広島大学かなた望遠鏡用自動追尾システムの開発

広島大学 理学部 物理科学科
高エネルギー宇宙·素粒子実験研究室
B044138 平木一至
主査 山下卓也　副査 生天目博文

平成 20 年 5 月 13 日

目 次

図目次

表 目 次

第1章　序論

1.1　1.5m 可視近赤外望遠鏡かなた

かなた望遠鏡は、2006 年に国立天文台から東広島天文台へ移設された主鏡径 1.5m の可視近赤外望遠鏡である。この望遠鏡は、国立天文台ハワイ観測所にある大型光学赤外望遠鏡「すばる」で使用する観測装置の開発 · 評価をする目的で国立天文台三鷹構内に建設されたものであるが、すばるの第一観測装置開発が全て終了した後、需要が減ってきたことも考慮され、本格的な天文 · 宇宙研究へ有効に活用する計画を提案した広島大学へ移管された。主鏡は有効径が 1540mm で、カセグレン及びナスミス焦点における焦点面スケールは 11.15 秒角/mm、焦点距離は 18501.7mm となっている。表 1.1 にかなた望遠鏡の仕様を紹介する。

かなた望遠鏡を用いた主な研究目的は、ガンマ線バーストなどの突発天体の即時観測による高エネルギー宇宙現象の解明である。これらの即時観測にはアラート対応観測が含まれるが、具体的には、ガンマ線モニター Swift 衛星や、2008 年に NASA から打ち上げられる次世代ガンマ線衛星 GLAST などがガンマ線バースト等の突発天体を検出した時にインターネットを通じて発するアラートを受信して、天体の座標を特定し、自動的に追尾観測することを目指している。また、X 線衛星 Suzaku などと連携した多波長観測 (可視からガンマ線領域) も行っている。

かなた望遠鏡は、高エネルギー現象の中でも、特に観測が困難なガンマ線バーストに主眼を置いた研究を推進している。これらの突発天体は、その発生後から劇的に光度が減少し、観測が困難になる。例えば、爆発直後 (1000 秒以内) の偏光観測はこれまでほとんど行われておらず、バーストの物理解明のためにもこういった即時対応観測の実現が求められている。このため、これらの天体を観測するためには発生後にいち早く望遠鏡を目的天体に向けることが要求される。かなた望遠鏡はガンマ線バーストの観測に対応するため、即時観測体制を整え、結果としてかなた望遠鏡をガンマ線バースト発生後 1 分程度で自動的に、かつ即座に目的天体に向けることが可能である。これは主鏡径 1.5m クラスの望遠鏡では世界最高水準の速度である。

図 1.1: かなた望遠鏡

表 1.1: かなた望遠鏡の仕様

項目	仕様
光学系	リッチ · クレチエン光学系
主鏡	有効径 1540mm/主鏡の F 比=2.0
焦点モード	カセグレン焦点 · ナスミス焦点とも (F/12.01)
焦点面スケール	カセグレン · ナスミスとも 11.15 秒角/mm
焦点距離	焦点距離 18,501.7mm
分解能	1”FWHM
視野	15 分角 φ
最大駆動速度	方位軸 5°/sec
	高度軸 2°/sec
最大加速度	1°/sec^2 以上
架台	経緯台方式

1.2 一露出型可視広視野偏光撮像器 HOWPol

HOWPol は広島大学宇宙科学センターで開発を進めている突発天体に特化した偏光観測装置である。全長はおよそ 1m、重量約 200kg の装置であり、かなた望遠鏡のナスミス焦点に常設されている。本格的な偏光装置は通常、非対称なナスミス焦点に取り付けることはないが、突発天体の観測に伴う常時設置の必要性から、本装置はあえてナスミス焦点に取り付けられている。これから大型化していく望遠鏡ではカセグレン焦点がなくなり、ナスミス焦点にのみ装置をつけることが検討されている例もあり、その意味でも、ナスミス焦点に取り付け、偏光装置がどれほどの性能を発揮できるかという点で注目されている。また、検出器には浜松ホトニクスで開発が進められた完全空乏型 CCD を搭載している。

を持っていることが特徴である。また、HOWPol のモードには測光、偏光モードの他に幅 2.2 秒角のスリットを焦点マスクに用いた分光モード、偏光分光モードがある。では次の表 1.2 に、HOWPol の基本仕様を明記しておく。

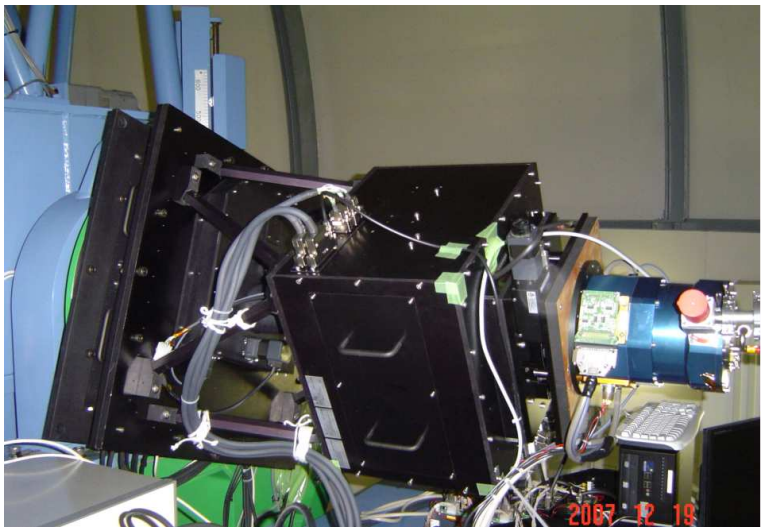

図 1.2: **HOWPol**

表 1.2: HOWPol の仕様

項目	仕様内容
視野	15 分角 (56 % 縮小光学系、0.3 秒角/pixel)
検出器	完全空乏型 CCD $2k \times 4k \times 2$
波長領域	4500-11000 Å
フィルタ	広帯域 (B,V,Rc,Ic,z')、各種狭帯域
モード	偏光撮像、撮像、分光、偏光分光モード
限界等級	測光モード (10min exp, $S/N = 50$ 19.8 mag)
	偏光モード (10min exp; $\delta P = 0.2\%$ 15.9 mag)

1.3 可視赤外線カメラ

可視赤外線カメラは広島大学宇宙科学センターで開発を進めているかなた望遠鏡のカセグレン焦点に常設予定の撮像カメラであり、突発・変動天体や、突発的変動を示す高エネルギー現象を可視光と近赤外線の両波長で同時にモニターできる装置である。このカメラは可視光検出器として 2k × 4k 完全空乏型 CCD1 素子、近赤外線検出器として 2k × 2k の VIRGO、さらにもう一つ別の 2k × 2k の検出器 1 素子の計 3 素子を搭載する予定で、撮像、偏光、分光の 3 つのモードを有する。可視赤外線カメラの簡単な仕様を次の表 1.3 に示す。

表 1.3: 可視赤外線カメラの仕様

項目	仕様内容
検出器	2k × 2k 赤外線検出器 2 素子
	2k × 2k 可視 CCD 1 素子
視野	10 分角
ピクセルスケール	0.3 秒角/pix
モード	撮像 · 分光 · 偏光

1.4 分光観測とオートガイダー

望遠鏡は通常、恒星の日周運動に応じた追尾機能を有しているが、長時間露光すると星が一ヶ所に留まらずズレを生じる。これを追尾誤差という。この誤差が無視できない分光観測のような長時間露光を必要とする観測では、観測者が観測を中断して望遠鏡の方向を修正することで追尾誤差を修正している。このような手間や効率の悪さを改善するのが今回開発するオートガイダーである。以下に、分光観測について説明し、その後になぜオートガイダーが必要かを述べる。この観測は以下の様な手順を踏む。

1. 望遠鏡の初期導入として望遠鏡を観測星の方向へ向ける。かなた望遠鏡の指向精度 3 秒角 (r.m.s) に対して例えば HOWPol のスリット幅は 2.2 秒角なので、この時点でスリットに観測星が入ることはまずない。

2. スリットマスクが入っている状態で 1 枚撮影し、スカイバックグラウンドの明かりを利用してスリットのイメージを取得する。

3. スリットを外した状態で目的とする天体の像を 1 枚撮影する。

4. 2 と 3 のイメージを比べてスリットと観測星とのズレを算出し、望遠鏡をその分だけ動かす。

5. 3 と 4 を繰り返して、スリット内に観測星が入るように調整する。

6. スリット内に観測星が入っている状態で分光観測開始

7. 数分観測すると観測星がズレて、スリットから外れるので、3 と 4 を行い、スリット内に観測星を入れ直し、観測を再開する。

つまり、スリット幅が 2.2 秒角であり、星の位置がスリット中心から 1.1 秒以上ズレると光を大きくロスしてしまうことになる。すると数分ごとに観測を中断して望遠鏡のズレを調節しなけれならず、分光観測をする上で非常に手間がかかる。しかもこの手順で観測を行うと望遠鏡の修正のために一時観測を中断せねばならない、観測を中断することは限られた時間内での観測で時間的にも、取得するデータがこま切れになるという点でも効率が悪い。
そこで、オートガイダーを用いることになる。オートガイダーとは星の挙動から望遠鏡に生じている追尾誤差を検出し、それを修正させる望遠鏡の自動追尾システムである。オートガイダーを用いると上の手順の 7 が不用となり、途中で観測を中断することもなくなるので長時間の観測が可能となる。また、オートガイダーは分光観測だけではなく通常の撮

は、十分な光量を確保するため長時間の観測を必要とする。長時間の観測をすると、観測中にその範囲内でのズレが生じる。一方で天体の観測は検出器の同じ場所、つまりCCD上の同じピクセルに天体を「固定」したままで観測をしたい。なぜなら天体がズレていくと像がボヤけてしまい分解能が悪くなり、単位時間に得られるカウントが少なくなるほか、空間的な解像度も悪くなってしまう。このためにオートガイダーが必要とされる。
今回開発したオートガイダーはナスミス焦点に常設される観測器HOWPol専用のものであるが、1.4節で紹介した可視赤外線カメラにもこれとは別のオートガイダーを設ける予定であり、機器構成や制御ソフトは今回開発するものにならって製作することを予定している。

1.5 オートガイダーに要求される機能

オートガイダーは長時間の観測中にたえず正確な位置情報を望遠鏡に送り続ける必要がある。本開発にあたり、それをC++言語を用いた機器制御ソフトウェアによって実現させる。そのためにまず、機器制御の通信プロトコルやC++言語によるソフトウェア開発等について学ぶ必要がある。なお、オートガイダーのハードウェアは2008年10月までに完成している。

オートガイダーには、撮像後、CCDカメラから得たカウントデータから星像の輝度分布の重心を算出する機能と星像のサイズよりも小さい位置精度で追尾する機能が最低限要求される。

また、今回開発を進めているオートガイダーでは、オートガイドの機能とは直接は関係ないが、星像の輝度分布のFWHMを検出する機能を搭載させることも目標とする。これは、CCDカメラから得たカウントデータをx、yそれぞれの方向に一次元化し、このデータにガウシアンでフィッティングを行うことで実現する。このFWHMを検出する機能によって、ガイド星の像の大きさを監視して、気流変化による星像状態(シーイング)をモニターして観測計画立案に役立てることが可能となるほか、焦点あわせを自動化して、観測を切替える合間にこまめに焦点合わせをすることで、常に最良の状態で観測を行うこともできるようになる。

本研究の主目的をまとめると、既に組み上がっているかなた望遠鏡用のオートガイドシステムにおいて

1. オートガイダー制御ソフトウェアを自分自身で一から開発すること
2. 制御ソフトウェア開発の上でC++言語とVisual C++についての知識およびパルスモーター制御、CCD制御、望遠鏡制御の方法や、非線型関数のフィッティングが必要となるのでそれらについて学ぶこと
3. かなた望遠鏡の制御システムに接続して試験観測を行い、所期の性能を確認すること

以上の3点であり、これによりHOWPolをはじめとするかなた望遠鏡による観測の効率を上げることを目指す。

第2章　オートガイダーの開発

2.1　装置構成

オートガイダーは検出部である冷却 CCD カメラ、$20 \times 20mm$ 45°直角プリズム、駆動部には CCD カメラを 2 次元で駆動させる X、Y ステージを備える。その他には分光観測に用いるコンパリソンランプとコンパリソンランプの出し入れを行う θ ステージが搭載されている。次の図 2.1 にオートガイダー内部の様子を示す。黄色の楕円で囲まれているのがステージ駆動用のモーターと軸である。縦向きのものが X 軸、横向きのものが Y 軸である。赤い楕円に囲まれているのがコンパリソンランプを出し入れする為の θ 軸で、紫色の円に囲まれている部分に CCD カメラが内装されている。

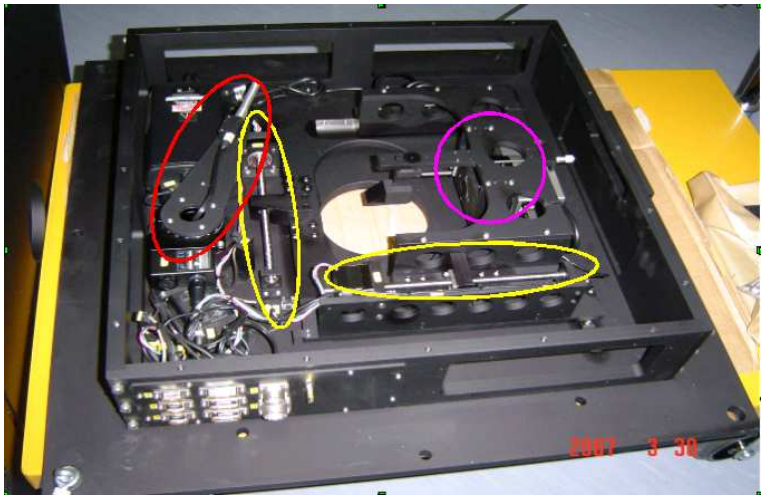

図 2.1: オートガイダーの内部図

図 2.2: **BTRAN** 社製 冷却 **CCD** カメラ

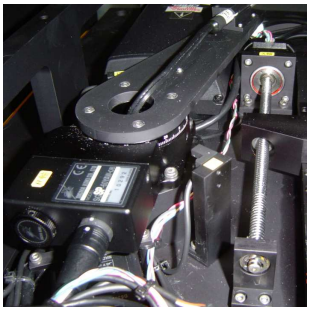

図 2.3: **x,**θ 軸とモータ

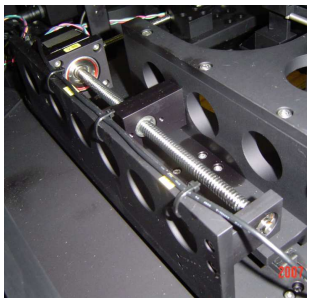

図 2.4: **y** 軸とモータ

2.2 光学系と検出器及び周辺機器

2.2.1 検出部

オートガイダーの検出部は導入用 $20 \times 20mm$ 45° 直角プリズム (シグマ光機 RPB2-20-550) と、BITRAN 製冷却 CCD カメラ BS-41L から成る。このカメラは有効画素数 1360×1024 画素、受光面積 $8.8 \times 6.45mm$(かなた望遠鏡に取り付けた時の視野は 97.8×73.6 秒角) であり、USB 通信によってノート PC からの CCD の制御を可能としている。この CCD カメラは 1×1、2×2、3×3、4×4、8×8、16×16、32×32 pixel の 7 種類のビンニング設定と 0.001 秒 ～ 6553 秒まで 0.001 秒刻での露出時間の設定が可能である。本オートガイダーではビンニングは 4×4 で固定し、露出時間では 0.001 秒 ～ 6553 秒 の間で可変とした。ビンニングを 4×4 に固定した理由は次の通りである。恒星像のサイズ (シーイングサイズ = 輝度分布の半値幅) は良好な日で約 1 秒角でそれを $3 \times 3 \sim 5 \times 5$ ピクセルの範囲に表示できると必要十分である。この条件については後述するが、最終的には 4×4 ビンニングを採用した。露出時間を可変としたのは、観測時に天候や星の明るさなどの諸条件で露出時間を変える必要があるからである。オートガイダーで用いた CCD カメラの仕様と波長特性を次の図 2.2、表 2.1 に示す。

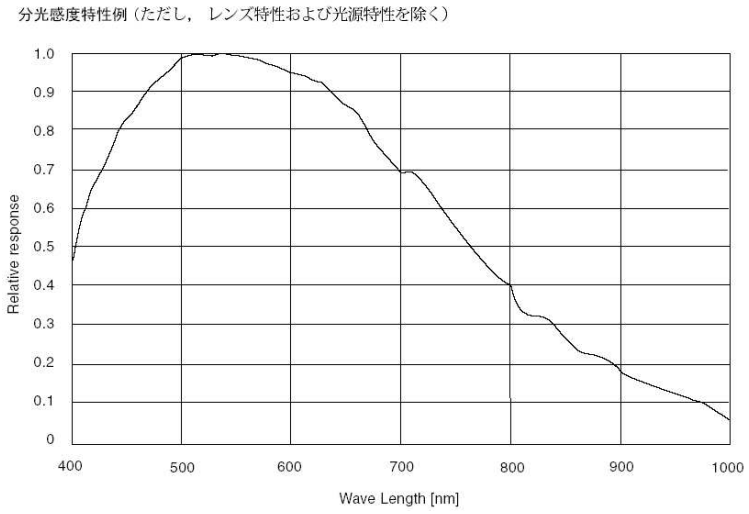

図 2.5: **CCD** の感度曲線

表 2.1: 冷却 CCD カメラの仕様

項目	仕様
メーカー	BITRAN
通信方式	USB
有効画素数	1360×1024 画素
受光面積	$8.8 \times 6.45mm$(97.8 × 73.6 秒角)
ピクセルサイズ	$6.45 \times 6.45\mu m$

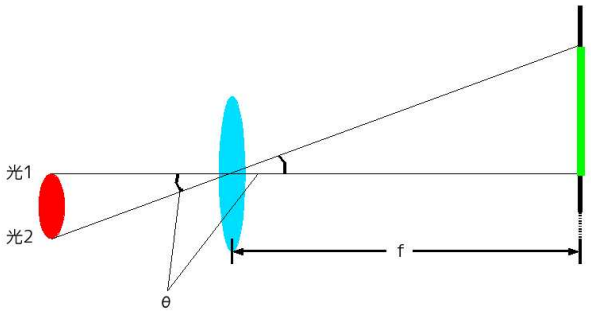

図 2.6: ビンニングと焦点距離 f

前述した焦点距離 f とビンニングについてもう少し詳しく説明していく。まず、図 2.3 のような場合を考える。赤い楕円は有限な大きさを持った天体、青い楕円はかなた望遠鏡の主鏡、緑色の線が天体像である。主鏡はレンズではないが今は説明を簡略化するために主鏡と等価な透過レンズとして書いている。天体の両端から出た光が主鏡を通り、観測装置で像を結ぶとして、図ではそれぞれの軸中心の光路を直線で示している。θ は天体の見込み角である、主鏡から星像までの直線距離は焦点距離 f で、かなた望遠鏡では f=18501.7mm である。オートガイダーに取り付けられている CCD カメラのピクセルサイズは $6.45\mu m\square/pix$ であるので 1 ピクセルあたりの見込み角は

$$\theta = \arctan\frac{0.00645mm}{18501.7} \simeq 0.072”$$

となる。ここで天体が恒星の場合を考える。恒星は非常に遠くにあるので実質的に無銀小の点光源と考えて良いが、地球大気のせいで有限の大きさを持つ。この大きさ (半値幅 FWHM) をシーイングサイズと呼ぶ。星のシーイングサイズは条件が良好な日で 1”角くらいなのでビンニングなしの状態だと星像は 14×14 ピクセルくらいの大きさになる。このままの状態だと解像度は非常に良いがデータの読み出しに時間がかかり、無駄が生じる。一方で、シーイングサイズが 1×1 ピクセル程度になると位置や測光の誤差が大きくなる。ナイキスト · サンプリング定理などを考慮すると 1”角が 3 ピクセル程度になるようにビンニングするのが丁度よい。そこでビンニングを 4×4 を採用した。

2.2.2 駆動 · 制御系

オートガイダーの筐体には CCD カメラを 2 次元で駆動させる X、Y ステージと分光観測用のコンパリソンランプ、コンパリソンランプを出し入れする θ ステージが搭載されている。X、Y、θ ステージはそれぞれ 1 つのモータと 2 つのリミットセンサー (フォトインターラプタ) を持つ。それぞれのモータはコンテック社 AL シリーズ対応 2 軸ステッピングモータドライバ内蔵コントローラスレイブ CD-772/ADB5331A を介して PC から発せられたコマンドによって駆動する。次の図はオートガイダー内部のステージの簡単なブロック図と制御系のブロック図で、表はモーターとステージの仕様である。

