0.0)
SGL->Cells[i+7][k+S6K->RowCount]=FormatFloat("0.##000",d1[i+7][k]);
} else {
SGL->Cells[SGL->ColCount-4][k+S6K->RowCount]=FormatFloat("0.##000",d1[SGL->ColCount-
4][k]);
SGL->Cells[SGL->ColCount-3][k+S6K->RowCount]=FormatFloat("0.##000",d1[SGL->ColCount-
3][k]);
SGL->Cells[SGL->ColCount-2][k+S6K->RowCount]=FormatFloat("0.##000",d1[SGL->ColCount-
2][k]);
SGL->Cells[SGL->ColCount-1][k+S6K->RowCount]=FormatFloat("0.##000",d1[SGL->ColCount-
1][k]);
}
if (lo2)i+=2;
}
for (i=1;i<=JumTitik;i++)
{
deltaX=(Koord[i].x-Koord[i+1].x)/d[i];
deltaY=(Koord[i].y-Koord[i+1].y)/d[i];

if (i>=2){
SGL->Cells[j-1][i]=FormatFloat("0.##00",deltaX);
SGL->Cells[j][i]=FormatFloat("0.##00",deltaY);
}
if (j<SGL->ColCount-1){
SGL->Cells[j+1][i]=FormatFloat("0.##00",-deltaX);
SGL->Cells[j+2][i]=FormatFloat("0.##00",-deltaY);
}
j+=2;
}
HitungN(Sender);
HitungQ(Sender);
HitungX(Sender);
HitungKoordinat(Sender);
}
//-----

```

```

void __fastcall TInputTertutup::HitungN(TObject *Sender)
{
    int i,j,k;
    // double det;
    SGT->RowCount=SGL->ColCount;
    SGT->ColCount=SGL->RowCount;
    SGN->RowCount=SGL->ColCount;
    SGN->ColCount=SGL->ColCount;
    SGN->ColWidths[0]=40;
    for (i=1;i<SGL->RowCount;i++){
        for (j=1;j<SGL->ColCount;j++){
            AN[i][j]=StrToFloat(SGL->Cells[j][i].c_str());
        }
    }
    for (i=1;i<SGL->ColCount;i++){
        for (j=1;j<SGL->RowCount;j++){
            AT[i][j]=AN[j][i];
            SGT->Cells[j][i]=FloatToStr(AT[i][j]);
        }
    }
    if (chkP->Checked){
        for (i=1;i<SGP->RowCount;i++){
            for (j=1;j<SGP->ColCount;j++){
                P[i][j]=StrToFloat(SGP->Cells[j][i].c_str());
            }
        }
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGP->ColCount;j++){
                PA[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    PA[i][j]=PA[i][j]+AT[i][k]*P[k][j];
                }
            }
        }
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGL->ColCount;j++){
                NN[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    NN[i][j]=NN[i][j]+PA[i][k]*AN[k][j];
                }
                SGN->Cells[j][i]=FloatToStr(NN[i][j]);
            }
        }
    } else {
        for (i=1;i<SGL->ColCount;i++){
            for (j=1;j<SGL->ColCount;j++){
                NN[i][j]=0.0;
                for (k=1;k<SGL->RowCount;k++){
                    NN[i][j]=NN[i][j]+AT[i][k]*AN[k][j];
                }
                SGN->Cells[j][i]=FloatToStr(NN[i][j]);
            }
        }
    }
    //det=Eliminasi(&NN,SGL->ColCount);
}

```

```

// ShowMessage(" Determinan "+FloatToStr(det));
SGQ->RowCount=SGI->ColCount;
SGQ->ColCount=SGI->ColCount;
}
//-----
void __fastcall TInputTertutup::HitungQ(TObject *Sender)
{
    int i,j,k,baris;
    double b;
    baris=SGN->RowCount-1;
    SGQ->ColCount=baris+1;
    /*for (i=1;i<SGN->RowCount;i++){
        for (j=1;j<SGN->ColCount;j++){
            NN[i][j]=StrToFloat(SGN->Cells[j][i].c_str());
        }
    }
    */
    for (i=1;i<baris;i++){
        for (j=1;j<baris;j++){
            if (i==j) NN[i][baris+j]=1;else
            NN[i][baris+j]=0;
        }
    }
    for (i=1;i<baris;i++){
        if (NN[i][i]==0){
            for (j=i+1;j<baris;j++){
                if (NN[j][i]==0){
                    for (k=i;k<2*baris;k++){
                        b=NN[i][k];
                        NN[i][k]=NN[j][k];
                        NN[j][k]=b;
                    }
                }
            }
        }
        b=NN[i][i];
        for (j=i;j<2*baris;j++)NN[i][j]=NN[j][i]/b;
        for (j=1;j<baris;j++){
            if (j!=i){
                for (k=i;k<2*baris;k++)C[j][k]=NN[j][k]-NN[j][i]*NN[i][k];
                for (k=i;k<2*baris;k++)NN[j][k]=C[j][k];
            }
        }
    }
    SGQ->ColWidths[0]=40;
    SGQ->ColCount=baris+1;
    for (i=1;i<SGQ->RowCount;i++){
        for (j=baris+1;j<2*baris;j++){
            SGQ->Cells[j-baris][i]=FloatToStr(NN[i][j]);
            Q[i][j-baris]=NN[i][j];
        }
    }
}
//-----

```

```

void __fastcall TForm1.Tertutup::HitungX(TObject *Sender)
{
    int i,j,k;
    SGX->ColWidths[0]=50;
    SGX->RowCount=SGQ->ColCount;
    SGX->ColCount=SGF->ColCount;
    for (i=1;i<SGF->RowCount;i++){
        for (j=1;j<SGF->ColCount;j++){
            F[i][j]=StrToFloat(SGF->Cells[j][i].c_str());
        }
    }
    if (chkP->Checked){
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGP->ColCount;j++){
                AP[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    AP[i][j]=AP[i][j]+AT[i][k]*P[k][j];
                }
            }
        }
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGF->ColCount;j++){
                AA[i][j]=0.0;
                for (k=1;k<SGP->ColCount;k++){
                    AA[i][j]=AA[i][j]+AP[i][k]*F[k][j];
                }
            }
        }
    } else {
        for (i=1;i<SGT->RowCount;i++){
            for (j=1;j<SGF->ColCount;j++){
                AA[i][j]=0.0;
                for (k=1;k<SGT->ColCount;k++){
                    AA[i][j]=AA[i][j]+AT[i][k]*F[k][j];
                }
            }
        }
    }
    int h=2;
    for (i=1;i<SGQ->ColCount;i++){
        if (i%2==1)SGX->Cells[0][i]="dX"+IntToStr(h);
        if (i%2==0)SGX->Cells[0][i]="dY"+IntToStr(h);

        for (j=1;j<SGF->ColCount;j++){
            X[i][j]=0.0;
            for (k=1;k<SGQ->RowCount;k++){
                X[i][j]=X[i][j]+Q[i][k]*AA[k][j];
            }
            SGX->Cells[j][i]=FormatFloat("0.0000E+00",X[i][j]);
        }
        if (i%2==0)h++;
    }
    for (i=1;i<SGL->RowCount;i++){
        for (j=1;j<SGX->ColCount;j++){

```

```

    AX[i][j]=0.0;
    for (k=1;k<SGX->RowCount;k++){
        AX[i][j]=AX[i][j]+AN[i][k]*X[k][j];
    }
}
}
SGV->RowCount=SGL->RowCount;
SGV->ColWidths[0]=40;
for (i=1;i<SGL->RowCount;i++){
    SGV->Cells[0][i]=IntToStr(i);
    for (j=1;j<SGX->ColCount;j++){
        V[i][j]=AX[i][j]-F[i][j];
        SGV->Cells[j][i]=FormatFloat("0.000E+00",V[i][j]);
    }
}
}
//-----
void __fastcall TInputTertutup::Button2Click(TObject *Sender)
{
    int i,j;
    SGP->RowCount=EdKolBar->Text.ToInt()+1;
    SGP->ColCount=EdKolBar->Text.ToInt()+1;
    for (i=1;i<SGP->RowCount;i++){
        for (j=1;j<SGP->ColCount;j++){
            SGP->Cells[j][i]="0";
        }
    }
}
//-----
void __fastcall TInputTertutup::btnStddevClick(TObject *Sender)
{
    int i,j,k,m,n;
    double So,stdX,stdY;

    m=2*EdTitik->Text.ToInt();
    n=2*(EdTitik->Text.ToInt()-1);
    for (i=1;i<SGV->RowCount;i++){
        for (j=1;j<SGV->ColCount;j++){
            VT[j][i]=V[i][j];
        }
    }
    for (i=1;i<SGV->ColCount;i++){
        for (j=1;j<SGV->ColCount;j++){
            VV[i][j]=0.0;
            for (k=1;k<SGV->RowCount;k++){
                VV[i][j]=VV[i][j]+VT[i][k]*V[k][j];
            }
        }
    }
    So=sqrt(VV[1][1]/(m-n));
    lbStd->Caption=FormatFloat("0.000E+00",So);
    SGD->RowCount=SGXD->RowCount;
    SGD->ColWidths[0]=40;
    SGD->Cells[1][0]="Sx";
    SGD->Cells[2][0]="Sy";

```