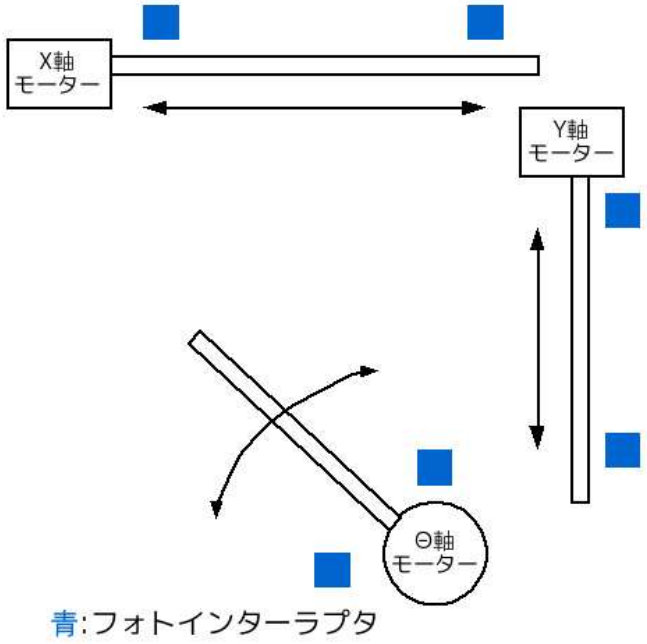

図 2.7: ステージのブロック図

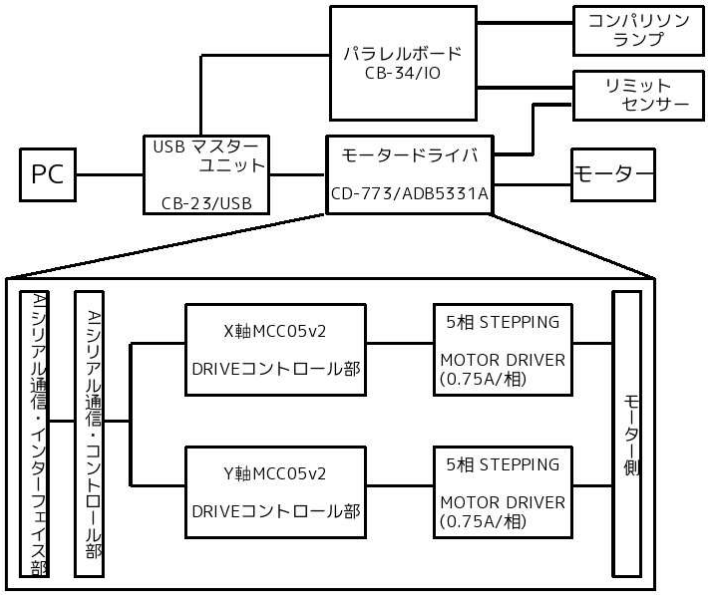

図 2.8: 制御系のブロック図

表 2.2: モーターおよびステージの仕様

項目	X 軸	Y 軸	θ 軸
メーカー	KSS/THK	KSS/THK	シグマ光機
型式	TS3667N14E2/SRS12M	TS3667N14E2/SRS12M	SGSP-60YAW-0B
1 パルス移動量	$4\mu m$	$4\mu m$	0.005°
送り方式	カップリングレス ボールねじ式	カップリングレス ボールねじ式	ウォーム ウォームホイール
ねじ軸外径	4mm	4mm	-

表 2.3: ステージの仕様

項目	仕様
通信方式	USB
軸	X · Y · θ 3 軸
ストローク	X 軸 115mm
	Y 軸 115mm
駆動速度	X,Y 軸 HIGH SPEED 11.06mm/s
	LOW SPEED 0.66 mm/s

2.3 オートガイダー制御ソフトウェア開発

私はオートガイダーを駆動させるためのソフトウェアを、Visual C++言語を用いて開発した。Visual C++言語を用いた理由は、C++言語を学びたかったこと、及び CCD やモータ制御のためのライブラリとして Visual C++用のものがメーカーより提供されていたことによる。オートガイダー駆動用のソフトウェアには、PC への画像の取り込みと表示、画像データから星像の重心と FWHM を決定して重心のズレとシーイングの変化を算出し、かなた望遠鏡へのフィードバック制御を行う機能などを実装する必要がある。

2.3.1 CCD カメラ制御

BITRAN 社製冷却 CCD カメラの制御用ソフトウェアの開発には同社製サンプルプログラムを参考に開発を進めた。オートガイダーに搭載される CCD カメラに要求される機能は、ガイド星の光度に合わせた露出時間で露光し、その 2 次元画像を生カウントで読み出し、正確な重心座標を導出してそのズレ量を算出することである。この節ではこのシステムの構築について述べていく。ではまず、 CCD の撮像から標準偏差の算出までの手順を次に示す。

ログラム上の配列変数に格納する

2. 表示ウィンドウ上に画像表示用のスペースを設け、そこへ 1 ピクセルずつ点を打っていき、星像を表示する。

3. ガイド星の輝度分布の重心座標を求める。

この 1 から 3 の手順を繰り返して行い、重心の位置座標の平均値を求め、そこから重心座標のズレをひとまずピクセル単位で求める。

ここまでを CCD カメラの制御としてシステムを構築していく。なお、CCD による撮像、ビンニング等の初期設定などは BITRAN 社 BS シリーズ通信仕様書記載のコマンドを用いて行った。

PC と CCD の通信仕様

BITRAN 冷却 CCD カメラの通信では PC 側がコマンドを出力して、返事があるコマンドに対してのみ CCD カメラが応答を返すという形をとる。コマンドは 1 バイトを基本として、必要な場合はパラメータが続くか、又は応答として戻りデータが来る。CCD カメラのコマンドについては機密保持契約 (有償) の締結者のみが使用できることになっていて詳しくは説明できないので今回使用したコマンドの項目だけ簡単に紹介する。

CCD 制御用コマンド

1. カメラのリセット

2. 環境設定 (A/D 変換や外部トリガの使用等を設定)

3. ビンニング設定 (4 × 4 ビンニングにすることで 16 ピクセルを 1 ピクセルとして扱い、1 ピクセル当たりの感度を上げる)

4. 露出時間の設定

5. 撮影開始

6. 冷却パワー設定 (CCD カメラ自身にあるノイズをできるだけ抑えるために冷却する)

7. カメラの ID コードを得る (カメラの型式によってはライブラリに適合しないことがあるのでそれを避けるため)

8. カメラの状態を得る (これによりカメラがアイドリング中なのか、コマンドの実行中なのか等の状態を取得する)

9. CCD の素子の温度を得る (CCD の温度を一定に管理するのに CCD 素子の温度を取得し続ける)

10. 指定された画素データを得る

ここからは CCD 制御の為の各機能と処理の流れについて説明していく。図 2.9 は製作したオートガイダー制御用の GUI の画面である。

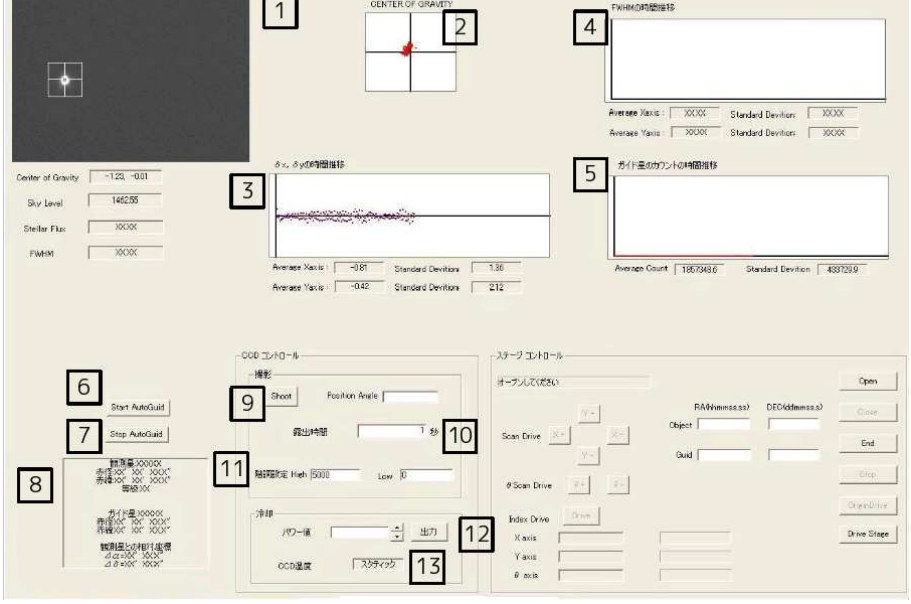

図 2.9: **CCD** 制御用の **GUI** 画

1. 星像の表示画面
   CCD カメラで撮影した星像をここに表示する。この画面に表示される画像は JPG 等の画像の形式にせず、CCD から送られたデータを 256(0-255) 段階の階調に変換してその階調したデータを 1 点 1 点打っていくという処理をしている。こうした方が後に説明する重心や FWHM といったものを算出するのに都合が良いし、この画面で表示できる 340×256 ピクセル程度の画像では表示の処理にそれほど時間がかからないので今回はこのような手法を用いた。

2. 重心の表示画面 (CENTER OF GRAVITY)
   オートガイド機能で一番最初に検出した重心の位置を中心にして 2 次元で重心がどう分布するかを知るためのグラフ。

3. $\delta x, \delta y$ の時間推移グラフ
   重心のズレ δx と δy の時間推移を表すためのグラフ。グラフの下にある Average X,Yaxis と Standard Deviation はそれぞれ重心からのズレの平均値と標準偏差を表示。

4. FWHM の時間推移グラフ
   x,y それぞれの方向の FWHM の値の時間推移を表すためのグラフ。グラフの下にある Average X,Yaxis と Standard Deviation はそれぞれ FWHM の平均値と標準偏差を表示。

5. ガイド星のカウントの時間推移グラフ
   ガイド星の全カウント数の時間推移を表したグラフ。グラフの下にある Average Count と Standard Deviation はそれぞれカウントの平均値と標準偏差を表示。

6. Start AutoGuid ボタン
   オートガイド開始ボタン。このボタンで CCD の全撮像領域を対象としてオートガ

処理の流れ:

(a) CCD の露出時間を得る

(b) CCD の環境設定

(c) ビンニングの設定

(d) 取得した露出時間を CCD に送り設定する

(e) 撮像

(f) 撮像したデータを CCD カメラから取得する

(g) 取得データを 256(0-255) 階調に変換

(h) 変換したデータを表示画面に表示

(i) 取得したデータから重心とスカイバックグラウンドを算出

(j) 算出した重心とスカイバックグラウンドの数値を表示し、「重心の表示画面」に重心位置をプロット

(k) 星像の全カウントとその平均値、標準偏差を算出し、平均値と標準偏差を表示、「ガイド星のカウントの時間推移グラフ」に全カウントをプロットする

(l) 星像の重心のズレとその平均値、標準偏差を算出し、平均値と標準偏差を表示、「 $\delta x, \delta y$ の時間推移グラフ」にズレ量をプロットする

(m) 星像の FWHM とその平均値、標準偏差を算出し、平均値と標準偏差を表示、「FWHM の時間推移グラフ」に FWHM の値をプロットする

(n) (e) にもどる

7. Stop AutoGuid ボタン
   オートガイド中止ボタン。

8. 観測星及びガイド星情報表示画面
   観測星の名前、座標 (赤径赤緯)、等級、ガイド星の座標 (赤径赤緯)、をカタログから検索、表示し、その情報から二つの星の相対座標 ($\Delta\alpha$、$\Delta\delta$) を算出、それらの情報を表示する (未完成)。

9. Shoot ボタン
   撮影のボタン。このボタンで 1 回だけ撮影できる。
   処理の流れ:

   (a) CCD の露出時間を得る

   (b) CCD の環境設定

   (c) ビンニングの設定

   (d) 取得した露出時間を CCD に送り設定する

   (e) 撮像

   (f) 撮像したデータを CCD カメラから取得する

   (g) 取得データを 256 階調に変換

(i) 取得データから重心を検出

10. 露出時間の設定
    0.001 秒から 6553 秒まで設定可能。単位は全て秒で、デフォルトは 1 秒。

11. 階調の設定
    カウントデータを 256 階調に変換する際に 0 と 255 に対応するカウント (0 ～ 65535) をそれぞれ設定する。

12. 冷却出力ボタン
    CCD の冷却をするボタン。CCD 素子の温度も知ることができる。
    処理の流れ:

    (a) 冷却パワー値を取得

    (b) CCD に冷却パワー値を送り、冷却開始

    (c) 一定時間ごとに CCD の温度を取得して表示

13. CCD 温度表示
    CCD 素子の温度をカメラから取得し、表示する

2.3.2 駆動系の制御

PC とステージの通信仕様

ステージの制御は、モータドライバ CD-733 に実装されている読み出し専用の STATUS PORT の 1~5 と、書き込み専用の DRIVE COMMAND PORT、および書き込みと読み出しの両用である DRIVE DATA PORT の 1~3 の各ポートに専用の関数を用いてコマンドを送り、必要な場合はパラメータが続くか、応答として戻り値が返されることを通じて行う。コマンド及びパラメータは 1 ポート 1 バイトを基本とする。ではここからは、ステージ制御の為の各機能と処理の流れについて説明していく。図 2.10 はオートガイダー制御用の GUI 画面の中でもステージ制御を行う画面である。

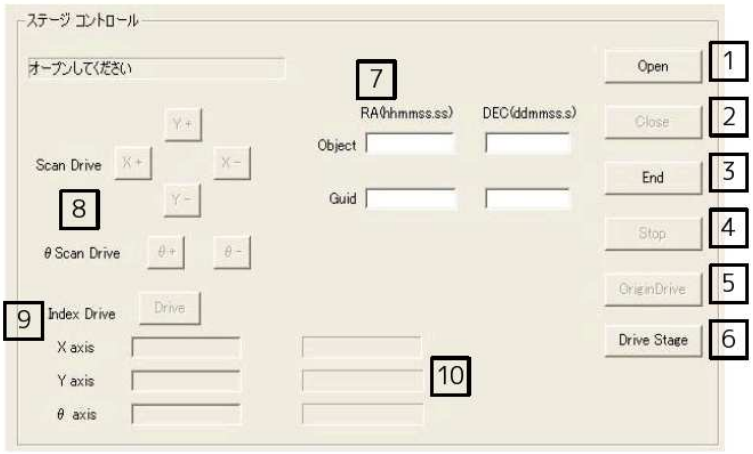

図 2.10: ステージ制御用の GUI 画

1. Open ボタン
   通信用デバイスのオープンとモーターの各種初期設定を行う
   処理の流れ:

   (a) x,y,θ 各軸のモーターの通信用デバイスを開く

   (b) DRIVE TYPE と LIMIT STOP TYPE を設定する。今回は DRIVE TYPE は L-TYPE、LIMIT STOP TYPE は即時停止を選択した。

   (c) モーターの加速 · 減速時定数と DRIVE の LOW SPEED と HIGH SPEED を 3 軸について設定する。今回は時定数を加速 · 減速ともに 50ms/1000Hz、LOW SPEED を 3000Hz、HIGHT SPEED を 50000Hz に設定した。時定数と SPEED については特にその数値に制限はなく、この値で現在は支障なく動作している。

   (d) ボタンの使用制限を解除し、ステージの制御に必要なボタンを使用可能にする。

2. Close ボタン
   通信用デバイスを閉じて、ステージ制御に必要なボタンに使用制限をかける

3. End ボタン
   オートガイダー制御用アプリケーションを終了させる

4. Stop ボタン
   モーターの DRIVE を即時停止させる

5. Origin Drive ボタン
   ステージの原点を検出し、そこまでステージの DRIVE を行う。X,Y 軸については CW(+) 方向のリミットセンサを原点とし、θ 軸については CCW(-) 方向のリミットセンサを原点とする。こうしたのは、X,Y 軸の原点が片方でもこの条件と逆になっていると θ 軸を起動してコンパリソンランプを出し入れする際にランプが導入用の平面鏡にぶつかってしまうためである。
   処理の流れ:

   (a) 原点として指定したリミットセンサが ON になるまで DRIVE する

   (b) 原点から 200pulse 逆方向に動かす

   (c) リミットに向かって 5pulse だけ動かし、リミットセンサが反応したかチェックする。これを繰り返す。

   (d) (c) を繰り返して最初にリミットが反応した点を原点とする

   X,Y 軸については上の処理で原点の検出を行ったが、θ 軸については既存の ORIGIN DRIVE コマンドを用いた、これは前述した理由に加えて、ORIGIN DRIVE コマンドでは CW 方向のリミットセンサを原点に指定できないからである。

6. Stage Drive ボタン
   ガイド星が CCD の視野中心に入るように駆動させるボタン。7 で紹介するボックスに観測星とガイド星の座標を入力することで CCD の視野中心にガイド星がくるようにステージを駆動させる。
   処理の流れ:

   (a) ステージの現在地を検出する

(c) 2 で算出した観測星とガイド星の距離 (arcsec 単位) をパルス数に変換する

(d) ステージの現在地と 3 で算出したパルス数からガイド星が撮像できる地点までステージを駆動させる

7. 座標入力ボックス
   観測星とガイド星の座標を入力する。

8. Scan Drive ボタン
   Scan Drive をさせるボタン。Scan Drive は Stop ボタンを押すかリミットに入るまで指定した方向に動き続ける DRIVE。

9. Index Drive ボタン
   指定した pulse 数分だけ DRIVE を行う。ボタン下のボックスに pulse 数を入力後 Drive ボタンを押すと DRIVE を行う。

10. 座標表示ボックス
    ステージの原点からの現在位置をパルス数単位で表示する。

2.4 かなた望遠鏡の制御

オートガイダーの役割は望遠鏡の追尾誤差を検出し、その誤差を望遠鏡にフィードバックして望遠鏡を天球上の一点を見続けるように精度良く制御することにある。ここではかなた望遠鏡とオートガイダーの通信と追尾誤差をはじめとする諸変数のやりとりについて説明していく。

2.4.1 望遠鏡に送る位置誤差情報

かなた望遠鏡へ送信するべき追尾誤差情報は、オートガイダーで算出されたズレを回転させ、さらにスケールファクターをかけたものになる。まず回転行列を掛けるが、一つはナスミス焦点にあるローテータの回転による寄与が、もう 1 つは装置の取り付け方向によるオフセット角の寄与とがある。経緯台の望遠鏡では、天体を観測していると刻々と天体像が回転してしまうためこれを補償するローテータが必要になるが、これを積極的に利用して、ある特定の視野の向きに固定して追尾する場合があり、その向きを考慮しなくてはならない。ローテータはカセグレン焦点 · 第一ナスミス焦点に設置されている。

これらの寄与を入れて星像の位置をピクセル単位から赤径赤緯へ変換するには次の式を用いる

$$\begin{pmatrix} \delta\alpha \\ \delta\Delta \end{pmatrix} = A \times SF \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix}$$

ここで SF はスケールファクターで単位は (arcsec/pix) である。ローテータによる寄与は右辺真ん中の回転行列によって表される。

SF についてであるが、オートガイダーの CCD カメラは 1 ピクセルあたり 0.072 秒角 (2.2.1 節 f 値とビンニング参照)、ビンニングが 4×4 なので SF は $SF = 0.072" \times 4$ となる。

赤径赤緯と表示画面の座標とのズレ θ_0 を用いて $\theta = \theta_{PA} + \theta_0$ と書ける。

追尾誤差にはローテータとは別に鏡の反射による寄与も考えなくてはならない。星の光がオートガイダーの CCD に届くまでに第三鏡、導入用平面鏡の 2 枚の鏡の反射による星像の反転の効果が加わる。A が鏡による星像の反転の効果を表し、A = 1 or -1 である。他のパラメタは前述の変換式と同じである。今回の場合は鏡による反転が 2 回であり、結果として A=(1,1) で良いことが試験観測の結果確認できた。

2.4.2 かなた望遠鏡との通信

ソケット通信

かなた望遠鏡とオートガイダーとの通信はソケット通信を使って行うようにした。ソケット通信を用いたのはかなた望遠鏡の制御プロセス(UNIX マシン上で起動) がそのような仕様となっているためである。Windows Socket の仕様は基本的に UNIX と同じであるので、UNIX ～ Windows 間のネットワーク通信は容易に実現することがでた。

ソケット通信にはストリームソケットとデータグラムソケットの 2 種類があり、どちらのソケットも全二重の同時双方向通信を行うことができる。今回開発したオートガイダーではストリームソケットを用いてかなた望遠鏡との通信を確立している。ストリームソケットとは連続的なデータを境界のない形式で送ることのできる方式で、実際のデータ送受信に際してコネクションを確立しておく必要があるが (コネクション型通信方式)、このために信頼性のあるデータの送受信ができる。データグラムソケットはここでは詳しくは割愛するが簡単に説明するとある大きさのレコードを単位としたコネクションレス型通信方式である。

かなた望遠鏡との通信の流れ

ここでかなた望遠鏡とオートガイダーとの通信の流れを説明する。ここではかなた望遠鏡を制御している PC をサーバ、オートガイダーを制御している方をクライアントとして話を進めていく。

1. socket 関数を呼び出してソケットを作成する。
2. connect 関数を呼び出してサーバに接続し、コネクションを確立する
3. recv もしくは send 関数を呼び出し、送受信を行う。
   この間サーバとクライアントのどちらかが送信側もう一方が受信側という状態である必要があるので予めどちらが先に受信側になり、もう一方が受信側になるか決めておかなくてはならない。
4. close 関数を呼び出して通信を終了させる。

ここで出てきた諸関数については付録 A を参照のこと。

第3章　測定と性能評価

3.1　試験撮影

CCD カメラの撮像用プログラムが完成したところで試験稼働を東広島天文台にて行った。次の図はその試験画像で、月を撮影したものである。全体的に左側が暗くなっているのは月の欠けている部分を撮影したからである。オートガイドのために重要なのは重心を求めるための光量を検知できることであるが、クレーターの形、影の有無、月の満ち欠けに対応した画像の明暗を見てとれることからこれは問題がないことがわかる。また、この試験稼働から画像に対応するステージと赤径赤緯の座標系の軸の位置関係も判明した。次の図 3.1、3.2 は試験稼働で撮影した月の画像と座標系の位置関係を示した図である。

図 3.1: テスト撮像画像

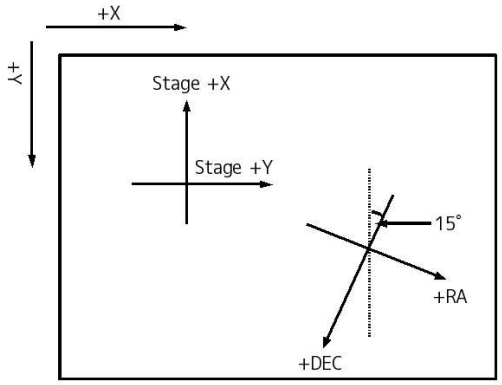

図 3.2: 座標系の位置関係

図 3.2 の左上にある矢印が GUI の星像の表示画面の座標系である。左上を原点として画面の下向きが Y の正方向、右向きが X の正方向である。これに対し、ステージの座標系