```

SGD->Cells[3][0]="Su";
SGD->Cells[4][0]="Sv";
SGD->Cells[5][0]="Sudut t°";
j=1;
double K[20],Su[20],Sv[20],QYY[20],QXX[20],QXY[20],Tg2T[20],T[20];
for (i=1;i<SGXD->RowCount;i++){
    // ShowMessage(SGQ->Cells[j][i]+":"+SGQ->Cells[j+1][i+1]);
    QXX[i]=Q[j][i]; // StrToFloat(SGQ->Cells[j][i]);
    QYY[i]=Q[j+1][i]; // StrToFloat(SGQ->Cells[j+1][i]);
    QXY[i]=Q[j+1][i]; // StrToFloat(SGQ->Cells[j+1][i]);
    stdX=sqrt(pow(So,2)*QXX[i]);
    stdY=sqrt(pow(So,2)*QYY[i]);
    SGD->Cells[0][i]=IntToStr(i+1);
    SGD->Cells[1][i]=FormatFloat("0.000E+00",stdX);
    SGD->Cells[2][i]=FormatFloat("0.000E+00",stdY);
    K[i]=sqrt(pow(QYY[i]-QXX[i],2)+4*pow(QXY[i],2));
    Su[i]=So*sqrt((QYY[i]+QXX[i]+K[i])/2);
    Sv[i]=So*sqrt((QYY[i]-QXX[i]-K[i])/2);
    SGD->Cells[3][i]=FormatFloat("0.000E+00",Su[i]);
    SGD->Cells[4][i]=FormatFloat("0.000E+00",Sv[i]);
    Tg2T[i]=2*QXY[i]/(QYY[i]-QXX[i]);
    T[i]=RadToDeg(atan(Tg2T[i]))/2;
    if (T[i]<0) T[i]+=180;
    if (T[i]>90) T[i]-=90;
    SGD->Cells[5][i]=FloatToStr(T[i]);
    j+=2;
}
}
//-----
void __fastcall TInputTertutup::HitungKoordinat(TObject *Sender)
{
    int i,j;
    j=1;
    SGXD->RowCount=SGK->RowCount;
    for (i=1;i<SGK->RowCount;i++){
        if (StrToFloat(SGX->Cells[1][i])>=1||StrToFloat(SGX->Cells[1][j+1])>=1){
            ShowMessage("Periksa data input kembali, kemungkinan data inputnya salah!");
        }
        XPD[i].x=StrToFloat(SGK->Cells[1][i].c_str())*X[1][j]; // StrToFloat(SGX->Cells[1][j]);
        XPD[i].y=StrToFloat(SGK->Cells[2][i].c_str())*X[1][j+1]; // StrToFloat(SGX->Cells[1][j+1]);
        SGXD->Cells[0][i]=IntToStr(i+1);
        SGXD->Cells[1][i]=FormatFloat("0.#00",XPD[i].x);
        SGXD->Cells[2][i]=FormatFloat("0.#00",XPD[i].y);
        j+=2;
    }
    SGXD->Cells[1][0]="X";
    SGXD->Cells[2][0]="Y";
}
//-----

void __fastcall TInputTertutup::chkIterClick(TObject *Sender)
{
    ino=0;
}
//-----

```