については赤緯の正方向が画面の Y 軸の正方向と 15°の角度を為している。ここでは、DEC は赤緯、RA は赤径を示す。また、次の表にこれらの座標系のスケールを示す。

表 3.1: 座標系のスケール

座標系	実スケール	設計値
赤径, 赤緯	12.5 pix/arcsec	13.9 pix/arcsec
ステージ X,Y	314 pix/10000pulse	310 pix/10000puls (1/20 の時)

3.2 試験駆動

オートガイダー制御用のプログラムが一通りできあがったところで東広島天文台にて試験観測を行った。2 月 4 日はガイド星として HD43384(V 等級 6.25) を指定し、2 月 5 日は双子座 SAO96084(V 等級 6.4) をガイド星と指定した。

3.2.1 2 月 4 日

まず、オートガイダーを用いてガイド星を監視し、検出したズレ量をマニュアルでかなた望遠鏡へオフセットをかけるという方法でズレの補正を行った。これは異常な値をかなた望遠鏡に送ることで不具合を生じさせることがないように万全を期してのことである。この補正はうまくいったのでオートガイダーでは正確にズレを検出できていることが確認できた。この後、かなた望遠鏡へのフィードバックをかける部分を実装してこの日の試験駆動は終了した。

3.2.2 2 月 5 日

この日はオートガイドモードを用いて初観測を行った。一度目に観測を開始したところガイド星が検出可能領域からどんどんズレていったので一旦観測を中止し、プログラムの見直しをした。すると、フィードバックをかける部分で送信するズレ量の正負が意図と逆になっていることが判明した。そこで RA、DEC 両方向で変えてみたところ正常にオートガイドモードが動作した。以降、この日はオープントラックで 4 回、オートガイドモードを用いて 9 回、計 13 回の観測を行った。ガイド星はいずれも双子座 SAO96084 を指定した。

3.3 追尾機能

2 月 5 日の試験観測では、実際にかなた望遠鏡にオートガイダーで検出したズレ量をフィードバックして望遠鏡の制御を行った。一方、比較のためにフィードバックを行わないとき (オープントラック) の望遠鏡の挙動についても観測の最初、途中 (2 回)、最後にデータ収得を行った。

3.3.1 オープントラックでの追尾精度

比較となるオープントラックでの追尾精度についてまず確認しておく。図 3.3 はその時の重心のズレの時間変化をプロットしたものである。これを見るとズレが直線的に変化するという傾向があることが明らかである。そこで、この変化を直線的な系統成分とそれ以外のランダム成分とに分離して考察することにする。ここでは、プロット点に対して直線をフィッティングした。それも併せて図 3.3 に示してある。

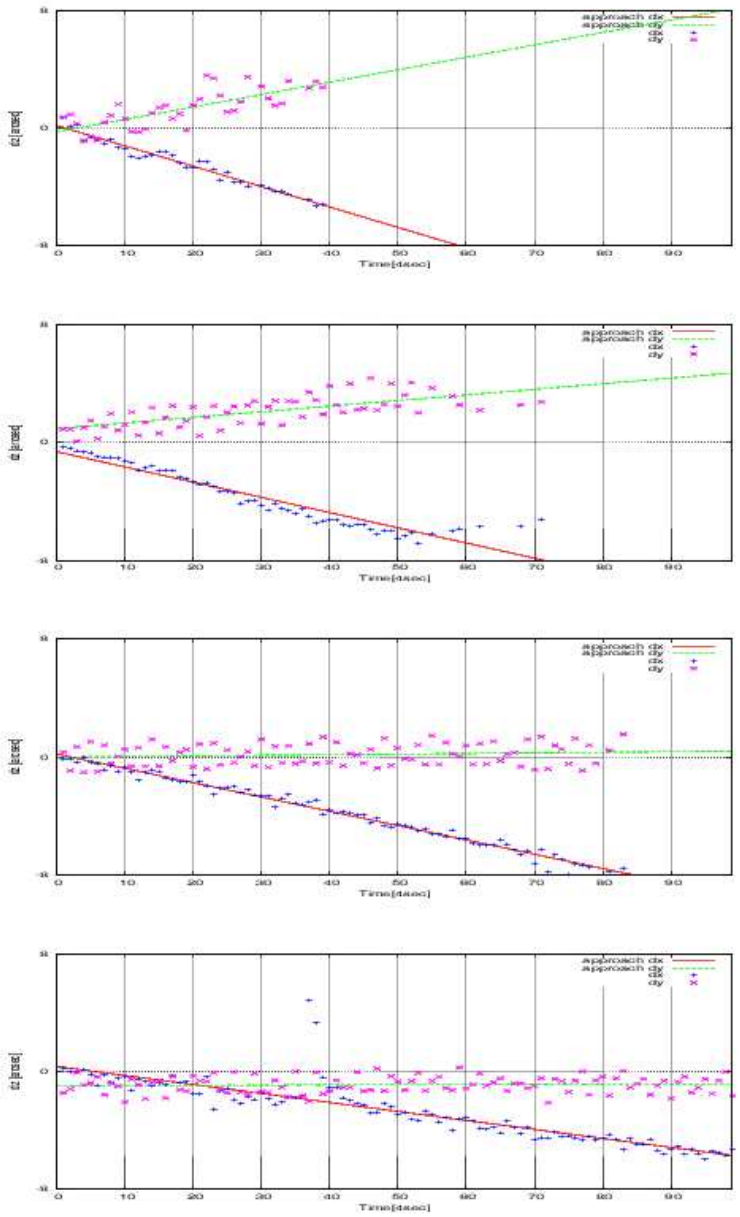

図 3.3: 重心のズレの時間変化
上から順に **1** 回目 ∼**4** 回目までのオープントラックでの観測
縦軸 δz**(arcsec)**、横軸ループ回数 **(4sec/1loop)**
$-8 \leq \delta z \leq 8, 0 \leq roop \leq 99$
1 回目**:**方位角**-12**゜、高度 **67**゜、**4** 回目**:**方位角**-62**゜、高度 **52**゜

ランダム成分をプロットしたのが次の図 3.4 である。

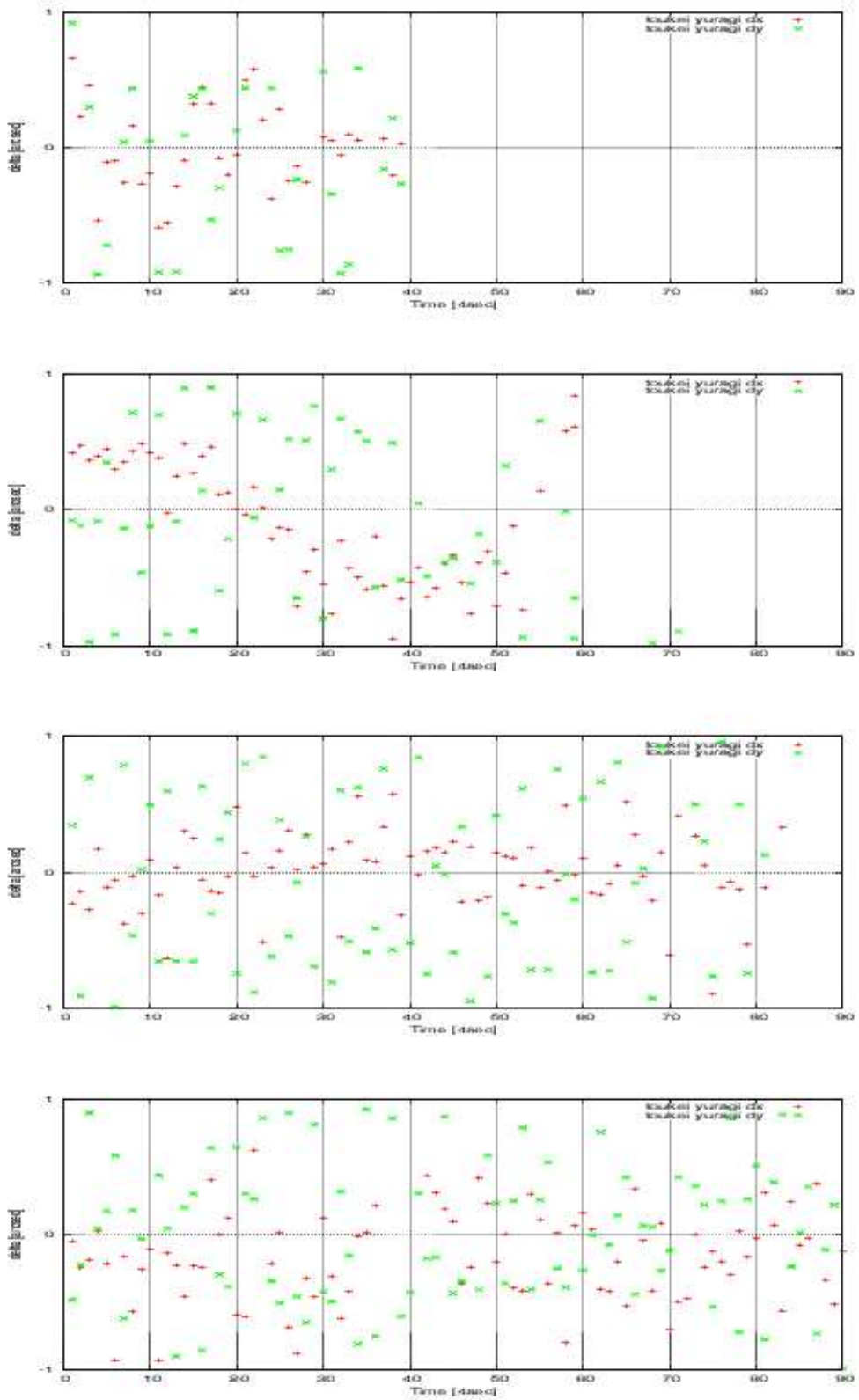

図 3.4: 重心のズレと残差
上から順に **1** 回目 ∼**4** 回目までのオープントラックでの観測
縦軸 δz**(arcsec)**、横軸ループ回数 **(4sec/1loop)**
$-1 \leq \delta z \leq 1, 0 \leq roop \leq 99$

のズレは後に考察するが、大きすぎるとオートガイダーを用いて望遠鏡の補正を行ってもすぐに観測装置の視野外へ星像が外れてしまうので今回の方法でのオートガイドが意味をなさないことになる。しかし、今回は ±1” より充分小さいということが判明し、今回の方法によるオートガイドで充分に補正できる見通しが立った。次の表はオープントラック観測での系統ズレ量とランダム成分の標準偏差をまとめたものである。

表 3.2: オープントラックのズレ

	系統ズレ		ランダム成分 (σ)	
観測回数	α(arcsec/秒)	δ(arcsec/秒)	α(arcsec)	δ(arcsec)
1	-0.0346	+0.0211	0.310	0.822
2	-0.0258	+0.0095	0.649	0.738
3	-0.0243	+0.0011	0.295	0.716
4	-0.0153	+0.0003	0.967	0.562
平均	-	-	0.555	0.710

3.3.2 オートガイドをかけた場合の挙動

図 3.5 は、move factor を 0.1、 0.3、 0.5、 0.7、 0.9 にしてオートガイドを行った時の星像の重心の挙動と x、y 方向のズレの時間変化を表したものである。

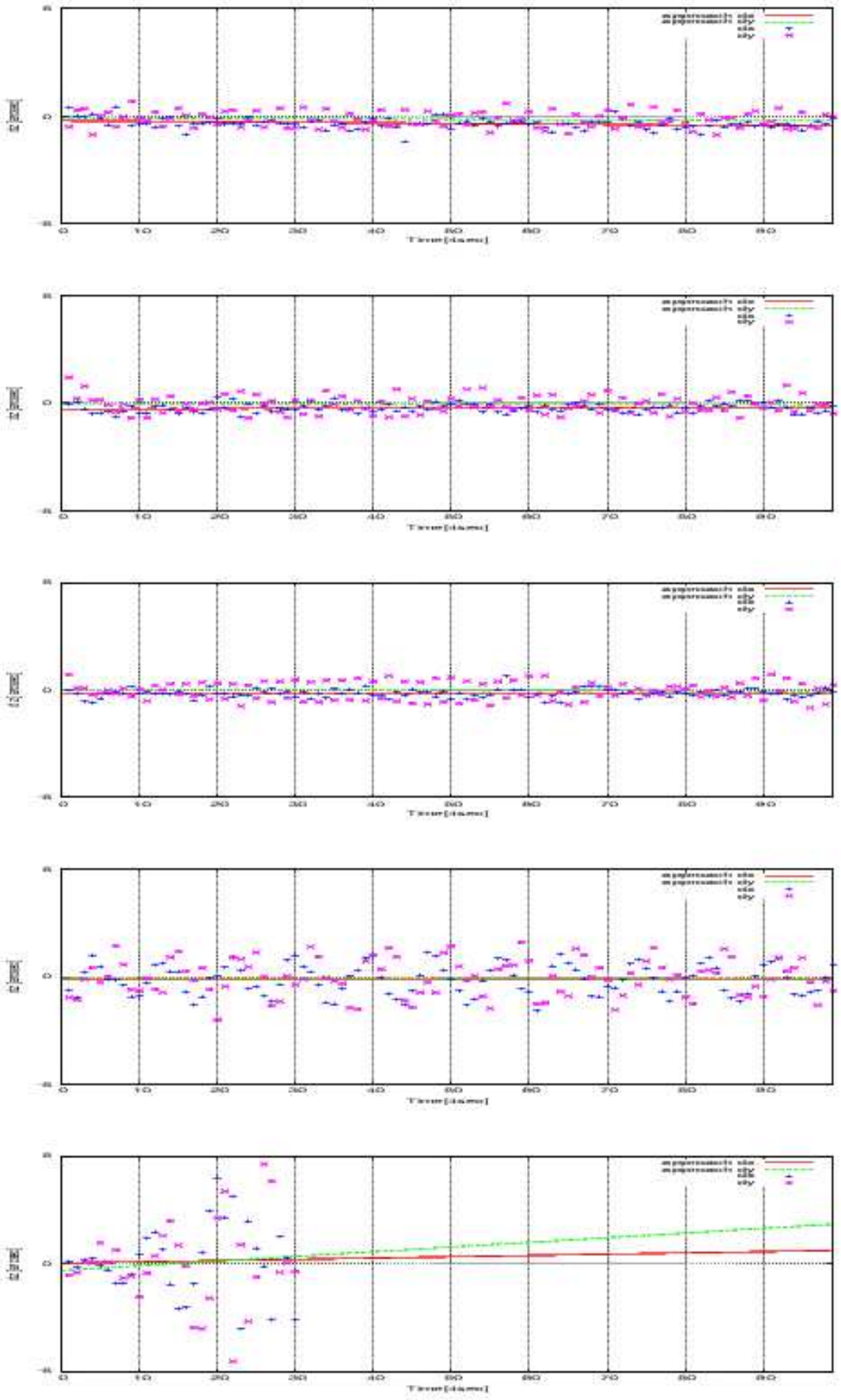

図 3.5: ズレの時間変化
縦軸 δz**(arcsec)(**$-8 \leq \delta z \leq 8$**),** 横軸 **Time(4sec)(**$0 \leq Time \leq 99$**)**
上から **move factor 0.1 0.3 0.5 0.7 0.9**

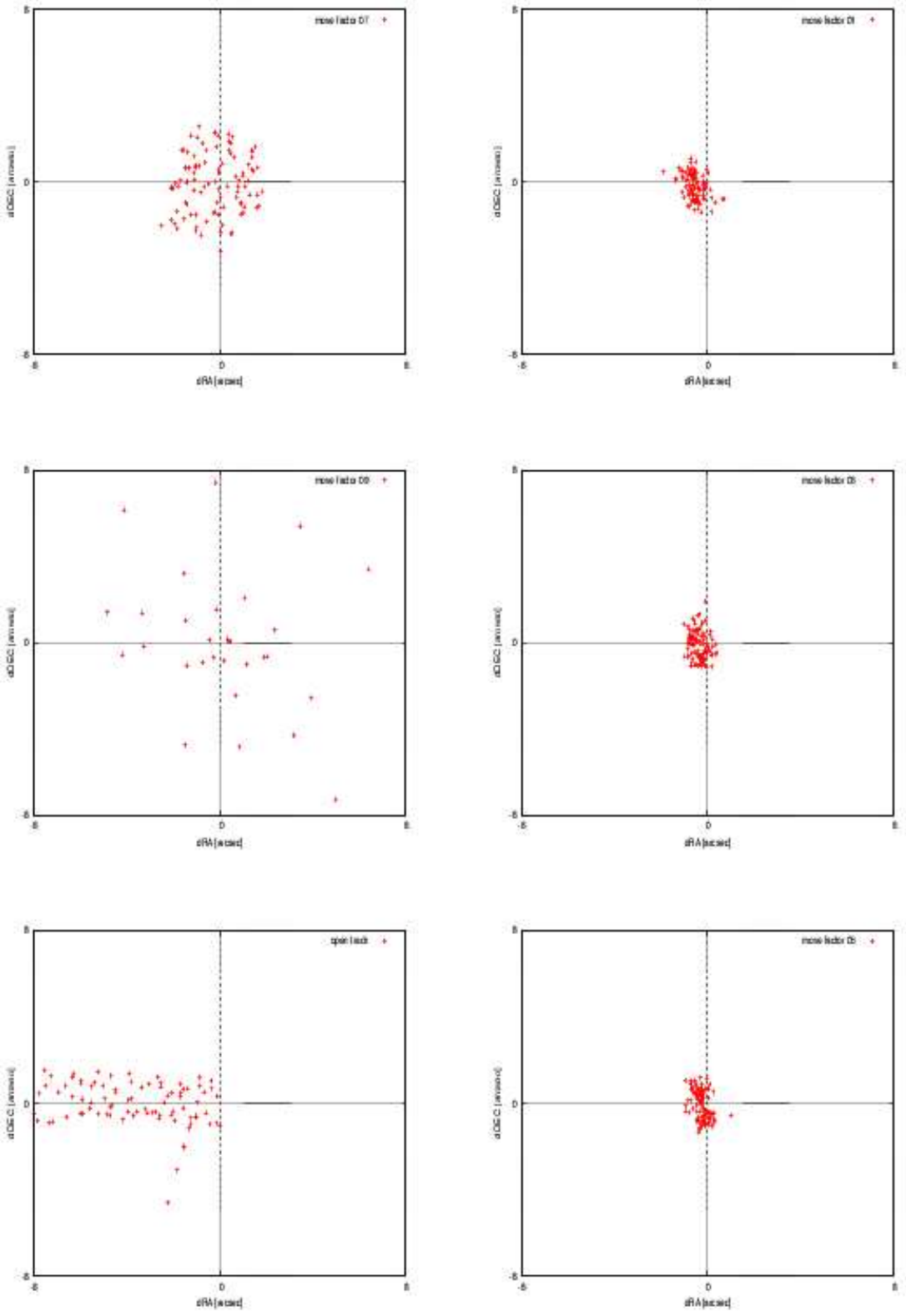

図 3.6: 重心の挙動
右上 **move factor 0.1**、右中 **0.3**、右下 **0.5**、左上 **0.7**、左中 **0.9**、左下 オープントラック
縦軸 **x(arcsec)**、横軸 **y(arcsec)**
$-8 \leq x, y \leq 8$

ていることがわかる。これからオートガイダーが正常に動作し、追尾誤差を補正していることがわかる。次に図 3.4 を見ると、move factor が 0.1 ～ 0.7 のときは重心が一定の領域で安定していることがわかる。$move\ factor = 0.9$ のときは重心の挙動が大きく、オートガイド開始 2 分程度で重心の検出領域外へ星像が外れてしまった。$move\ factor = 0.1$ のときは検出領域の中心から少し赤径の負の方向へズレた位置で重心が安定した。表 3.3 はズレの平均と統計ズレ量からの残差を観測順にまとめたものである。

表 3.3: オートガイドを掛けた場合のズレ

		ズレ量の平均		ランダム成分 (σ)	
観測順	move factor	Δx(arcsec)	Δy(arcsec)	α(arcsec)	δ(arcsec)
1	0.7	-0.234	-0.052	1.126	1.126
2	0.9	+0.189	+0.417	2.640	3.097
3	0.5	-0.256	-0.057	0.308	0.658
4	0.3	-0.372	-0.096	0.322	0.647
5	0.1	-0.542	-0.233	0.371	0.583

表 3.3 を見るとズレの平均値は move factor が 0.3、0.5 くらいで最小となっている。またランダム成分は move factor 0.3、0.5 のときに最小をとることがわかる。このランダム成分の値はオープントラックで観測したときの平均に近い。以上のことから move factor の値は 0.3 ～ 0.5 が最適であることがわかった。

move factor がこの値に同定できるのは前節 3.3.2 でも紹介した系統成分とランダム成分の寄与によるものである。系統成分とはポインティングアナリシスの不十分さによる指向精度の悪さや長周期エンコーダ誤差を反映した追尾誤差と考えられ、オートガイドによる補正は主にこの誤差を少なくすることを狙ったものである。一方、ランダム成分は大気の擾乱による星像位置の揺れや望遠鏡駆動系の不完性を反映した成分と考えられる。それではなぜ 0.1 や 0.7~0.9 といった数字ではいけないのかを説明していく。まず、0.1 の場合だと表 3.3 のズレ量の平均が悪いという点で 0.3 ～ 0.5 といった値におとる。これに加えて、もともとのズレ量よりも小さい量しか補正せず、望遠鏡が予定していた位置 (今の場合図 3.6 のグラフ中央) まで戻りきらずに少しズレた位置で安定してしまう。以上の理由から 0.1 は今回開発したオートガイダーには適合しない。次に 0.7 ～ 0.9 の場合では、図 3.6 の左上、左中の図を見てもらうとわかるが重心の位置が不安定になる (0.9 に至っては途中で重心の検出領域から星像が外れてしまう)。これは、ズレの検出から望遠鏡へのフィードバックにタイムラグがあるので検出時のズレと望遠鏡が補正を始める時のズレに差異が生じる。これによって望遠鏡が予定していた位置より行き過ぎる状況が生まれる。その上、0.7 ～ 0.9 ではランダム成分が大きい (表 3.3 参照) ので重心が安定せず、オートガイダーの move factor としては適合しない。理想的にはオートガイダーは系統成分のみを検出し、それに対して補正を行うことができると良いので、望遠鏡を向ける方位で系統成分の近似直線を導出し、検出した時刻から補正開始時タイムラグを考慮したズレ量の予測値をを望遠鏡に送るようにできると move factor を定義する必要無く、精度の良い追尾を実現できると考えられる。

move factor を 0.3 ～ 0.5 に設定して観測することで系統ズレ量はほぼ 0 となり、ランダム成分によるズレも 0.7 秒角程度に抑えられる。これはスリットの半幅 1.1 秒角ないしは

の観測を可能とする。

第4章　まとめ

私はかなた望遠鏡ナスミス焦点常設の広視野偏光撮像器 HOWPol 立ち上げの一貫として、これの補助観測機器であるオートガイダーの制御システムの開発に取り組んだ。

CCD カメラを乗せた X,Y ステージを自在に駆動させられるようなプログラムを C++言語を用いて作成するとともに、CCD カメラで撮像したデータを取り込み、PC 上へ画像を表示し、輝度分布の重心の検出するプログラムを作成した。さらにその輝度分布のうち、50×50 ピクセルの領域で重心、カウント、重心のズレを検出し、そのズレ量を補正するよう望遠鏡に駆動命令を送ることによって自動追尾させるシステムを構築した。これにより、目標であったオートガイド機能が働くようになり、かなた望遠鏡ナスミス焦点での長時間自動観測が可能となった。また、GUI を駆使したプログラムを作成したことで、視野中の適当な星像をクリックすればすぐにその星像を用いたオートガイドが開始できるという便利な機能も実装できた。

また、オートガイドの性能を試験観測を通じて調査した。オートガイドを掛けた場合と掛けなかった場合との星像重心の時間変化、およびオートガイドのパラメタ (move factor) をいくつか変えて時間変化を取得し、そのズレ量の系統的成分とランダム成分とを比較して、最適な move factor(求めたズレ量に対する、フィードバック量の比) の値が 0.3～0.5 であることを求めた。この場合の追尾誤差は、0.7”程度であり、充分な性能が得られている。

今回は星像の輝度分布の半値幅をガウス関数フィットにより自動的に求めるルーチンが未解決のバグのため完成までは至らなかった。また、恒星カタログからガイド星と観測星の情報を検索、表示できるプログラムも未完成である。今後はなるべく早くこれらの機能の実装を目指して、より便利かつ有用なオートガイドシステムの完成を目指したい。

第5章　謝辞

本研究にあたり東広島天文台への移動、装置の設置につきあっていただいたり、装置の仕様やプログラミングについて懇切丁寧に教えてくださった川端先生に心より感謝いたします。また、ソフトウェア作成でのアドバイスをしてくださった深沢先生、植村さん、可視赤外線カメラについてアドバイスをしてくださった宮本さん、松井さん、HOWPol のことでアドバイスをいただいた田中さん、自身の論文作成で忙しい中ソフトウェア開発についてアドバイスをいただいた保田さんにも心より感謝いたします。

環境面では、研究室のみなさまの明るい雰囲気のおかげで充実した研究生活を送ることができました。夜遅くまで一緒に研究にせいをだしたみなさまに心より感謝いたします。

References

[1] William H. Press「NUMERICAL RECIPES in C」(技術評論社, 1993)

[2] 千代延真吾 「かなた望遠鏡用 1 露出型偏光撮像装置 HOWPol の筐体及び駆動機構の開発」(広島大学 修士論文 2006)

[3] 千代延真吾 「広島大学 1.5m 望遠鏡移設地シーイングのモニター装置開発と測定」(広島大学 卒業論文 2004)

[4] 「AL シリーズ USB マスターユニット CB-23/USB 取扱い説明書」(Melec)

[5] 「AL シリーズ汎用入出力 CB-34/IO 取扱い説明書」(Melec)

[6] 「ステッピングモータコントロータドライバ CD-773/ADB5331A 取扱い説明書」(Melec)

[7] 「STEPPING & SERVO MOTOR CONTROLLER'S OPTION MPL-28/ALUSBWXP 取扱説明書 (デバイスドライバ AL MCC05 ユニット編)」(Melec)

付 録A

A.1　ステージ制御用コマンド

1. 00H:NO OPERATION COMMAND
   STATUS1 PORT の DREND BIT と ERROR BIT をクリアする

2. 01H:SPEC INITIALIZE1 COMMAND
   動作仕様を定義する。

3. 03H:ADDRESS INITIALIZE COMMAND
   現在位置を指定された絶対 ADDRESS として、定義･記憶する。

4. 06H:RATE SET COMMAND
   加速･減速に必要な加速時定数、減速時定数を設定する。

5. 07:LSPD SET COMMAND
   DRIVE に必要な LOW SPEED を設定する。

6. 08:HSPD SET COMMAND
   DRIVE に必要な HIGH SPEED を設定する。

7. 12H & 13H:+/-SCAN COMMAND
   SCAN DRIVE を行う。12H で+(CW) 方向、13H で-(CCW) 方向。

8. 14H:INCREMENTAL INDEX COMMAND
   相対的なパルス指定による DRIVE を行う。

9. 1EH:ORIGIN COMMAND
   ステージの原点を検出し、そこまでの DRIVE を行う。

10. FFH:FAST STOP COMMAND
    DRIVE を即時停止させる。

A.2　ステージ制御用関数

ステージ制御用関数

前述のコマンドを次に示す関数に代入することで各種のドライブやステータスの読み出しを行う。ここではまず、関数内で用いられる主要な引数について説明する。

hDev → デバイスハンドルを格納するための変数ポインタ
psResult → 関数の実行結果を格納するためのポインタ
Cmd → コマンドコード
psData → 書き込むデータが格納されているポインタ

pData→読み込んだデータを格納するためのポインタ

1. デバイスオープン関数
   書式:AC05_BOpen(WORD IfNo, WORD SlaveAddr, WORD Axis, WORD SlaveType,
   DWORD FAR *phDev, AC05_S_RESULT FAR *psResult)
   引数:IfNo → AC05_USB
   SlaveAddr→ スレーブアドレス
   Axis→X 軸は AC05_X、Y 軸は AC05_Y、Z 軸は AC05_Z。
   SlaveType→ スレーブタイプを指定、今回は AC05_SLAVE_CD773 を使用
   phDev、psResult

2. デバイスクローズ関数
   書式:AC05_BClose(DWORD hDev, AC05_S_RESULT FAR *psResult)
   引数:phDev、psResult

3. DRIVE COMMAND 一括書き込み関数
   書式:AC05_IWDrive(DWORD hDev, DWORD Cmd, AC05_S_DATA FAR *psData, AC05_S_RESULT
   FAR *psResult)
   引数:hDev、Cmd、psData、psResult

4. DRIVE DATA PORT 一括書き込み関数
   書式:AC05_IWData(DWORD hDev, AC05_S_DATA FAR *psData, AC05_S_RESULT
   FAR *psResult)
   引数:hDev、psData、psResult

5. DRIVE COMMAND PORT 一括書き込み関数
   書式:AC05_BWDriveCommand(DWORD hDev, WORD FAR *pCmd, AC05_S_RESULT
   FAR *psResult)
   引数:hDev、pCmd、psResult

6. STATU1 PORT 読み出し関数
   書式:AC05_BRStatus1(DWORD hDev, WORD FAR *psStatus, AC05_S_RESULT FAR
   *psResult)
   引数:pStatus→ 読み出した内容が格納されるポインタ
   hDev、psResult

7. STATUS2 PORT 読み出し関数
   書式:AC05_BRStatus2(DWORD hDev, WORD FAR *psStatus, AC05_S_RESULT FAR
   *psResult)
   引数:hDev、pStatus、psResult

書式:AC05_IRDrive(DWORD hDev, AC05_S_DATA FAR∗psData, AC05_S_RESULT FAR ∗psResult)
引数:hDev、psData、psResult

9. データセット関数
書式:AC05_SetData(DWORD Data, AC05_S_DATA FAR ∗psData)
引数:Data→24 ビットのデータを代入
psData

10. データゲット関数
書式:AC05_GetData(AC05_S_DATA FAR ∗psData)
引数:psData

A.3 通信用ポート

1. STATUS1 PORT
各々の軸の現在の状態を読み出す為のポート

D^7	D^6	D^5	D^4	D^3	D^2	D^1	D^0

D^0:BUSY BIT
0 で対応する軸へ COMMAND が書き込み可能かを示す。
ここが 0 でないと COMMAND の書き込みができない。
D^1:DRIVE BIT
1 で対応する軸が DRIVE 中であることを示す。
D^2:DREND BIT
1 で対応する軸の DRIVE が終わったことを示す。
D^3:ERROR BIT
書き込まれた COMMAND か DATA にエラーがあったことを示す。
D^4:未使用
D^5:LSEND BIT
$D^1 = 1$ の時、リミット信号が入力されたことを示す。
D^6:SSEND BIT
$D^1 = 1$ の時、FAST STOP COMMAND が入力されたことを示す。
D^7:今回は未使用

2. STATUS2 PORT
各々の軸の入力信号の状態を読み出す為のポート各々の軸の現在の状態を読み出す為のポート

D^7	D^6	D^5	D^4	D^3	D^2	D^1	D^0

D^0:未使用
D^1:$\overline{ORG}$ 信号

D^2:未使用
D^3:CWLM 信号
CW 方向の LIMIT の状態を示す。
D^4:CCWLM 信号
CCW 方向の LIMIT の状態を示す。
D^5:未使用
D^6:未使用
D^7:未使用

ステータスポートは 1～5 まであるが 3～5 は本研究では未使用。

A.4　重心の導出

本研究で開発したシステムでは星像の座標と各ピクセルのカウント数から星の重心を導出した。その手法はいかの通りである。

1. 撮像領域の四隅の 10×10 ピクセルの領域のカウントを求め、カウント数の平均を求めてこれを SKY とする。
2. 1 で求めたカウントを全撮像領域から引き、これを光度の分布とする。
3. 2 の分布を x、y それぞれの方向について 1 次元化する
4. 3 で作った 1 次元化された分布と星の座標を用いて重心を求める。重心を求めるアルゴリズムは次の通りである。

$$x_G = \frac{\sum(x_i \times Count_i)}{\sum Count_i} \quad \text{(A.4.1)}$$

上の手順で x、y それぞれについて重心を求める。

A.5　FWHM の導出

FWHM の導出

今回開発したオートガイダーでは星像のシーイングも監視できるように FWHM もカウントと座標から導出できるようにプログラムを組んである。撮像した星像は 2 次元であるので FWHM も x、y それぞれの方向について求める。本研究では撮像した星のカウントはすべてガウス分布に従うものとして各処理を行う。
ガウス分布はある値 μ を中心にして左右対称な曲線となり、x 軸を漸近線とし、次式で表される。

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \quad \text{(A.5.1)}$$

星像の場合、分布の中心 μ は重心 x_G にあたる。ただし今回の場合、規格化されていないガウス分布関数を求めたいのでガウス分布関数を

$$G(x) = A\exp(-\frac{(x-\mu)^2}{2\sigma^2}) \quad \text{(A.5.2)}$$

と置くことにする。A と σ はフィッティングパラメタである。
半値全幅、通称 FWHM はガウス分布の最大値のちょうど半分の値をとるところの幅で、ガウス分布の標準偏差 σ と次のような関係を持つ。

$$FWHM = 2\sigma\sqrt{2\ln 2} \quad \text{(A.5.3)}$$

FWHM を導出するためにまずこの σ を求めることにする。σ を求めるのにまず、x、y それぞれの方向について 1 次元化した分布を作り、その分布をガウシアンでフィッティングするという方法をとる。
C++のプログラムによるフィッティングとしてはマーカート法 (Levenberg-Marquardt 法) を用いる。次にマーカート法について述べる。

マーカート法 (Levenberg-Marquardt 法)

1. フィットしたいモデル (ガウス分布関数) を決める。
2. χ 評価関数

$$\chi^2(a) = \sum_{i=1}^{N}[\frac{y_i - f(x;a)}{\sigma_i}]^2 \quad \text{(A.5.4)}$$

   を求める。a はフィッティングパラメタ A と σ。
3. 2 次のベクトル β と 2×2 行列 B の kl 成分 α_{kl} を求める。α、β については以下のように求める。

$$\beta_k = -\frac{1}{2}\frac{\partial\chi^2}{\partial a_k} \quad \alpha_{kl} = \frac{1}{2}\frac{\partial^2\chi^2}{\partial a_k \partial a_l} \quad \text{(A.5.5)}$$

   ただし、k = A,σ、l = A,σ。
4. ベクトル $\delta a_l = \frac{1}{\lambda\alpha_{ll}}$ を計算する。ここで λ は適当な値を代入する。今回は $\lambda = 0.001$ を用いた。
5. $a + \delta a_l$ を χ 評価関数に再度代入して χ^2 を求める。

2～5 を 1 回のループごとに λ 変えて (例えば 10 分の 1 等) 繰り返し計算させ、χ^2 の値に制限を設け、その値までループを行う。今回行ったフィッティングでは、$\chi^2(a+\delta a_l) \geq \chi^2(a_l)$ であれば λ を 10 倍し、$\chi^2(a+\delta a_l) < \chi^2(a_l)$ であれば λ を 1/10 倍してループさせた。
上の 1～5 が一般的なマーカート法の流れである。では次から今回用いた、フィッティングパラメタが 2 つの時のマーカート法について詳しく解説していく。
まず、一般的な χ^2 の関数は

$$\chi^2(a) = \gamma - d\cdot a + \frac{1}{2}a\cdot D\cdot a \quad \text{(A.5.6)}$$

と近似できる (a は変数、ここではフィッティングパラメタを指す)。ここで任意の a の値を a_{cur}、χ^2 を最小とする a を a_{min} とすると a_{min} は $\nabla\chi^2 = -d + D\cdot a = 0$ より、$a_{min} = \frac{d}{D}$ これから、

$$a_{min} = a_{cur} + D^{-1}[-\nabla\chi^2(a_{cur})] \quad \text{(A.5.7)}$$

を a_{cur} の近くの点とすると、

$$a_{next} = a_{cur} - (\text{定数}) \times \nabla\chi^2(a_{cur}) \quad \text{(A.5.8)}$$

と書くことができる。この式を a_{next} が χ^2 を最小とするような値まで絞り込んでいくのがフィッティングという作業である。しかし、上の式はこのままではプログラムに組み込めないのでもっと使い易い形にしていく。
ではまず、当てはめたいモデル関数を決める。ここではガウシアンを用いる。このとき χ^2 は

$$\chi^2(a) = \sum_{i=1}^{N} [\frac{y_i - y(x_i;a)}{\sigma_i}]^2 \quad \text{(A.5.9)}$$

と置くことができ、ガウシアンを用いた場合は

$$\chi^2 = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} [C - A\exp(-\frac{(x-X)^2}{2\sigma^2})]^2$$

と置ける。ここで、x はピクセル座標、X は重心の座標、C は座標 x でのカウント数、σ は星像の輝度分布の標準偏差である。
パラメータ a についての χ^2 の勾配は χ^2 が最小のときに 0 になるので

$$\frac{\partial\chi^2}{\partial a_k} = -2\sum_{i=1}^{N} \frac{1}{\sigma_i^2} [y_i - y(x_i;a)] \frac{\partial y(x_i;a)}{\partial a_k} \quad \text{(k = A,}\sigma\text{)} \quad \text{(A.5.10)}$$

これをさらに偏微分すると

$$\frac{\partial^2\chi^2}{\partial a_k \partial a_l} = 2\sum_{i=1}^{N} \frac{1}{\sigma_i^2} [\frac{\partial y(x_i;a)}{\partial a_k} \frac{\partial y(x_i;a)}{\partial a_l} - [y_i - y(x_i;a)] \frac{\partial^2 y(x_i;a)}{\partial a_l \partial a_k}]$$

しかし、実際には 2 階導関数の部分は 0 と見てよいので

$$\frac{\partial^2\chi^2}{\partial a_k \partial a_l} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} [\frac{\partial y(x;a)}{\partial a_k} \frac{\partial y(x_i;a)}{\partial a_l}] \quad \text{(A.5.11)}$$

となる。慣習的に (A.5.10) 式と (A.5.11) 式は次のように置かれる。

$$\beta_k = -\frac{1}{2}\frac{\partial\chi^2}{\partial a_k}, \quad \alpha_{kl} = \frac{1}{2}\frac{\partial^2\chi^2}{\partial a_k \partial a_l}$$

これをフィッティングパラメタ A,σ について解くと

$$\beta_A = \sum_{i=1}^{N} \frac{1}{\sigma_i} (C - G(x)) \exp(-\frac{(x-X)^2}{2\sigma^2})$$

$$\beta_\sigma = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{(x-X)^2}{\sigma^3} (C - G(x)) G(x)$$

$$\alpha_{AA} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \exp(-\frac{(x-X)^2}{\sigma^2})$$

$$\alpha_{\sigma\sigma} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{(x-X)^4}{\sigma^3} G(x)^2$$

$$\alpha_{A\sigma} = \alpha_{\sigma A} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{A(x-X)^4}{\sigma^3} \exp(-\frac{(x-X)^2}{\sigma^2})$$

となる。すると $[\alpha] = {}_2D$ より式 (A.5.6) は次のように書ける。

$$\sum_{l=A,\sigma} \alpha_{kl}\delta_l = \beta_k \quad \text{(A.5.12)}$$

また、式 (A.5.8) は次のように書くとする

$$\delta a_l = (\text{定数}) \times \beta_l \quad \text{(A.5.13)}$$

マーカート法においては β_k は $1/a_k$ と同じ次元を持つので式 (A.5.13) の定数は ${a_k}^2$ の次元を持ち、$[\alpha]$ の中ではこれと同じ次元を持つものは対角要素の逆数 $1/a_{kk}$ だけである。よってこれが式 (A.5.13) の定数の大きさの目安となる。しかしこの目安は大きすぎるかもしれないので定数を結果に影響を与えないファッジファクター λ で割っておく、こうすることで $\lambda \gg 1$ と置いて一歩の大きさを小さくもできる。つまり式 (A.5.13) を

$$\delta a_l = \frac{1}{\lambda \alpha_{ll}}\beta_l \quad \text{(A.5.14)}$$

と置き換える。これにともなって新しい行列 $[\alpha']$ を次の様に定義する。

$$\alpha'_{ii} = \alpha_{ii}(1+\lambda)$$
$$\alpha'_{ij} = \alpha_{ji} \quad (i \neq j)$$

こうすることで式 (A.5.12) が

$$\sum_{l=a,\sigma} = \alpha'_{kl}\delta a_l = \beta_k \quad \text{(A.5.15)}$$

と置ける。この表式を用いて最初に紹介した 1 ～ 5 の手順を繰り返し行うことをマーカート法と呼ぶ。

A.6 ソケット通信用関数

1. socket 関数
   書式: SOCKET socket(af, type, protocol);
   引数:
   int af → アドレスフォーマット。AF_INET(インターネット) と AF_UNIX(端末内) のいずれかを指定。今回は USB を通じて PC 同士を接続するので AF_INET を使用した。
   int type → ソケットタイプ。ストリームソケットを使用しているので SOCK_STREAM を指定。
   int procotol → ソケットプロトコル。type に SOCK_STREAM を指定したので IPPROTO_TCP を指定。

2. connect 関数
   書式:int connect(sock, addr, len);
   引数:
   SOCK sock → 受け入れるソケットを指定
   struct sockaddr *addr → 接続するアドレスを格納した sockaddr 構造体へのポインタを指定
   int len → sockaddr 構造体のサイズを指定

3. send 関数
   書式:int send(sock, buf, len, flag);
   引数:
   SOCKET sock → 送信のためのソケットを指定
   char *buf → 送信データバッファ
   int len → 送信データサイズを指定
   int flag → 呼び出し方法を指定

4. recv 関数
   書式:int recv(sock, buf, len, flag);
   引数:
   SOCKET sock → 受信のためのソケットを指定
   char *buf → 受信バッファ
   int len → 受信バッファサイズを指定
   int flag → 呼び出し方法を指定