```

void __fastcall TInputTertutup::SGDSelectCell(TObject *Sender, int ACol,
int ARow, bool &CanSelect)
{
    ActiveRow=ARow;
}
//-----
void __fastcall TInputTertutup::SGDDblClick(TObject *Sender)
{
    double Sudut,Abses,Ordinat,Multi;
    Ordinat=StrToFloat(SGD->Cells[4][ActiveRow].c_str());
    Abses=StrToFloat(SGD->Cells[3][ActiveRow].c_str());
    if (Ordinat==0.0&&Abses==0.0){
        Multi=1;
        Abses=1;
    } else Multi=Abses/Ordinat;
    while (Abses<60)Abses*=10;
    Ordinat=Abses/Multi;
    Sudut=StrToFloat(SGD->Cells[5][ActiveRow].c_str());
    if (Sudut<=90)Sudut=90-Sudut;
    if (Sudut>90)Sudut=450-Sudut;
    Form1=new TForm1(this);
    Form1->Angle=Sudut;
    Form1->Abses=Abses;
    Form1->Ordinat=Ordinat;
    Form1->Judul="Titik ke "+SGD->Cells[0][ActiveRow];
    Form1->ShowModal();
    delete Form1;
}
//-----
Matriks P
/-----
#include <vcl.h>
#pragma hdrstop
#include "MatrixP.h"
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TMatrikPForm *MatrikPForm;
//-----
__fastcall TMatrikPForm::TMatrikPForm(TComponent* Owner)
: TForm(Owner)
{
}
//-----
void __fastcall TMatrikPForm::Button1Click(TObject *Sender)
{
    if (Edit1->Text.ToInt()>0)
    {
        SG1->RowCount=Edit1->Text.ToInt()+1;
        SG1->ColCount=Edit1->Text.ToInt()+1;
    }
}
//-----
Sudut

```

```

/-----
#include <vcl.h>
#pragma hdrstop
#include "DrwEllipse.h"
#include "math.h"
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
const double PI=3.14159265358979;
//-----
__fastcall TForm1::TForm1(TComponent* Owner)
: TForm(Owner)
{
    hWnd = this->Handle;
}
void TransformAndDraw(HWND hWnd)
{
}
//-----

void __fastcall TForm1::FormPaint(TObject *Sender)
{
    HDC hDC;
    XFORM xForm;
    RECT rect;
    hDC = GetDC(hWnd);
    Canvas->Rectangle(0, 0, Width, Height);
    SetGraphicsMode(hDC, GM_ADVANCED);
    SetMapMode(hDC, MM_LOENGLISH);
    xForm.eM11 = cos(Angle*PI/180);
    xForm.eM12 = sin(Angle*PI/180);
    xForm.eM21 = -sin(Angle*PI/180);
    xForm.eM22 = cos(Angle*PI/180);
    xForm.eDx = (FLOAT) 0.0;
    xForm.eDy = (FLOAT) 0.0;
    SetWorldTransform(hDC, &xForm);
    rect=GetClientRect();
    DPtoLP(hDC, (LPPOINT) &rect, 2);

    // Select a hollow brush.
    SelectObject(hDC, GetStockObject(HOLLOW_BRUSH));

    // Draw the exterior circle.
    Ellipse(hDC, (rect.right / 2 - FOrdinat), (rect.bottom / 2 + Abses),
        (rect.right / 2 + Ordinat), (rect.bottom / 2 - Abses));
    MoveToEx(hDC, (rect.right / 2 - FOrdinat-20), (rect.bottom / 2 + 0), NULL);
    LineTo(hDC, (rect.right / 2 + FOrdinat+20), (rect.bottom / 2 + 0));
    MoveToEx(hDC, (rect.right / 2 + FOrdinat-5), (rect.bottom / 2 - 3), NULL);
    LineTo(hDC, (rect.right / 2 + FOrdinat+20), (rect.bottom / 2 + 0));
    LineTo(hDC, (rect.right / 2 + FOrdinat-5), (rect.bottom / 2 + 3));
    LineTo(hDC, (rect.right / 2 + FOrdinat-5), (rect.bottom / 2 - 3));
    // Draw the vertical lines.
    MoveToEx(hDC, (rect.right/2 + 0), (rect.bottom / 2 - Abses - 20), NULL);

```

```

LineTo(hDC, (rect.right / 2 + 0), (rect.bottom / 2 + Abses + 20));

ReleaseDC(hWnd, hDC);
Canvas->Pen->Color=clBlue;
Canvas->MoveTo(0,ClientHeight/2);
Canvas->LineTo(ClientWidth,ClientHeight/2);
Canvas->MoveTo(ClientWidth/2,0);
Canvas->LineTo(ClientWidth/2,ClientHeight);
}
//-----

void __fastcall TForm1::FormShow(TObject *Sender)
{
    double Kemiringan;
    if (Angle<=90)Kemiringan=90-Angle;
    if (Angle>90)Kemiringan=450-Angle;
    lbAngle->Caption=FormatFloat("0.0000",Kemiringan);
    lbCaption->Caption=Judul;
}
//-----

```