付 録B

B.1 ソースコード

```
// AutoGuiderDlg.cpp : 実装
ファイル

#include <stdlib.h> include "windows.h" include "stdafx.h" include
#"AutoGuider.h" include "AutoGuiderDlg.h" include "communi.h" include
#<math.h> include <windows.h>

#ifdef _DEBUG define new DEBUG_NEW undef THIS_FILE static char
#THIS_FILE[] = __FILE__; endif


// アプリケーションのバージョン情報に使われる CAboutDlg ダイアログ

class CAboutDlg : public CDialog { public: CAboutDlg();

// ダイアログ データ enum { IDD = IDD_ABOUTBOX };

        protected: virtual void DoDataExchange(CDataExchange* pDX); //
        DDX/DDV サポート

// 実装 protected: DECLARE_MESSAGE_MAP() };

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
END_MESSAGE_MAP()


// CAutoGuiderDlg ダイアログ



CAutoGuiderDlg::CAutoGuiderDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CAutoGuiderDlg::IDD, pParent)
        , m_Stop_AG(false)
{
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

static DWORD                            hDev1;
                                // デバイスハンドル変数
static DWORD                            hDev2;
                                // デバイスハンドル変数
static DWORD                            hDev3;
                                // デバイスハンドル変数

static DWORD                            hPort;
                                // 汎用 I/O PORT ハンドル変数
static ALK_S_RESULT                     AlkResult;
                        // RESULT 格納エリア
static ALK_S_SLAVE_INFO                 AlkSlaveInfo;
                        // SLAVE INFO 格納エリア
static AC05_S_DATA                      Ac05Data;
                        // データエリア
static AC05_S_RESULT                    Ac05Result;
                        // RESULT 格納エリア
static ACIO_S_RESULT                    AcIoResult;
                        // RESULT 格納エリア
static WORD                             Cmd;
                                // コマンド設定変数
static WORD                             Data;
                                // データ設定変数
static WORD                             Status1;
                        // スタータスエリア
static DWORD                            Signal;
                                // SIGNAL 変数
static WORD                             SignalStatus;
                        // SIGNAL ステータス変数
static BYTE                             StopCode;
                        // STOP CODE
static WORD                             StopFlag;
                        // ストップフラグ
static char                             Buffer[20];
static char                             Buffer_x[20];
static char                             Buffer_y[20];
static char                             Buffer_z[20];
static WORD                             x_flag;
static WORD                             y_flag;
static WORD                             z_flag;
static WORD                             Status1_x;
static WORD                             Status1_y;
static WORD                             Status1_z;
static AC05_S_DATA                      Ac05Data_x;
static AC05_S_DATA                      Ac05Data_y;
static AC05_S_DATA                      Ac05Data_z;
static WORD                             Status2_x;
static WORD                             Status2_y;
static WORD                             Status2_z;

void CAutoGuiderDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        DDX_Control(pDX, IDC_OPEN, m_open);
        DDX_Control(pDX, IDC_CLOSE, m_close);
        DDX_Control(pDX, IDC_END, m_end);
        DDX_Control(pDX, IDC_STOP, m_stop);
        DDX_Control(pDX, IDC_ORIGINDRIVE, m_origindrive);
        DDX_Control(pDX, IDC_SDXPL, m_sdxpl);
        DDX_Control(pDX, IDC_SDXNE, m_sdxne);
        DDX_Control(pDX, IDC_SDYPL, m_sdypl);
        DDX_Control(pDX, IDC_SDYNE, m_sdyne);
        DDX_Control(pDX, IDC_SDTPL, m_sdtpl);
        DDX_Control(pDX, IDC_SDTNE, m_sdtne);
        DDX_Control(pDX, IDC_INDRIVE, m_indrive);
        DDX_Control(pDX, IDC_LABEL_MESSAGE, m_label_message);
        DDX_Control(pDX, IDC_ADRESS_XAX, m_adress_xax);
        DDX_Control(pDX, IDC_ADRESS_YAX, m_adress_yax);
        DDX_Control(pDX, IDC_ADRESS_ZAX, m_adress_zax);
        DDX_Control(pDX, IDC_EDITX, m_edit_x);
        DDX_Control(pDX, IDC_EDITY, m_edit_y);
        DDX_Control(pDX, IDC_EDITTHETA, m_edit_theta);
        DDX_Control(pDX, IDC_COOLER, m_Cooler);
        DDX_Control(pDX, IDC_SHOOT, m_Shoot);
        DDX_Control(pDX, IDC_CCD_STATUS, m_CCD_Temp);
        //DDX_Control(pDX, IDC_COMBO2, m_Region);
        //DDX_Control(pDX, IDC_COMBO1, m_Expo_Time);
        DDX_Control(pDX, IDC_STATIC_SCREEN, m_Static_Screen);
        DDX_Control(pDX, IDC_CGravity_Graph, m_CGravity_Graph);
        DDX_Control(pDX, IDC_CENTER_OF_GRAVITY, m_Center_of_Gravity);
        DDX_Control(pDX, IDC_SKY_LEVEL, m_Sky_Level);
        DDX_Control(pDX, IDC_AVERAGE_CREEP_X, m_Average_Creep_X);
        DDX_Control(pDX, IDC_AVERAGE_CREEP_Y, m_Average_Creep_Y);
        DDX_Control(pDX, IDC_SD_CREEP_X, m_SD_Creep_X);
        DDX_Control(pDX, IDC_SD_CREEP_Y, m_SD_Creep_Y);
        DDX_Control(pDX, IDC_AVERAGE_COUNT, m_Average_Count);
        DDX_Control(pDX, IDC_SD_COUNT, m_SD_Count);
        DDX_Control(pDX, IDC_TIMER_SET, m_Timer_Set);
        DDX_Control(pDX, IDC_CHIGH, m_CHigh);
        DDX_Control(pDX, IDC_CLOW, m_CLow);
        DDX_Control(pDX, IDC_POSITION_ANGLE, m_Position_Angle);
}

BEGIN_MESSAGE_MAP(CAutoGuiderDlg, CDialog)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        //}}AFX_MSG_MAP
        ON_BN_CLICKED(IDOK, &CAutoGuiderDlg::OnBnClickedOk)
        ON_BN_CLICKED(IDC_OPEN, &CAutoGuiderDlg::OnBnClickedOpen)
        ON_BN_CLICKED(IDC_CLOSE, &CAutoGuiderDlg::OnBnClickedClose)
        ON_BN_CLICKED(IDC_END, &CAutoGuiderDlg::OnBnClickedEnd)
        ON_BN_CLICKED(IDC_STOP, &CAutoGuiderDlg::OnBnClickedStop)
        ON_BN_CLICKED(IDC_ORIGINDRIVE, &CAutoGuiderDlg::OnBnClickedOrigindrive)
        ON_BN_CLICKED(IDC_SDXPL, &CAutoGuiderDlg::OnBnClickedSdxpl)
        ON_BN_CLICKED(IDC_SDXNE, &CAutoGuiderDlg::OnBnClickedSdxne)
        ON_BN_CLICKED(IDC_SDYPL, &CAutoGuiderDlg::OnBnClickedSdypl)
        ON_BN_CLICKED(IDC_SDYNE, &CAutoGuiderDlg::OnBnClickedSdyne)
        ON_BN_CLICKED(IDC_SDTPL, &CAutoGuiderDlg::OnBnClickedSdtpl)
        ON_BN_CLICKED(IDC_SDTNE, &CAutoGuiderDlg::OnBnClickedSdtne)
        ON_BN_CLICKED(IDC_INDRIVE, &CAutoGuiderDlg::OnBnClickedIndrive)
        ON_BN_CLICKED(IDC_COOLER, &CAutoGuiderDlg::OnBnClickedCooler)
        ON_BN_CLICKED(IDC_SHOOT, &CAutoGuiderDlg::OnBnClickedShoot)
        ON_BN_CLICKED(IDC_AUTOGUID, &CAutoGuiderDlg::OnBnClickedAutoguid)
        ON_BN_CLICKED(IDC_STOP_AG, &CAutoGuiderDlg::OnBnClickedStopAg)
        ON_WM_LBUTTONDBLCLK()
END_MESSAGE_MAP()


// CAutoGuiderDlg メッセージ ハンドラ

BOOL CAutoGuiderDlg::OnInitDialog()
{
        char buf[30], Mes[30];
        CString Mes2((const char*)Mes, 30);

        CDialog::OnInitDialog();

        // "バージョン情報..." メニューをシステム メニューに追加します。

        // IDM_ABOUTBOX は、システム コマンドの範囲内になければなりません。

        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
                CString strAboutMenu;
                strAboutMenu.LoadString(IDS_ABOUTBOX);
                if (!strAboutMenu.IsEmpty())
                {
```

```
                        pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
                }
        }

        // このダイアログのアイコンを設定します。アプリケーションのメイン ウィンドウがダイアログでない場合、
        //  Framework は、この設定を自動的に行います。
        SetIcon(m_hIcon, TRUE);                 // 大きいアイコンの設定
        SetIcon(m_hIcon, FALSE);                // 小さいアイコンの設定

        // TODO: 初期化をここに追加します。
        CCDOpenDevice();
        CCDControllerReset();
        CCDGetInfomation(NULL);

                // 環境設定関数は、アプリケーションの初期化で行うようにして下さい。
        ALK_EnvironmentInfo_Tool( &AlkResult );
        // スレーブアドレス 1 に C-772 が接続されていない場合、アプリケーションを終了します。

        ALK_SlaveInfo_Read( ALK_USB, &AlkSlaveInfo, &AlkResult );
        sprintf_s( buf, " %02x %02x %02x", AlkSlaveInfo.ALK_SlaveType[0],\
                        AlkSlaveInfo.ALK_SlaveType[1], AlkSlaveInfo.ALK_SlaveType[2] );

        if( AlkSlaveInfo.ALK_SlaveType[0] != ALK_SLAVE_CD773 ){
                sprintf_s( Mes, "スレーブアドレス1は C-773  じゃないので終了。 %s", buf );
                AfxMessageBox( Mes2, MB_ICONEXCLAMATION | MB_OK );
                CDialog::OnOK( );
        return TRUE;
        }
        if( AlkSlaveInfo.ALK_SlaveType[1] != ALK_SLAVE_CD773 ){
                sprintf_s( Mes, "スレーブアドレス 2 は C-773  じゃないので終了。 %s", buf );
                AfxMessageBox( Mes2, MB_ICONEXCLAMATION | MB_OK );
                CDialog::OnOK( );
        return TRUE;
        }
        if( AlkSlaveInfo.ALK_SlaveType[2] != ALK_SLAVE_CB34 ){
                sprintf_s( Mes, "スレーブアドレス1は CB-34  じゃないので終了。 %s", buf );
                AfxMessageBox( Mes2, MB_ICONEXCLAMATION | MB_OK );
                CDialog::OnOK( );
        return TRUE;
        }

        m_open.EnableWindow( TRUE );
        m_close.EnableWindow( FALSE );
        m_end.EnableWindow(TRUE);
        m_stop.EnableWindow( FALSE );
        m_origindrive.EnableWindow(FALSE);
        m_sdxpl.EnableWindow(FALSE);
        m_sdxne.EnableWindow(FALSE);
        m_sdypl.EnableWindow(FALSE);
        m_sdyne.EnableWindow(FALSE);
        m_sdtpl.EnableWindow(FALSE);
        m_sdtne.EnableWindow(FALSE);
        m_indrive.EnableWindow(FALSE);
        m_edit_x.EnableWindow(FALSE);
        m_edit_y.EnableWindow(FALSE);
        m_edit_theta.EnableWindow(FALSE);

        char c10[30] = "オープンしてください";
        CString Op3((const char*)c10, 30);
        m_label_message.SetWindowText(Op3);

        return TRUE;  // フォーカスをコントロールに設定した場合を除き、TRUE を返します。
}

void CAutoGuiderDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// ダイアログに最小化ボタンを追加する場合、アイコンを描画するための
//  下のコードが必要です。ドキュメント/ビュー モデルを使う MFC アプリケーションの場合、
//  これは、Framework によって自動的に設定されます。

void CAutoGuiderDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // 描画のデバイス コンテキスト

                SendMessage(WM_ICONERASEBKGND,
                reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

                // クライアントの四角形領域内の中央
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // アイコンの描画
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }

        CWnd *h = GetDlgItem(IDC_CGravity_Graph);
        CDC *pDC=h->GetDC();
        CRect graph;
        h->GetClientRect(&graph);
        pDC->FillSolidRect(graph, RGB(255, 255, 255));
        int pw,ph;
        pw=graph.right;
        ph=graph.bottom;
        CPen p(PS_SOLID,1,RGB(0,255,255));
        CPen *oldp=pDC->SelectObject(&p);

        pDC->SelectObject(oldp);
        CPen pen(PS_SOLID,2,RGB(0,0,0));
        CPen *oldpen=pDC->SelectObject(&pen);
        int cx,cy;
        cx = pw/2;
        cy = ph/2;
        pDC->MoveTo(0,cy);
        pDC->LineTo(pw,cy);
        pDC->MoveTo(cx,0);
        pDC->LineTo(cx,ph);
        pDC->SelectObject(oldpen);
        h->ReleaseDC(pDC);


        CWnd *h2 = GetDlgItem(IDC_CREEP);
        CDC *pDC2=h2->GetDC();
        CRect creep;
        h2->GetClientRect(&creep);
        pDC2->FillSolidRect(creep, RGB(255, 255, 255));
        int pw2, ph2;
        pw2 = creep.right;
        ph2 = creep.bottom;
        CPen p2(PS_SOLID,1,RGB(0,255,255));
        CPen *oldp2=pDC2->SelectObject(&p2);

        pDC2->SelectObject(oldp2);
        CPen pen2(PS_SOLID,2,RGB(0,0,0));
        CPen *oldpen2=pDC2->SelectObject(&pen2);
        int cy2 = ph2/2;
        pDC2->MoveTo(10,cy2);
        pDC2->LineTo(pw2,cy2);
        pDC2->MoveTo(10,0);
        pDC2->LineTo(10,ph2);
        pDC2->SelectObject(oldpen2);
        h2->ReleaseDC(pDC2);


        CWnd *h3 = GetDlgItem(IDC_FWHM);
        CDC *pDC3=h3->GetDC();
        CRect fwhm;
        h3->GetClientRect(&fwhm);
        pDC3->FillSolidRect(fwhm, RGB(255, 255, 255));
        int pw3, ph3;
        pw3 = fwhm.right;
        ph3 = fwhm.bottom;
        CPen p3(PS_SOLID,1,RGB(0,255,255));
        CPen *oldp3=pDC3->SelectObject(&p3);

        pDC3->SelectObject(oldp3);
        CPen pen3(PS_SOLID,2,RGB(0,0,0));
        CPen *oldpen3=pDC3->SelectObject(&pen3);
        pDC3->MoveTo(10,ph3-5);
        pDC3->LineTo(pw3,ph3-5);
        pDC3->MoveTo(10,0);
        pDC3->LineTo(10,ph3-5);
        pDC3->SelectObject(oldpen3);
        h3->ReleaseDC(pDC3);

        CWnd *h4 = GetDlgItem(IDC_COUNT);
        CDC *pDC4=h4->GetDC();
        CRect count_sum;
        h4->GetClientRect(&count_sum);
        pDC4->FillSolidRect(count_sum, RGB(255, 255, 255));
        int pw4, ph4;
        pw4 = count_sum.right;
        ph4 = count_sum.bottom;
        CPen p4(PS_SOLID,1,RGB(0,255,255));
        CPen *oldp4=pDC4->SelectObject(&p3);

        pDC4->SelectObject(oldp4);
        CPen pen4(PS_SOLID,2,RGB(0,0,0));
        CPen *oldpen4=pDC4->SelectObject(&pen4);
        pDC4->MoveTo(10,ph4-5);
        pDC4->LineTo(pw4,ph4-5);
        pDC4->MoveTo(10,0);
        pDC4->LineTo(10,ph4-5);
        pDC4->SelectObject(oldpen4);
        h4->ReleaseDC(pDC4);
}

// ユーザーが最小化したウィンドウをドラッグしているときに表示するカーソルを取得するために、
```

```
HCURSOR CAutoGuiderDlg::OnQueryDragIcon()
{
        return static_cast<HCURSOR>(m_hIcon);
}

////////////////////////////////////////////
//              関数の定義
////////////////////////////////////////////

void CAutoGuiderDlg::XAxiOpen()                         //X 軸のポー
トオープン
{
        AC05_BOpen( AC05_USB, 1, AC05_X, AC05_SLAVE_CD773,
         &hDev1, &Ac05Result );
        Cmd = 0xF9;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );
}

void CAutoGuiderDlg::YAxiOpen()                         //Y 軸のポー
トオープン
{
        AC05_BOpen( AC05_USB, 1, AC05_Y, AC05_SLAVE_CD773,
         &hDev2, &Ac05Result );
        Cmd = 0xF9;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
}

void CAutoGuiderDlg::ThetaAxiOpen()                     //θ軸のポー
トオープン
{
        AC05_BOpen( AC05_USB, 2, AC05_X, AC05_SLAVE_CD773,
         &hDev3, &Ac05Result );
        Cmd = 0xF9;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
}

void CAutoGuiderDlg::OriginDrive()                      //原点検出、
上から X 軸、Y 軸、θ軸
{
        AC05_BWaitDriveCommand( hDev1, 0x00, &Ac05Result );
        AC05_BWaitDriveCommand( hDev2, 0x00, &Ac05Result );
        AC05_BWaitDriveCommand( hDev3, 0x00, &Ac05Result );

        Cmd = 0x12;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );
        Cmd = 0x12;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
        WaitLimit();

        AC05_SetData( (DWORD)-200, &Ac05Data );
        AC05_IWDrive( hDev1, 0x14, &Ac05Data, &Ac05Result );
        AC05_IWDrive( hDev2, 0x14, &Ac05Data, &Ac05Result );
        WaitLimit();

                do{
                        AC05_SetData( 5, &Ac05Data );
                        AC05_IWDrive( hDev1, 0x14, &Ac05Data, &Ac05Result );
                        AC05_IWDrive( hDev1, 0x14, &Ac05Data, &Ac05Result );
                        //ReadyWait( &Ac05Result, &StopCode );
                        //ReadyWait2( &Ac05Result, &StopCode );
                        AC05_BRStatus1( hDev1, &Status1_x, &Ac05Result );
                        AC05_BRStatus1( hDev2, &Status1_y, &Ac05Result );
                        NowAddressDisplay( &Ac05Result );
                }while( Status1_x & Status1_y & 0x00 );
                NowAddressDisplay( &Ac05Result );
}

void CAutoGuiderDlg::Reset()
     //ADDRESS COUNTER PRESET 上から X 軸、Y 軸、θ軸
{
    AC05_SetData( 0, &Ac05Data );
    AC05_IWDrive( hDev1, 0x03, &Ac05Data, &Ac05Result );

    AC05_SetData( 0, &Ac05Data );
        AC05_IWDrive( hDev2, 0x03, &Ac05Data, &Ac05Result );

    AC05_SetData( 0, &Ac05Data );
        AC05_IWDrive( hDev3, 0x03, &Ac05Data, &Ac05Result );
}

void CAutoGuiderDlg::CCwScanXaxis()
{
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        Cmd = 0x12;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );
                        // -SCAN DRIVE COMMAND
}

void CAutoGuiderDlg::CwScanXaxis()
{
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        Cmd = 0x13;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );
                        // +SCAN DRIVE COMMAND
}

void CAutoGuiderDlg::CCwScanYaxis()
{
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        Cmd = 0x12;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
                        // -SCAN DRIVE COMMAND
}
```

```
{
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        Cmd = 0x13;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
                        // +SCAN DRIVE COMMAND
}

void CAutoGuiderDlg::CCwScanThetaaxis()
{
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        Cmd = 0x13;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
                        // -SCAN DRIVE COMMAND
}

void CAutoGuiderDlg::CwScanThetaaxis()
{
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        Cmd = 0x12;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
                        // +SCAN DRIVE COMMAND
}

void CAutoGuiderDlg::RateSet()
{
                /*** RATE SET ***/     //X 軸の RATESET
        //   DRATE : 10ms/1000Hz       URATE : 10ms/1000Hz
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
    Data = 0x15;
        AC05_BWDriveData2( hDev1, &Data, &Ac05Result );
        Data = 0x15;
        AC05_BWDriveData3( hDev1, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );

        /*** LSPD SET ***/
    //   LSPD : 1000Hz
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( 3000, &Ac05Data );
        AC05_IWDrive( hDev1, 0x07, &Ac05Data, &Ac05Result );

        /*** HSPD SET ***/
    //   HSPD : 5000Hz
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( 5000, &Ac05Data );
        AC05_IWDrive( hDev1, 0x08, &Ac05Data, &Ac05Result );

                /*** RATE SET ***/     //Y 軸の RATESET
        //   DRATE : 10ms/1000Hz       URATE : 10ms/1000Hz
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
    Data = 0x11;
        AC05_BWDriveData2( hDev2, &Data, &Ac05Result );
        Data = 0x11;
        AC05_BWDriveData3( hDev2, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );

        /*** LSPD SET ***/
    //   LSPD : 1000Hz
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( 3000, &Ac05Data );
        AC05_IWDrive( hDev2, 0x07, &Ac05Data, &Ac05Result );

        /*** HSPD SET ***/
    //   HSPD : 5000Hz
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( 5000, &Ac05Data );
        AC05_IWDrive( hDev2, 0x08, &Ac05Data, &Ac05Result );

                /*** RATE SET ***/     //θ軸の RATESET
        //   DRATE : 10ms/1000Hz       URATE : 10ms/1000Hz
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
    Data = 0x11;
        AC05_BWDriveData2( hDev3, &Data, &Ac05Result );
        Data = 0x11;
        AC05_BWDriveData3( hDev3, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );

        /*** LSPD SET ***/
    //   LSPD : 1000Hz
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        AC05_SetData( 3000, &Ac05Data );
        AC05_IWDrive( hDev3, 0x07, &Ac05Data, &Ac05Result );

        /*** HSPD SET ***/
    //   HSPD : 5000Hz
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        AC05_SetData( 5000, &Ac05Data );
        AC05_IWDrive( hDev3, 0x08, &Ac05Data, &Ac05Result );
}
UINT CAutoGuiderDlg::ReadyWait( AC05_S_RESULT *Result, BYTE *StopCode
    )           //X 軸の Message
{
        do{
                // ループ中、一旦他のタスクに制御を渡す為に PeekMessage を
実行する。
                while( PeekMessage( &Message, NULL, 0, 0, PM_REMOVE ) ){
                        if ( Message.message == WM_QUIT ){
                                return( Message.message );
                        }
                        TranslateMessage( &Message );
                        DispatchMessage( &Message );
                }
```

```
                NowAddressDisplay( Result );
                AC05_BRStatus1( hDev1, &Status1_x, Result );
                //AC05_BRStatus1( hDev2, &Status1_y, Result );
                //AC05_BRStatus1( hDev3, &Status1_z, Result );

        }while( Status1_x &0x01 );//& Status1_y & Status1_z & 0x01 );

        // 現在位置 ADDRESS 表示
        NowAddressDisplay( Result );

        /*** DRIVE 停止要因のセット ***/
        if( Status1 & 0x20 )
                *StopCode = 3;                          // LSEND
        else if( Status1 & 0x40 )
                *StopCode = 2;                          // SSEND
        else if( Status1 & 0x80 )
                *StopCode = 1;                          // FSEND
        else
                *StopCode = 0;                          // PASS
        return TRUE;
}

UINT CAutoGuiderDlg::ReadyWait2( AC05_S_RESULT *Result, BYTE *StopCode )
軸の Message
{
        do{
                // ループ中、一旦他のタスクに制御を渡す為に PeekMessage を
実行する。
                while( PeekMessage( &Message, NULL, 0, 0, PM_REMOVE ) ){
                        if ( Message.message == WM_QUIT ){
                                return( Message.message );
                        }
                        TranslateMessage( &Message );
                        DispatchMessage( &Message );
                }
                // 現在位置 ADDRESS 表示
                NowAddressDisplay( Result );
                AC05_BRStatus1( hDev2, &Status1_y, Result );
                }while( Status1_y & 0x01 );

        // 現在位置 ADDRESS 表示
        NowAddressDisplay( Result );

        /*** DRIVE 停止要因のセット ***/
        if( Status1 & 0x20 )
                *StopCode = 3;                          // LSEND
        else if( Status1 & 0x40 )
                *StopCode = 2;                          // SSEND
        else if( Status1 & 0x80 )
                *StopCode = 1;                          // FSEND
        else
                *StopCode = 0;                          // PASS
        return TRUE;
}

UINT CAutoGuiderDlg::ReadyWait3( AC05_S_RESULT *Result, BYTE *StopCode )
θ軸の Message
{
        do{
                // ループ中、一旦他のタスクに制御を渡す為に PeekMessage を
実行する。
                while( PeekMessage( &Message, NULL, 0, 0, PM_REMOVE ) ){
                        if ( Message.message == WM_QUIT ){
                                return( Message.message );
                        }
                        TranslateMessage( &Message );
                        DispatchMessage( &Message );
                }
                // 現在位置 ADDRESS 表示
                NowAddressDisplay( Result );
                AC05_BRStatus1( hDev3, &Status1_z, Result );
                }while( Status1_z & 0x01 );

        // 現在位置 ADDRESS 表示
        NowAddressDisplay( Result );

        /*** DRIVE 停止要因のセット ***/
        if( Status1 & 0x20 )
                *StopCode = 3;                          // LSEND
        else if( Status1 & 0x40 )
                *StopCode = 2;                          // SSEND
        else if( Status1 & 0x80 )
                *StopCode = 1;                          // FSEND
        else
                *StopCode = 0;                          // PASS
        return TRUE;
}

void CAutoGuiderDlg::StopCodeDisplay( BYTE StopCode )
{
        char c1[30] = "FSSTOP が入力されました";
        CString s1((const char*)c1, 30);
        char c2[30] = "SLSTOP が入力されました";
        CString s2((const char*)c2, 30);
        char c3[30] = "LIMIT が入力されました";
        CString s3((const char*)c3, 30);
        char c4[30] = "DRIVE が終了しました";
        CString s4((const char*)c4, 30);

        switch( StopCode ){
                case 1:
                        m_label_message.SetWindowText(s1);
                        break;
                case 2:
                        m_label_message.SetWindowText(s2);
                        break;

                case 3:
                        m_label_message.SetWindowText(s3);
                        break;

                default:
                        m_label_message.SetWindowText(s4);
        }
}

BOOL CAutoGuiderDlg::NowAddressDisplay( AC05_S_RESULT *Result )
{
        AC05_IRDrive( hDev1, &Ac05Data_x, Result );
        sprintf_s( Buffer_x, "%ld", AC05_GetData( &Ac05Data_x ) );

        CString Bux ((const char*)Buffer_x, 20);
        //SetDlgItemText( IDC_LABEL_NOW_ADDRESS, Buffer );
        m_adress_xax.SetWindowText(Bux);

        AC05_IRDrive( hDev2, &Ac05Data_y, Result );
      //Ysprintf_s( Buffer_y, "%ld", AC05_GetData( &Ac05Data_y ) );

        CString Buy ((const char*)Buffer_y, 20);
        m_adress_yax.SetWindowText(Buy);

        AC05_IRDrive( hDev3, &Ac05Data_z, Result );
        sprintf_s( Buffer_z, "%ld", AC05_GetData( &Ac05Data_z ) );

        CString Buz ((const char*)Buffer_z, 20);
        m_adress_zax.SetWindowText(Buz);
        return TRUE;
}

void CAutoGuiderDlg::ButtonDisable( void )
//指令ボタンを無効にする
{
        m_open.EnableWindow( FALSE );
        m_close.EnableWindow( FALSE );
        m_end.EnableWindow(FALSE);
        m_stop.EnableWindow( TRUE );
        m_origindrive.EnableWindow(FALSE);
        m_sdxpl.EnableWindow(FALSE);
        m_sdxne.EnableWindow(FALSE);
        m_sdypl.EnableWindow(FALSE);
        m_sdyne.EnableWindow(FALSE);
        m_sdtpl.EnableWindow(FALSE);
        m_sdtne.EnableWindow(FALSE);
        m_indrive.EnableWindow(FALSE);
        m_edit_x.EnableWindow(FALSE);
        m_edit_y.EnableWindow(FALSE);
        m_edit_theta.EnableWindow(FALSE);
}

void CAutoGuiderDlg::ButtonEnable( void )
//指令ボタンを有効にする
{
        m_open.EnableWindow( FALSE );
        m_close.EnableWindow( TRUE );
        m_end.EnableWindow(FALSE);
        m_stop.EnableWindow( FALSE );
        m_origindrive.EnableWindow(TRUE);
        m_sdxpl.EnableWindow(TRUE);
        m_sdxne.EnableWindow(TRUE);
        m_sdypl.EnableWindow(TRUE);
        m_sdyne.EnableWindow(TRUE);
        m_sdtpl.EnableWindow(TRUE);
        m_sdtne.EnableWindow(TRUE);
        m_indrive.EnableWindow(TRUE);
        m_edit_x.EnableWindow(TRUE);
        m_edit_y.EnableWindow(TRUE);
        m_edit_theta.EnableWindow(TRUE);
}
void CAutoGuiderDlg::WaitLimit()
{
        flagx = 0;
        flagy = 0;

        while( flagx == 0 || flagy == 0){
                if( flagx == 0){
                        AC05_BRStatus2( hDev1, &Status2_x, &Ac05Result );
                        NowAddressDisplay( &Ac05Result );
                        if( Status2_x & 0x08 ){
                                flagx = 1;
                        }
                }
                if( flagy == 0){
                        AC05_BRStatus2( hDev2, &Status2_y, &Ac05Result );
                        NowAddressDisplay( &Ac05Result );
                        if( Status2_y & 0x08 ){
                                flagy =1;
                        }
                }
        }
}

void CAutoGuiderDlg::Jushin(unsigned int pix[][NY], double &x_grav,
                                        double &y_grav, double &SkyLev)
{
        int i, j, x, y;
        double sigma_x = 0.0, sigma_y = 0.0, count = 0.0;
        double Sky = 0.0;
        int n_Sky = 0;
```

```
        x = NX;
        y = NY;

        for( j = 0; j <= 10; j++)
        {
                for( i = 0; i <= 10; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = 0; j <= 10; j++)
        {
                for( i = y-11; i <= y-1; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = x-11; j <= x-1; j++)
        {
                for( i = 0; i <= 5; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = x-11; j <= x-1; j++)
        {
                for( i = y-11; i <= y-1; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }

        SkyLev = Sky / n_Sky;

        for( j = 0; j < y; j++)
        {
                for( i = 0; i < x; i++)
                {
                        sigma_x += i * (pix[i][j] - SkyLev);
                        sigma_y += j * (pix[i][j] - SkyLev);
                        count += ( pix[i][j] - SkyLev );
                }
        }

        x_grav = sigma_x / count;
        y_grav = sigma_y / count;

        if( x_grav < 0.0 )  x_grav = x / 2.0;
        if( x_grav > x ) x_grav = x / 2.0;
        if( y_grav < 0.0 )  y_grav = y / 2.0;
        if( y_grav > y ) y_grav = y / 2.0;
}

void CAutoGuiderDlg::Jushin_smallpix(unsigned int pix[][ny],
 double &x_grav, double &y_grav, double &SkyLev, double &StarCount)
{
        int i, j, x, y;
        double sigma_x = 0.0, sigma_y = 0.0, count = 0.0;
        double Sky = 0.0;
        int n_Sky = 0;

        x = nx;
        y = ny;

        for( j = 0; j <= 5; j++)
        {
                for( i = 0; i <= 5; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = 0; j <= 5; j++)
        {
                for( i = 45; i <= 49; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = 45; j <= 49; j++)
        {
                for( i = 0; i <= 5; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }
        for( j = 45; j <= 49; j++)
        {
                for( i = 45; i <= 49; i++)
                {
                        Sky += pix[i][j];
                        n_Sky++;
                }
        }

        SkyLev = Sky / n_Sky;

        for( j = 0; j < y; j++)
        {
                for( i = 0; i < x; i++)
                {
                        sigma_x += i * (pix[i][j] - SkyLev);
                        sigma_y += j * (pix[i][j] - SkyLev);
                        count += ( pix[i][j] - SkyLev );
                }
        }

        x_grav = sigma_x / count;
        y_grav = sigma_y / count;

        StarCount = count;

        if( x_grav < 0.0 )  x_grav = 24.5;
        if( x_grav > 49.0 ) x_grav = 24.5;
        if( y_grav < 0.0 )  y_grav = 24.5;
        if( y_grav > 49.0 ) y_grav = 24.5;
}

double CAutoGuiderDlg::Average( unsigned int pix[][NY])
{
        int i, j, x, y;
        double Sum_Av = 0.0, Average2;
        x = NX;
        y = NY;

        for(j = 0; j < y; j++)
        {
                for(i = 0; i < x; i++)
                {
                        Sum_Av += pix[i][j];
                }
        }

        Average2 = Sum_Av / (x*y);

        return Average2;
}

double CAutoGuiderDlg::Ave_smallpix( unsigned int pix[][ny])
{
        int i, j, x, y;
        double Sum_Av = 0.0, Average2;
        x = nx;
        y = ny;

        for(j = 0; j < y; j++)
        {
                for(i = 0; i < x; i++)
                {
                        Sum_Av += pix[i][j];
                }
        }

        Average2 = Sum_Av / (x*y);

        return Average2;
}

double CAutoGuiderDlg::SDevition( unsigned int pix[][NY], double Aver)
{
        int i, j, x, y;
        double Sum_Dev = 0.0, Devition;
        x = NX;
        y = NY;

        for(j = 0; j < y; j++)
        {
                for(i = 0; i < x; i++)
                {
                        Sum_Dev += pow((pix[i][j] - Aver), 2.0);
                }
        }

        Devition = sqrt(Sum_Dev / (x * y));

        return Devition;
}

double CAutoGuiderDlg::SD_smallpix( unsigned int pix[][ny], double Aver)
{
        int i, j, x, y;
        double Sum_Dev = 0.0, Devition;
        x = nx;
        y = ny;

        for(j = 0; j < y; j++)
        {
                for(i = 0; i < x; i++)
                {
                        Sum_Dev += pow((pix[i][j] - Aver), 2.0);
                }
        }

        Devition = sqrt(Sum_Dev / (x * y));

        return Devition;
}

double CAutoGuiderDlg::Count(unsigned int pix[][NY], double SkyLev)
{
        int i, j;
        double CSum = 0;
        for(j = 0; j < NY; j++)
        {
```

```
                {
                        CSum += (pix[i][j] - SkyLev);
                }
        }

        return CSum;
}

double CAutoGuiderDlg::Count_smallpix(unsigned int pix[][ny],
 double SkyLev)
{
        int i, j;
        double CSum = 0;
        double SumSkyLev = 50 * 50 * SkyLev;
        for(j = 0; j < ny; j++)
        {
                for(i = 0; i < nx; i++)
                {
                        CSum += pix[i][j];
                }
        }

        CSum -= SumSkyLev;

        return CSum;
}

void CAutoGuiderDlg::Maximam(unsigned int pix[][NY],
int &M_x, int &M_y, int &M_count)
{
        int i, j;
        int Max = -100;


        for ( j = 0; j < NY; j++)
        {
                for ( i = 0; i < NX; i++)
                {
                        if(Max <= pix[i][j])
                        {
                                Max = pix[i][j];
                                M_x = i;
                                M_y = j;
                        }

                }
        }
        M_count = Max;
}

void CAutoGuiderDlg::Max_smallpix(unsigned int pix[][ny],
 int &M_x, int &M_y)
{
        int i, j;
        double Sum1 = 0.0, Sum2 = 0.0;
        int Max = 5;
        int Min = 1000;

        for ( j = 0; j < ny; j++)
        {
                for ( i = 0; i < nx; i++)
                {
                        if(Max <= pix[i][j])
                        {
                                Max = pix[i][j];
                                M_x = i;
                                M_y = j;
                        }
                        else
                        {
                                Max = Max;
                                M_x = M_x;
                                M_y = M_y;
                        }

                        if(Min >= pix[i][j])
                                Min = pix[i][j];
                        else
                                Min = Min;
                }
        }
}

void CAutoGuiderDlg::ffunc1(double distri[], double grav, int n,
double sig, double A, double &al_AA, double &al_ss, double &al_As,
double be_A, double be_s)
{
        double sigma;
        al_AA = 0.0;
        al_ss = 0.0;
        al_As = 0.0;
        be_A = 0.0;
        be_s = 0.0;
        for(int i = 0; i < n; i++)
        {
                sigma = sqrt(distri[i] / 100);
                al_AA += 1. / pow(sigma, 2.0) * exp(-(pow((i - grav),
                 2.0) / ( sig * sig ) ) );
                al_ss += pow(A, 2.0) / pow(sigma, 2.0) * pow((i -
                grav), 4.0) / pow(sig, 6.0) * exp(-pow((i - grav),
                2.0) / ( sig * sig ));
                al_As += A / pow(sigma, 2.0) * pow((i - grav), 2.0) /
                pow(sig, 3.0) * exp(-(pow((i - grav), 2.0) / ( sig *
                sig )));
                be_A += pow((distri[i] - A * exp(-pow((i - grav),
                (2 * sig * sig ))) * A * exp(-pow((i - grav), 2.0) /
                (2 * sig * sig));
                be_s += pow((i - grav), 2.0) /(pow(sigma, 2.0) *
                pow(sig, 3.0)) * (distri[i] - A * exp(-pow((i - grav),
                2.0) / ( 2. * sig * sig ))) * A * exp(-pow((i - grav),
                2.0) / (2 * sig * sig));
        }
}

void CAutoGuiderDlg::ffunc2(double ram, double al_AA, double al_ss,
 double al_As, double be_A, double be_s, double &del_A, double &del_s)
{
        double determ = pow((1 + ram), 2.0) * al_AA * al_ss - pow(al_As, 2.0);
        del_A = ((1 + ram) * al_ss * be_s - al_As * be_A) / determ;
        del_s = ((1 + ram) * al_AA * be_A - al_As * be_s) / determ;
}

void CAutoGuiderDlg::Gaufits(unsigned int pix[][NY], double A0, double
 sig0, double ram, double grav_x, double grav_y, double SkyLev, double &fwhm)
{
        double distri_x[NX] = {0.}, distri_y[NY] = {0.};
        double A_x[100], A_y[100], sig_x[100], sig_y[100],
        chisq_x[100] = {0}, chisq_y[100] = {0.};
        double AA_al = 0, ss_al = 0, As_al = 0, A_be = 0, s_be = 0;
        double del_A, del_s, seido1, seido2, sigma;
        int i, j, k;

        char gauf[300];
        sprintf_s(gauf, "A=%10e sig0=%10e ram=%10e grav_x=%10e
        SkyLev=%10e", A0, sig0, ram, grav_x, SkyLev);
        AfxMessageBox(gauf);

        for(i = 0; i < NX; i++)
        {
                for(j = 0; j < NY; j++)
                {
                        distri_x[i] += (pix[i][j] -SkyLev);
                }
        }

        A_x[0] = A0 * 4.;
        sig_x[0] = sig0;

        for(i = 0; i < NX; i++)
        {
                sigma = sqrt(distri_x[i] / 100);
                chisq_x[0] += 1 / pow(sigma, 2.0) * (distri_x[i] -
        A_x[0] * exp(-pow(i - grav_x, 2.0) / (2 * pow(sig_x[0], 2.0))));
        }

        sprintf_s(gauf, "A=%10e A_x[0]=%10e %10e %10e %10e", A0,
        A_x[0], sig_x[0], sigma, chisq_x[0]);
        CString gg ((const char*)gauf, 300);
        AfxMessageBox(gg);

        for(k = 1;k < 100; k++)
        {
                ffunc1(distri_x, grav_x, NX, A_x[k-1], sig_x[k-1],
                AA_al, ss_al, As_al, A_be, s_be);
                ffunc2(ram, AA_al, ss_al, As_al, A_be, s_be, del_A, del_s);

                for(i = 0; i < NX; i++)
                {
                        sigma = sqrt(distri_x[i] / 100);
                        chisq_x[k] += 1 / pow(sigma, 2.0) *
                (distri_x[i] - A_x[k] * exp(-pow(i - grav_x, 2.0) / (2
                * pow(sig_x[k], 2.0))));
                }

                if(chisq_x[k] >= chisq_x[k-1])
                        ram *=10.;
                else
                {
                        A_x[k] = A_x[k-1] + del_A;
                        sig_x[k] = sig_x[k-1] + del_s;

                        ram /= 10.;
                        seido1 = fabs( del_A / A_x[k] );
                        seido2 = fabs( del_s / sig_x[k] );
                        if(seido1 <= 0.01 && seido2 <= 0.01)
                                break;
                }
        }

        fwhm = 2 * sig_x[k] * sqrt(2 * log(2.0));

}

void CAutoGuiderDlg::Gaufits_smallpix(unsigned int pix[][50], double
sig0, double ram, double grav_x, double grav_y, double SkyLev, double
&fwhm_x, double &fwhm_y)
{
        double distri_x[NX] = {0.}, distri_y[NY] = {0.};
        double A_x[100], A_y[100], sig_x[100], sig_y[100],
               chisq_x[100] = {0}, chisq_y[100] = {0.};
        double AA_al1 = 0, ss_al1 = 0, As_al1 = 0, A_be1 = 0, s_be1 = 0;
        double AA_al2 = 0, ss_al2 = 0, As_al2 = 0, A_be2 = 0, s_be2 = 0;
        double del_A1, del_s1, del_A2, del_s2, seido1, seido2, sigma;
        int i, j, k;
        double Max_x = 10;
        double Max_y = 10;

        for(i = 0; i < 50; i++)
        {
```

```
                {
                        distri_x[i] += (pix[i][j] -SkyLev);
                }
        }

        for(j = 0; j < 50; j++)
        {
                for(i = 0; i < 50; i++)
                {
                        distri_y[j] += (pix[i][j] - SkyLev);
                }
        }

        for ( i = 0; i < 50; i++)
        {
                if(Max_x <= distri_x[i])
                        Max_x = distri_x[i];

                if(Max_y <= distri_y[i])
                        Max_y = distri_y[i];
        }

        A_x[0] = Max_x;
        A_y[0] = Max_y;
        sig_x[0] = sig0;
        sig_y[0] = sig0;

        for(i = 0; i < 50; i++)
        {
                sigma = sqrt(distri_x[i] / 100);
                chisq_x[0] += 1 / pow(sigma, 2.0) * (distri_x[i] -
                A_x[0] * exp(-pow(i - grav_x, 2.0) / (2 *
                pow(sig_x[0], 2.0))));

                sigma = sqrt(distri_y[i] / 100);
                chisq_y[0] += 1 / pow(sigma, 2.0) * (distri_y[i] -
                A_y[0] * exp(-pow(i - grav_y, 2.0) / (2 *
                pow(sig_y[0], 2.0))));
        }

        for(k = 1;k < 100; k++)
        {
                ffunc1(distri_x, grav_x, 50, A_x[k-1], sig_x[k-1],
                AA_al1, ss_al1, As_al1, A_be1, s_be1);
                ffunc2(ram, AA_al1, ss_al1, As_al1, A_be1, s_be1,
                del_A1, del_s1);

                for(i = 0; i < 50; i++)
                {
                        sigma = sqrt(distri_x[i] / 100);
                        chisq_x[k] += 1 / pow(sigma, 2.0) *
                        (distri_x[i] - A_x[k] * exp(-pow(i - grav_x,
                        2.0) / (2 * pow(sig_x[k], 2.0))));
                }

                if(chisq_x[k] >= chisq_x[k-1])
                        ram *=10.;
                else
                {
                        A_x[k] = A_x[k-1] + del_A1;
                        sig_x[k] = sig_x[k-1] + del_s1;

                        ram /= 10.;
                        seido1 = fabs( del_A1 / A_x[k] );
                        seido2 = fabs( del_s1 / sig_x[k] );
                        if(seido1 <= 0.01 && seido2 <= 0.01)
                                break;
                }
        }

        fwhm_x = 2 * sig_x[k] * 4 * 0.072 * sqrt(2 * log(2.0));

        for(k = 1;k < 100; k++)
        {
                ffunc1(distri_y, grav_y, 50, A_y[k-1], sig_y[k-1],
                AA_al2, ss_al2, As_al2, A_be2, s_be2);
                ffunc2(ram, AA_al2, ss_al2, As_al2, A_be2, s_be2,
                del_A2, del_s2);

                for(i = 0; i < 50; i++)
                {
                        sigma = sqrt(distri_y[i] / 100);
                        chisq_x[k] += 1 / pow(sigma, 2.0) *
                        (distri_y[i] - A_y[k] * exp(-pow(i - grav_y,
                        2.0) / (2 * pow(sig_y[k], 2.0))));
                }

                if(chisq_y[k] >= chisq_y[k-1])
                        ram *=10.;
                else
                {
                        A_y[k] = A_y[k-1] + del_A2;
                        sig_y[k] = sig_y[k-1] + del_s2;

                        ram /= 10.;
                        seido1 = fabs( del_A2 / A_y[k] );
                        seido2 = fabs( del_s2 / sig_y[k] );
                        if(seido1 <= 0.01 && seido2 <= 0.01)
                                break;
                }
        }

        fwhm_y = 2 * sig_y[k] * 4 * 0.072 * sqrt(2 * log(2.0));
}
```

```
void CAutoGuiderDlg::GetRaDecInformation(double dx, double dy, double
                                &dalp, double &ddel, double pangle)
{
        double dtheta, thetaorg, SF;

        thetaorg = 15.;
        dtheta = 0.0174533 * (pangle + thetaorg);
        SF = 0.072 * 4;

        dalp = SF * (dx * cos(dtheta) + dy * sin(dtheta));
        ddel = SF * (-dx * sin(dtheta) + dy * cos(dtheta));
}

void CAutoGuiderDlg::TransStage()
{
        double obj_RA, obj_DEC, guid_RA, guid_DEC, dRA, dDEC, SF,
               dtheta,thetaorg, pangle;
        int Px, Py, time_ob_RA, min_ob_RA, time_guid_RA, min_guid_RA,
            time_ob_DEC, min_ob_DEC, time_guid_DEC, min_guid_DEC;
        CString object_RA, object_DEC, guid2_RA, guid2_DEC, pulse;
        double sec_ob_RA, sec_guid_RA, sec_ob_DEC, sec_guid_DEC,
               sign_obj_DEC, sign_guid_DEC, RA_ob, RA_gui, DEC_ob, DEC_gui;

        SF = 0.072 * 4;
        thetaorg = 15.;

        AC05_IRDrive( hDev1, &Ac05Data_x, &Ac05Result );
        int PS_x = AC05_GetData( &Ac05Data_x );
        AC05_IRDrive( hDev2, &Ac05Data_y, &Ac05Result );
        int PS_y = AC05_GetData( &Ac05Data_y );

        m_Position_Angle.GetWindowText(pulse);
        m_Obs_Coord_RA.GetWindowText(object_RA);
        m_Obs_Coord_DEC.GetWindowText(object_DEC);
        m_Guid_Coord_RA.GetWindowText(guid2_RA);
        m_Guid_Coord_DEC.GetWindowText(guid2_DEC);

        pangle = atof(pulse);
        obj_RA = atof(object_RA);
        obj_DEC = atof(object_DEC);
        guid_RA = atof(guid2_RA);
        guid_DEC = atof(guid2_DEC);

        if(obj_DEC < 0){
                obj_DEC *= -1.0;
                sign_obj_DEC = -1.0;
        }
        else
                sign_obj_DEC = 1.0;

        if(guid_DEC < 0){
                guid_DEC *= -1.0;
                sign_guid_DEC = -1.0;
        }
        else
                sign_guid_DEC = 1.0;

        time_ob_RA = obj_RA/10000;
        min_ob_RA = (obj_RA/10000 - int(obj_RA/10000)) * 100;
        sec_ob_RA = (obj_DEC/100 - int(obj_DEC/100)) * 100;
        time_guid_RA = guid_RA/10000;
        min_guid_RA = (guid_RA/10000 - int(guid_RA/10000)) * 100;
        sec_guid_RA = (guid_RA/100 - int(guid_RA/100)) * 100;
        time_ob_DEC = obj_DEC/10000;
        min_ob_DEC = (obj_DEC/10000 - int(obj_DEC/10000)) * 100;
        sec_ob_DEC = (obj_DEC/100 - int(obj_DEC/100)) * 100;
        time_guid_DEC = guid_DEC/10000;
        min_guid_DEC = (guid_DEC/10000 - int(guid_DEC/10000)) * 100;
        sec_guid_DEC = (guid_DEC/100 - int(guid_DEC/100)) * 100;

        RA_ob = (time_ob_RA * 3600. + min_ob_RA * 60 + sec_ob_RA) * 15.0;
        RA_gui = (time_guid_RA * 3600. + min_guid_RA * 60 + sec_guid_RA) * 15.0;
        DEC_ob = time_ob_DEC  * 3600. + min_ob_DEC * 60 + sec_ob_DEC;
        DEC_gui = time_guid_DEC * 3600. + min_guid_DEC * 60 + sec_guid_DEC;
        RA_ob *= cos(DEC_ob/3600 * 0.0174533);
        RA_gui *= cos(DEC_gui/3600 * 0.0174533);

        dRA = RA_gui - RA_ob;
        dDEC = sign_guid_DEC * DEC_gui - sign_obj_DEC * DEC_ob;

        dtheta = 0.0174533 * (pangle - thetaorg);
        Px = -1 / SF * 100 / 3.0 * (dRA * sin(dtheta) + dDEC * cos(dtheta));
        Py = 1 / SF * 100 / 3.0 * (dRA * cos(dtheta) - dDEC * sin(dtheta));

        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( (DWORD)-(CStage_x + Px - PS_x), &Ac05Data );
        AC05_IWDrive( hDev1, 0x14, &Ac05Data, &Ac05Result );
        ReadyWait( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( (DWORD)-(CStage_y + Py - PS_y), &Ac05Data );
        AC05_IWDrive( hDev2, 0x14, &Ac05Data, &Ac05Result );
        ReadyWait2( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );
}
////////////////////////////////////////////////////////////
//         命令文
////////////////////////////////////////////////////////////

void CAutoGuiderDlg::OnBnClickedOk()
{
        //CString aa;
        //int aa_i;
        //char buf[30];
```

```
        //aa_i = atoi(aa);
        //sprintf_s(buf, "%s %d", aa, aa_i);
        //AfxMessageBox(buf);
        //OnOK();
}

void CAutoGuiderDlg::OnBnClickedOpen()
{
        XAxiOpen();
        YAxiOpen();
        ThetaAxiOpen();

        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        Data = 0x08;
        AC05_BWDriveData1( hDev1, &Data, &Ac05Result );
        Cmd = 0x01;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        Data = 0x08;
        AC05_BWDriveData1( hDev2, &Data, &Ac05Result );
        Cmd = 0x01;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        Data = 0x08;
        AC05_BWDriveData1( hDev3, &Data, &Ac05Result );
        Cmd = 0x01;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );

        RateSet();

        //char c5[30] = "原点調整中";
        //CString Op1((const char*)c5, 30);
        ////SetDlgItemText( IDC_LABEL_MESSAGE, "原点調整中" );
        //m_label_message.SetWindowText(Op1);
        //OriginDrive();
        //ReadyWait( &Ac05Result, &StopCode );
        //ReadyWait2( &Ac05Result, &StopCode );
        //AC05_BWaitDriveCommand( hDev3, 0x00, &Ac05Result );
        //Data = 0x0B;
        //AC05_BWDriveData1( hDev3, &Data, &Ac05Result );
        //Cmd = 0x1E;
        //AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
        //ReadyWait3( &Ac05Result, &StopCode );

        //Reset();
        //ReadyWait( &Ac05Result, &StopCode );

        m_open.EnableWindow( FALSE );
        m_close.EnableWindow( TRUE );
        m_end.EnableWindow(FALSE);
        m_stop.EnableWindow( FALSE );
        m_origindrive.EnableWindow(TRUE);
        m_sdxpl.EnableWindow(TRUE);
        m_sdxne.EnableWindow(TRUE);
        m_sdypl.EnableWindow(TRUE);
        m_sdyne.EnableWindow(TRUE);
        m_sdtpl.EnableWindow(TRUE);
        m_sdtne.EnableWindow(TRUE);
        m_indrive.EnableWindow(TRUE);
        m_edit_x.EnableWindow(TRUE);
        m_edit_y.EnableWindow(TRUE);
        m_edit_theta.EnableWindow(TRUE);

        char c6[30] = "オープンされました";
        CString Op2((const char*)c6, 30);
        //SetDlgItemText( IDC_LABEL_MESSAGE, "オープンされました" );
        m_label_message.SetWindowText(Op2);
}

void CAutoGuiderDlg::OnBnClickedClose()
{
        char c7[30] = "オープンしてください";
        CString Cl((const char*)c7, 30);

        AC05_BClose( hDev1, &Ac05Result );
        m_open.EnableWindow( TRUE );
        m_close.EnableWindow( FALSE );
        m_end.EnableWindow(TRUE);
        m_stop.EnableWindow( FALSE );
        m_origindrive.EnableWindow(FALSE);
        m_sdxpl.EnableWindow(FALSE);
        m_sdxne.EnableWindow(FALSE);
        m_sdypl.EnableWindow(FALSE);
        m_sdyne.EnableWindow(FALSE);
        m_sdtpl.EnableWindow(FALSE);
        m_sdtne.EnableWindow(FALSE);
        m_indrive.EnableWindow(FALSE);
        m_edit_x.EnableWindow(FALSE);
        m_edit_y.EnableWindow(FALSE);
        m_edit_theta.EnableWindow(FALSE);

        m_label_message.SetWindowText(Cl);
}

void CAutoGuiderDlg::OnBnClickedEnd()
{
        CDialog::OnOK( );;
}

void CAutoGuiderDlg::OnBnClickedStop()
{
        Cmd = 0xFF;
        StopFlag = 1;

        Cmd = 0xFF;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
        StopFlag = 1;

        Cmd = 0xFF;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
        StopFlag = 1;

        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedOrigindrive()
{
        char c8[30] = "原点調整中";
        CString OD1((const char*)c8, 30);
        char c9[30] = "原点調整終了！";
        CString OD2((const char*)c9, 30);

        //SetDlgItemText( IDC_LABEL_MESSAGE, "原点調整中" );
        m_label_message.SetWindowText(OD1);

        ButtonDisable( );
        OriginDrive();
        ReadyWait( &Ac05Result, &StopCode );
        ReadyWait2( &Ac05Result, &StopCode );
        AC05_BWaitDriveCommand( hDev3, 0x00, &Ac05Result );
        Data = 0x0B;
        AC05_BWDriveData1( hDev3, &Data, &Ac05Result );
        Cmd = 0x1E;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );
        ReadyWait3( &Ac05Result, &StopCode );
                                // DRIVE 終了待ち

        if (AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result ) &&
                AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result ) &&
                AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result )) {
                        Reset();
                        ReadyWait( &Ac05Result, &StopCode);
                        ReadyWait2( &Ac05Result, &StopCode);
                        ReadyWait3( &Ac05Result, &StopCode);
                        //NowAddressDisplay(&Ac05Result);
                        ButtonEnable();
                        m_label_message.SetWindowText(OD2);
        }
}

void CAutoGuiderDlg::OnBnClickedSdxpl()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData2( hDev1, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev1, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev1, 0xF7, &Ac05Data, &Ac05Result );

        CwScanXaxis();
        ReadyWait( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedSdxne()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData2( hDev1, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev1, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev1, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev1, 0xF7, &Ac05Data, &Ac05Result );

        CCwScanXaxis();
        ReadyWait( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedSdypl()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData2( hDev2, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev2, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );
```

```
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev2, 0xF7, &Ac05Data, &Ac05Result );

        CwScanYaxis();
        ReadyWait2( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedSdyne()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData2( hDev2, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev2, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev2, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev2, 0xF7, &Ac05Data, &Ac05Result );

        CCwScanYaxis();
        ReadyWait2( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedSdtpl()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData2( hDev3, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev3, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev3, 0xF7, &Ac05Data, &Ac05Result );

        CwScanThetaaxis();
        ReadyWait3( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable();
}

void CAutoGuiderDlg::OnBnClickedSdtne()
{
        ButtonDisable();
        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
         Data = 0x03;
        AC05_BWDriveData2( hDev3, &Data, &Ac05Result );
        Data = 0x03;
        AC05_BWDriveData3( hDev3, &Data, &Ac05Result );
        Cmd = 0x06;
        AC05_BWDriveCommand( hDev3, &Cmd, &Ac05Result );

        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        AC05_SetData( 1000, &Ac05Data );
        AC05_IWDrive( hDev3, 0xF7, &Ac05Data, &Ac05Result );

        CCwScanThetaaxis();
        ReadyWait3( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        RateSet();
        ButtonEnable( );
}

void CAutoGuiderDlg::OnBnClickedIndrive()
{
        ButtonDisable();

        int xx_i, yy_i, zz_i;
        CString xx, yy, zz;

        m_edit_x.GetWindowText(xx);
        m_edit_y.GetWindowText(yy);
        m_edit_theta.GetWindowText(zz);
        xx_i = atoi(xx);
        yy_i = atoi(yy);
        zz_i = atoi(zz);

        AC05_BWaitDriveCommand( hDev1, 0, &Ac05Result );
        AC05_SetData( (DWORD)-xx_i, &Ac05Data );
        AC05_IWDrive( hDev1, 0x14, &Ac05Data, &Ac05Result );
        ReadyWait( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        AC05_BWaitDriveCommand( hDev2, 0, &Ac05Result );
        AC05_SetData( (DWORD)-yy_i, &Ac05Data );
        AC05_IWDrive( hDev2, 0x14, &Ac05Data, &Ac05Result );
        StopCodeDisplay( StopCode );

        AC05_BWaitDriveCommand( hDev3, 0, &Ac05Result );
        AC05_SetData( (DWORD)zz_i, &Ac05Data );
        AC05_IWDrive( hDev3, 0x14, &Ac05Data, &Ac05Result );
        ReadyWait3( &Ac05Result, &StopCode );
        StopCodeDisplay( StopCode );

        m_edit_x.SetWindowText("");
        m_edit_y.SetWindowText("");
        m_edit_theta.SetWindowText("");

        ButtonEnable();
}

void CAutoGuiderDlg::OnBnClickedCooler()
{
        m_Cooler.EnableWindow(FALSE);
        m_Shoot.EnableWindow(FALSE);

        CCDCoolerPower(80);

        BYTE data2[4];
        int temp;
        char ss[50];

        do{
                AfxMessageBox("OK");
                while( PeekMessage( &Message, NULL, 0, 0, PM_REMOVE ) ){
                        if ( Message.message == WM_QUIT ){
                                //return( Message.message );
                        }
                        TranslateMessage( &Message );
                        DispatchMessage( &Message );
                }
                WaitTimerMessage(1000);
                //CCDGetTemperature(0, temp);
                OutPort(NULL, 0xa2);
                InPort(data2, sizeof(data2));
                AfxMessageBox("OK2");
                temp = (data2[2] >>4) *100 + (data2[2] &0xf) *10 +
                (data2[3] &0xf);
                if (data2[3] & 0x80)
                        temp = -temp;
                sprintf_s(ss, "%02x %02x %02x %02x", data2[0],
                data2[1], data2[2], data2[3]);
                CString temp2 ((const char*)ss, 50);
                AfxMessageBox(ss);
                //AfxMessageBox(ss);
                m_CCD_Temp.SetWindowText(temp2);
        }while(temp > -10.0);

        m_Cooler.EnableWindow(TRUE);
        m_Shoot.EnableWindow(TRUE);
}

void CAutoGuiderDlg::OnBnClickedShoot()
{
        char expres[50];
        CSize size;
        HGLOBAL hCCD = NULL;
        int i, j;

        CString timer;
        double timer_f, Ti = 0.1;
        m_Timer_Set.GetWindowText(timer);
        timer_f = atof(timer);
        if(timer_f<0.1)
                timer_f = 1.0;
        Ti = timer_f * 10.;

        BYTE data1[] = {0x20, 0x10, 0, 0, 0};
        OutPort(data1, sizeof(data1));

        size = CCDSizeExposure(0, 3, 0);

        BYTE data3[] = {0x22, LOBYTE(Ti), HIBYTE(Ti), 0};
        OutPort(data3, sizeof(data3));

        BYTE data4[] = {0x23, 0};
        OutPort(data4, sizeof(data4));

        Sleep((DWORD)Ti*100);

        hCCD=::GlobalAlloc(GMEM_MOVEABLE | GMEM_ZEROINIT, size.cx * size.cy * 2 );
        CCDTransferImage(0, hCCD);

        CCDFinishExposure();

        LPBYTE pCCD = (LPBYTE) ::GlobalLock(hCCD);

        unsigned int pixel[NX][NY];
        unsigned int pixdata[NX][NY];
        for ( j = NY-1; j >= 0; j--)
        {
                for ( i = 0; i < NX; i++)
                {
                        ULONG data = MAKEWORD(pCCD[0], pCCD[1]);
                        pixdata[i][j] = data;
                        pCCD += 2;
                }
        }
        ::GlobalUnlock(hCCD);
```

```
	int C_high, C_low;
	CString high, low;

	m_CHigh.GetWindowText(high);
	m_CLow.GetWindowText(low);

	if(high == "")
		C_high = 65535;
	else
		C_high = atoi(high);

	if(low == "")
		C_low = 0;
	else
		C_low = atoi(low);

	for( j = 0; j <NY; j++)
	{
		for( i = 0; i < NX; i++)
		{
			pixel[i][j] = (pixdata[i][j] - C_low) * 256 / (C_high - C_low);

			if(pixel[i][j] < 0)
				pixel[i][j] = 0;

			else if(pixel[i][j] > 255)
				pixel[i][j] = 255;
		}
	}

	double Ave, hensa, GX, GY, LOS;
	int Max_x, Max_y, Max_count;
	Ave = Average(pixdata);
	hensa = SDevition(pixdata, Ave);
	Jushin(pixdata, GX, GY, LOS);
	Maximam(pixdata, Max_x, Max_y, Max_count);

	CClientDC myPictDC(&m_Static_Screen);
	CRect myRECT;
	m_Static_Screen.GetClientRect(myRECT);
	myPictDC.FillSolidRect(myRECT, RGB(255, 255, 255));

	for( j = 0; j < NY; j++)
	{
		for( i = 0; i < NX; i++)
		{
			myPictDC.SetPixel(i, j, RGB(pixel[i][j],
			pixel[i][j], pixel[i][j]));
		}
	}

	sprintf_s(expres, "%6.2f, %6.2f", GX, GY);
	CString value ((const char*)expres, 50);
	m_Center_of_Gravity.SetWindowText(value);

	sprintf_s(expres, "%6.2f", LOS);
	CString LeSky ((const char*)expres, 50);
	m_Sky_Level.SetWindowText(LeSky);

	sprintf_s(expres, "%ld %10e", Max_count, LOS);
	CString ff ((const char*)expres, 100);
	AfxMessageBox(ff);
	double FWHM_x;
	Gaufits(pixdata, Max_count - LOS, 100, 0.001, GX, GY, LOS, FWHM_x);

	char expres2[100];
	sprintf_s(expres2, "%6.2f", FWHM_x);
	CString tt ((const char*)expres2, 100);
	AfxMessageBox(tt);
}

void CAutoGuiderDlg::OnBnClickedAutoguid()
{
	char expres[50];
	CSize size;
	HGLOBAL hCCD = NULL;
	int i, j, lp;
	double gx_cre[100], gy_cre[100], delta_x[100], delta_y[100], star_count[100];
	MSG msg;

	CString timer;
	double timer_f, Ti;
	m_Timer_Set.GetWindowText(timer);
	timer_f = atof(timer);
	Ti = timer_f * 10;

	int C_high, C_low;
	CString high, low;

	m_CHigh.GetWindowText(high);
	m_CLow.GetWindowText(low);
	C_high = atoi(high);
	C_low = atoi(low);

	CBrush brushRed(RGB(255, 0, 0));
	CBrush brushBlue(RGB(0, 0, 255));

	m_Stop_AG = FALSE;
	for(lp = 0; lp < 100; lp++)
	{
		if(m_Stop_AG)
			break;
		BYTE data1[] = {0x20, 0x10, 0, 0, 0};
		OutPort(data1, sizeof(data1));

		BYTE data3[] = {0x22, LOBYTE(Ti), HIBYTE(Ti), 0};
		OutPort(data3, sizeof(data3));

		BYTE data4[] = {0x23, 0};
		OutPort(data4, sizeof(data4));

		Sleep((DWORD)Ti*100);

		hCCD=::GlobalAlloc(GMEM_MOVEABLE | GMEM_ZEROINIT,
		size.cx * size.cy * 2 );
		CCDTransferImage(0, hCCD);

		CCDFinishExposure();

		LPBYTE pCCD = (LPBYTE) ::GlobalLock(hCCD);

		unsigned int pixel[NX][NY];
		unsigned int pixdata[NX][NY];
		for ( j = NY-1; j >= 0; j--)
		{
			for ( i = 0; i < NX; i++)
			{
				ULONG data = MAKEWORD(pCCD[0], pCCD[1]);
				pixdata[i][j] = data;
				pCCD += 2;
			}
		}
		::GlobalUnlock(hCCD);

		for( j = 0; j <NY; j++)
		{
			for( i = 0; i < NX; i++)
			{
				pixel[i][j] = (pixdata[i][j] - C_low)
				* 256 / (C_high - C_low);

				if(pixel[i][j] < 0)
					pixel[i][j] = 0;

				else if(pixel[i][j] > 255)
					pixel[i][j] = 255;
			}
		}

		double Ave, hensa, GX, GY, LOS;
		int Max_x, Max_y, Max_count;
		Ave = Average(pixdata);
		hensa = SDevition(pixdata, Ave);
		Jushin(pixdata, GX, GY, LOS);
		Maximam(pixdata, Max_x, Max_y, Max_count);

		CClientDC myPictDC(&m_Static_Screen);
		CRect myRECT;
		m_Static_Screen.GetClientRect(myRECT);
		myPictDC.FillSolidRect(myRECT, RGB(255, 255, 255));

		for( j = 0; j < NY; j++)
		{
			for( i = 0; i < NX; i++)
			{
				myPictDC.SetPixel(i, j,
				RGB(pixel[i][j], pixel[i][j],
				pixel[i][j]));
			}
		}

		sprintf_s(expres, "%6.2f, %6.2f", GX, GY);
		CString exp1 ((const char*)expres, 50);
		m_Center_of_Gravity.SetWindowText(exp1);

		sprintf_s(expres, "%6.2f", LOS);
		CString exp2 ((const char*)expres, 50);
		m_Sky_Level.SetWindowText(exp2);

		CWnd *COG = GetDlgItem(IDC_CGravity_Graph);
		CDC *pDC=COG->GetDC();
		CRect graph;
		COG->GetClientRect(&graph);

		int gw,gh;
		gw=graph.right;
		gh=graph.bottom;
		int cx = GX * 97 / 340;
		int cy = GY * 85 / 256;
		CRect rect1(cx-1, cy-1, cx+1, cy+1);
		pDC->FillRect(rect1, &brushRed);

		star_count[lp] = Count(pixdata, LOS);
		CWnd *h4 = GetDlgItem(IDC_COUNT);
		CDC *pDC4=h4->GetDC();
		CRect count_sum;
		h4->GetClientRect(&count_sum);

		int contgra;
		contgra = count_sum.bottom;
		cy = contgra / 2 - (star_count[lp] * 40 / (NX * NY * 255));
		CRect rect2(lp*5+9, cy-1, lp*5+11, cy+1);
		pDC4->FillRect(rect2, &brushRed);

		Sleep(500);

		gx_cre[lp] = GX;
		gy_cre[lp] = GY;
```

```
                double sum_star_count=0, sum2_star_count=0,
                sum_delta_x=0, sum_delta_y=0, sum2_delta_x=0,
                sum2_delta_y=0;
                if(lp >= 1)
                {
                        for(i = 0; i <= lp; i++){
                                sum_star_count += star_count[i];
                        }
                        double Ave_star_count = sum_star_count / (lp + 1);
                        for(i = 0; i <= lp; i++){
                                sum2_star_count += pow(star_count[i] -
                                Ave_star_count, 2.0);
                        }
                        double SD_star_count = sqrt(sum2_star_count / 2);
                        sprintf_s(expres, "%8.1f", Ave_star_count);
                        CString exp3 ((const char*) expres, 50);
                        m_Average_Count.SetWindowText(exp3);
                        sprintf_s(expres, "%6.1f", SD_star_count);
                        CString exp4 ((const char*) expres, 50);
                        m_SD_Count.SetWindowText(exp4);

                        delta_x[0] = 0.0, delta_y[0] = 0.0;
                        delta_x[lp] = gx_cre[lp] - gx_cre[0];
                        delta_y[lp] = gy_cre[lp] - gy_cre[0];

                        CWnd *h2 = GetDlgItem(IDC_CREEP);
                        CDC *pDC2=h2->GetDC();
                        CRect creep;
                        h2->GetClientRect(&creep);

                        int ddelta_y;
                        ddelta_y = creep.bottom;
                        cx = ddelta_y / 2 - (delta_x[lp] * 40 / 340);
                        cy = ddelta_y / 2 - (delta_y[lp] * 40 / 256);
                        CRect rect3(lp*5+9, cx-1, lp*5+11, cx+1);
                        CRect rect4(lp*5+9, cy-1, lp*5+11, cy+1);
                        pDC2->FillRect(rect3, &brushRed);
                        pDC2->FillRect(rect4, &brushBlue);

                        if(lp >=2)
                        {
                                for(i = 1; i <=lp; i++){
                                        sum_delta_x += delta_x[i];
                                        sum_delta_y += delta_y[i];
                                }
                                double Ave_delta_x = sum_delta_x / lp;
                                double Ave_delta_y = sum_delta_y / lp;
                                for(i = 1; i <= lp; i++){
                                        sum2_delta_x += pow(delta_x[i]
                                        - Ave_delta_x, 2.0);
                                        sum2_delta_y += pow(delta_y[i]
                                        - Ave_delta_y, 2.0);
                                }
                                double SD_delta_x = sqrt(sum2_delta_x / lp);
                                double SD_delta_y = sqrt(sum2_delta_y / lp);
                                sprintf_s(expres, "%5.2f", Ave_delta_x);
                                CString exp5((const char*) expres, 50);
                                m_Average_Creep_X.SetWindowText(exp5);
                                sprintf_s(expres, "%5.2f", Ave_delta_y);
                                CString exp6((const char*) expres, 50);
                                m_Average_Creep_Y.SetWindowText(exp6);
                                sprintf_s(expres, "%4.2f", SD_delta_x);
                                CString exp7((const char*) expres, 50);
                                m_SD_Creep_X.SetWindowText(exp7);
                                sprintf_s(expres, "%4.2f", SD_delta_y);
                                CString exp8((const char*) expres, 50);
                                m_SD_Creep_Y.SetWindowText(exp8);
                        }
                }

                if(::PeekMessage(&msg, NULL, 0, 0, PM_NOREMOVE))
                {
                        if(!::GetMessage(&msg, NULL, 0, 0))
                                return;
                        ::TranslateMessage(&msg);
                        ::DispatchMessage(&msg);
                }
        }
}

void CAutoGuiderDlg::OnBnClickedStopAg()
{
        m_Stop_AG = TRUE;
}

void CAutoGuiderDlg::OnLButtonDblClk(UINT nFlags, CPoint point)
{
        // TODO: ここにメッセージ ハンドラ コードを追加するか、既定の処理
を呼び出します。
        CBrush brushRed(RGB(255, 0, 0));
        CBrush brushBlue(RGB(0, 0, 255));
        CBrush brushWhite(RGB(255, 255, 255));
        FILE *fp;
        char logfn[50];
        char expres[50];
        CSize size;
        HGLOBAL hCCD = NULL;
        int i, j, lp;
        double gx_cre[100], gy_cre[100], delta_x[100], delta_y[100], star_count[100];
        MSG msg;

        double StarCount;

        CString timer;
        m_Timer_Set.GetWindowText(timer);
        timer_f = atof(timer);
        if(timer_f<1.0)
                timer_f = 1.0;
        Ti = timer_f * 10;

        int C_high, C_low;
        CString high, low;

        time_t current_time;
        struct tm *local;

        time( &current_time );
        local = localtime( &current_time );

        sprintf( logfn, "ag%02d%02d%02d%02d%02d%02d.log",\
                                local->tm_year-100, local->tm_mon+1,
                                local->tm_mday, local->tm_hour,
                                local->tm_min,\
                                local->tm_sec);


        m_CHigh.GetWindowText(high);
        m_CLow.GetWindowText(low);
        if(high == "")
                C_high = 65535;
        else
                C_high = atoi(high);

        if(low == "")
                C_low = 0;
        else
                C_low = atoi(low);


        CWnd *h = GetDlgItem(IDC_STATIC_SCREEN);
        CDC *pDC=h->GetDC();
        CRect graph;
        h->GetClientRect(&graph);

        int ccx = point.x - 13;
        int ccy = point.y - 11;
        CRect rect(ccx - 25, ccy - 25, ccx + 25, ccy + 25);
        pDC->FrameRect(rect, &brushWhite);
        CRect rects1(ccx - 25, ccy - 25, ccx + 25, ccy );
        pDC->FrameRect(rects1, &brushWhite);
        CRect rects2(ccx - 25, ccy - 25, ccx, ccy + 25);
        pDC->FrameRect(rects2, &brushWhite);

        m_Stop_AG = FALSE;
        for(lp = 0; lp < 100; lp++)
        {
                if(m_Stop_AG)
                        break;

                BYTE data1[] = {0x20, 0x10, 0, 0, 0};
                OutPort(data1, sizeof(data1));

                size = CCDSizeExposure(0, 3, 0);

                BYTE data3[] = {0x22, LOBYTE(Ti), HIBYTE(Ti), 0};
                OutPort(data3, sizeof(data3));

                BYTE data4[] = {0x23, 0};
                OutPort(data4, sizeof(data4));

                Sleep((DWORD)Ti*100);

                hCCD=::GlobalAlloc(GMEM_MOVEABLE | GMEM_ZEROINIT,
                size.cx * size.cy * 2 );
                CCDTransferImage(0, hCCD);

                CCDFinishExposure();

                LPBYTE pCCD = (LPBYTE) ::GlobalLock(hCCD);

                unsigned int pixel[NX][NY], small_pixel[50][50];
                unsigned int pixdata[NX][NY];
                for ( j = NY-1; j >= 0; j--)
                {
                        for ( i = 0; i < NX; i++)
                        {
                                ULONG data = MAKEWORD(pCCD[0], pCCD[1]);
                                pixdata[i][j] = data;
                                pCCD += 2;
                        }
                }
                ::GlobalUnlock(hCCD);

                for( j = 0; j < NY; j++)
                {
                        for( i = 0; i < NX; i++)
                        {
                                pixel[i][j] = (pixdata[i][j] - C_low)
                                * 256 / (C_high - C_low);

                                if(pixel[i][j] < 0)
                                        pixel[i][j] = 0;

                                else if(pixel[i][j] > 255)
                                        pixel[i][j] = 255;
                        }
                }
```

```
for(j = 0; j < 50; j++)
{
        for(i = 0; i < 50; i++)
        {
                small_pixel[i][j] = pixdata[ccx-24+i][ccy-24+j];
        }
}

double Ave, hensa, GX, GY, LOS;
int Max_x, Max_y;
Ave = Ave_smallpix(small_pixel);
hensa = SD_smallpix(small_pixel, Ave);
Jushin_smallpix(small_pixel, GX, GY, LOS, StarCount);
Max_smallpix(small_pixel, Max_x, Max_y);
GX -= 24.5;
GY -= 24.5;

CClientDC myPictDC(&m_Static_Screen);
CRect myRECT;
m_Static_Screen.GetClientRect(myRECT);
myPictDC.FillSolidRect(myRECT, RGB(255, 255, 255));

for( j = 0; j < NY; j++)
{
        for( i = 0; i < NX; i++)
        {
                myPictDC.SetPixel(i, j,
                RGB(pixel[i][j], pixel[i][j],
                pixel[i][j]));
        }
}

CRect rects3(ccx - 25, ccy - 25, ccx + 25, ccy + 25);
pDC->FrameRect(rects3, &brushWhite);
CRect rects4(ccx - 25, ccy - 25, ccx + 25, ccy );
pDC->FrameRect(rects4, &brushWhite);
CRect rects5(ccx - 25, ccy - 25, ccx, ccy + 25);
pDC->FrameRect(rects5, &brushWhite);

int gxi = int( GX );
int gyi = int( GY );

CRect rects6( gxi - 2 + ccx, gyi - 2 + ccy, gxi + 2 +
              ccx, gyi + 2 + ccy);
pDC->FrameRect(rects6, &brushRed);


sprintf_s(expres, "%+6.2f, %+6.2f", GX, GY);
CString exp1 ((const char*)expres, 50);
m_Center_of_Gravity.SetWindowText(exp1);

sprintf_s(expres, "%6.2f", LOS);
CString exp2 ((const char*)expres, 50);
m_Sky_Level.SetWindowText(exp2);

CWnd *COG = GetDlgItem(IDC_CGravity_Graph);
CDC *pDC=COG->GetDC();
CRect graph;
COG->GetClientRect(&graph);

int gw,gh;
gw=graph.right;
gh=graph.bottom;
int cx = (GX + 24.5) / 50. * gw;
int cy = (GY + 24.5) / 50. * gh;

CRect rect1(cx-1, cy-1, cx+1, cy+1);
pDC->FillRect(rect1, &brushRed);

star_count[lp] = StarCount;
CWnd *h4 = GetDlgItem(IDC_COUNT);
CDC *pDC4=h4->GetDC();
CRect count_sum;
h4->GetClientRect(&count_sum);

int contgra;
contgra = count_sum.bottom - 5;
cy = contgra - ((2/3 * contgra) / star_count[0]) * star_count[lp];
if(cy >= contgra)
        cy = contgra;
else if (cy <= 5)
        cy = 5;
CRect rect2(lp*2+9, cy-1, lp*2+11, cy+1);
pDC4->FillRect(rect2, &brushRed);

//CWnd *h3 = GetDlgItem(IDC_FWHM);
//CDC *pDC3 = h3=>GetDC();
//CRect FWidth;
//h3->GetClientRect(&FWidth);

//int widgra;
//widgra = FWidth.bottom - 5;
//cx = widgra - FWHM_x;
//cy = widgra - FWHM_y;

while( TGetValueInt( 138 ) == 0 )
        Sleep(500);
Sleep(3000);

gx_cre[lp] = GX;
gy_cre[lp] = GY;

double sum_star_count, sum2_star_count, sum_delta_x,
       sum_delta_y, sum2_delta_x, sum2_delta_y;
if(lp >= 1)
{
sum_star_count = 0;
sum2_star_count = 0;
sum_delta_x = 0;
sum_delta_y = 0;
sum2_delta_x = 0;
sum2_delta_y = 0;

for(i = 0; i <= lp; i++){
        sum_star_count += star_count[i];
}
double Ave_star_count = sum_star_count / (lp + 1);

for(i = 0; i <= lp; i++){
        sum2_star_count += pow((star_count[i]
        - Ave_star_count), 2.0);
}
double SD_star_count = sqrt(sum2_star_count /
                       (lp + 1));

sprintf_s(expres, "%8.1f", Ave_star_count);
CString exp3 ((const char*) expres, 50);
m_Average_Count.SetWindowText(exp3);
sprintf_s(expres, "%6.1f", SD_star_count);
CString exp4 ((const char*) expres, 50);
m_SD_Count.SetWindowText(exp4);

delta_x[0] = 0.0, delta_y[0] = 0.0;
delta_x[lp] = gx_cre[lp] - gx_cre[0];
delta_y[lp] = gy_cre[lp] - gy_cre[0];

CWnd *h2 = GetDlgItem(IDC_CREEP);
CDC *pDC2=h2->GetDC();
CRect creep;
h2->GetClientRect(&creep);

int ddelta_y;
ddelta_y = creep.bottom;
cx = ddelta_y / 2. - (delta_x[lp] * ddelta_y /
     2. / 25);
cy = ddelta_y / 2. - (delta_y[lp] * ddelta_y /
     2. / 25);
//sprintf_s(expres, "%5.2f %5.2f %d %d",
        delta_x[lp], delta_y[lp], cx, cy);
//CString qq ((const char*)expres, 50);
//AfxMessageBox(qq);
CRect rect3(2*lp+9, cx-1, 2*lp+11, cx+1);
CRect rect4(2*lp+9, cy-1, 2*lp+11, cy+1);
pDC2->FillRect(rect3, &brushRed);
pDC2->FillRect(rect4, &brushBlue);

CString PA;
double PA_f, dalp, ddel, dx, dy;
m_Position_Angle.GetWindowText(PA);
if(PA == "")
        PA_f = 0.0;
else
        PA_f = atof(PA);

dx = delta_x[lp];
dy = delta_y[lp];
CTransformation(dx, dy, dalp, ddel, PA_f);

//sprintf_s(expres, "%5.2f %5.2f", dalp, ddel);
//CString ww ((const char*)expres, 50);
//AfxMessageBox(ww);

if(TELFLAG == 1){
        double movefact = 0.1;
        SendTelescopeOffset( dalp * movefact,
        ddel * movefact );
}


if(lp >=2)
{
        for(i = 1; i <=lp; i++){
                sum_delta_x += delta_x[i];
                sum_delta_y += delta_y[i];
        }
        double Ave_delta_x = sum_delta_x / lp;
        double Ave_delta_y = sum_delta_y / lp;
        for(i = 1; i <= lp; i++){
                sum2_delta_x += pow(delta_x[i]
                - Ave_delta_x, 2.0);
                sum2_delta_y += pow(delta_y[i]
                - Ave_delta_y, 2.0);
        }
        double SD_delta_x = sqrt(sum2_delta_x / lp);
        double SD_delta_y = sqrt(sum2_delta_y / lp);
        sprintf_s(expres, "%5.2f", Ave_delta_x);
        CString exp5((const char*) expres, 50);
        m_Average_Creep_X.SetWindowText(exp5);
        sprintf_s(expres, "%5.2f", Ave_delta_y);
        CString exp6((const char*) expres, 50);
        m_Average_Creep_Y.SetWindowText(exp6);
        sprintf_s(expres, "%4.2f", SD_delta_x);
        CString exp7((const char*) expres, 50);
        m_SD_Creep_X.SetWindowText(exp7);
        sprintf_s(expres, "%4.2f", SD_delta_y);
        CString exp8((const char*) expres, 50);
        m_SD_Creep_Y.SetWindowText(exp8);
}

/*** log file ***/
```

```
                        local = localtime( &current_time );
                        if( ( fp = fopen( logfn, "at" ) ) == NULL ){
                                AfxMessageBox( "Cannot open log file" );
                                return;
                        }
                        fprintf( fp, "%03d %04d%02d%02d %02d:%02d:%02d
                                %+6.2f %+6.2f %+6.2f %+6.2f %6.1f %8.1f\n",
                                lp, local->tm_year+1900,
                                local->tm_mon+1, local->tm_mday,
                                local->tm_hour, local->tm_min,\
                                local->tm_sec, dx, dy, dalp, ddel, LOS, StarCount );
                        fclose( fp );

                }

                if(::PeekMessage(&msg, NULL, 0, 0, PM_NOREMOVE))
                {
                        if(!::GetMessage(&msg, NULL, 0, 0))
                                return;
                        ::TranslateMessage(&msg);
                        ::DispatchMessage(&msg);
                }
        }

        CDialog::OnLButtonDblClk(nFlags, point);
}

//
// Initializing Socket communication
//
int CAutoGuiderDlg::SockInit( void ){
        WSADATA wsa;
        int ret;
        if( (ret=WSAStartup(MAKEWORD(1,1), &wsa))){
                char buf[80];
                wsprintf(buf, "%d is the err", ret );
                AfxMessageBox( buf );
                exit( -1 );
        }
        return FALSE;
}

//
// Socket Connection (for Client)
//
int CAutoGuiderDlg::SockConnect( void ){
        SOCKADDR_IN cl_sin;

        sock = socket( AF_INET, SOCK_STREAM, IPPROTO_TCP );
        if( sock==INVALID_SOCKET){
                AfxMessageBox( "Socket() failed" );
                return TRUE;
        }
        memset( &cl_sin, 0x00, sizeof( cl_sin ) );
        cl_sin.sin_family = AF_INET;
        cl_sin.sin_port = htons( PORT );
    cl_sin.sin_addr.s_addr = inet_addr( HOST_NAME );

        if( connect( sock, (LPSOCKADDR)&cl_sin, sizeof(cl_sin))==SOCKET_ERROR ){
                if( WSAGetLastError()!=WSAEWOULDBLOCK){
                        closesocket( sock );
                        sock=INVALID_SOCKET;
                        AfxMessageBox( "connect() failed" );
                        return TRUE;
                }
        }
        return FALSE;
}
int CAutoGuiderDlg::TGetOffset( TELEPARAM *offset ){
        char wbuf[1024], rbuf[1024];
        memset( wbuf, 0, sizeof( wbuf ) );
        memset( rbuf, 0, sizeof( wbuf ) );

        sprintf( wbuf, "A %03d %03d %03d %03d %03d %03d\n",\
                50, 51, 52, 53, 54, 306 );
        if( send( sock, wbuf, strlen(wbuf), 0 )==SOCKET_ERROR ){
                AfxMessageBox( "Sending Socket connection failed." );
        }
        Sleep( 100 );
        if( recv( sock, rbuf, 1024, 0 ) <= 0 ){
                AfxMessageBox( "Receiving null data in Socket Connection." );
        }
        //AfxMessageBox( rbuf );
        sscanf( rbuf, "A %lf %lf %lf %lf %lf %lf", &offset->ra, &offset->dec,\
                &offset->azi, &offset->al, &offset->ir1, &offset->ir2 );

        return FALSE;
}
int CAutoGuiderDlg::TOffset( const TELEPARAM *offset ){
        char wbuf[1024], rbuf[1024];
        memset( wbuf, 0, sizeof( wbuf ) );
        memset( rbuf, 0, sizeof( wbuf ) );

        sprintf( wbuf, "P %.1lf %.1lf %.1lf %.1lf %.1lf %.1lf\n",\
                offset->ra, offset->dec, offset->ir1, offset->ir2,\
                offset->azi, offset->al );
//      AfxMessageBox( wbuf );
        if( send( sock, wbuf, strlen(wbuf), 0 )==SOCKET_ERROR ){
                AfxMessageBox( "Sending Socket connection failed." );
        }
        Sleep( 100 );
        if( recv( sock, rbuf, 1024, 0 ) <= 0 ){
                AfxMessageBox( "Receiving null data in Socket Connection." );
        }
                return FALSE;
        }
        //
        // GetTelescopeValue
        //
int CAutoGuiderDlg::TGetValueInt( int command ){
        char wbuf[1024], rbuf[1024];
        memset( wbuf, 0, sizeof( wbuf ) );
        memset( rbuf, 0, sizeof( wbuf ) );
        SockInit();
        SockConnect();

        sprintf( wbuf, "A %03d\n", command );
        if( send( sock, wbuf, strlen(wbuf), 0 )==SOCKET_ERROR ){
                AfxMessageBox( "Sending Socket connection failed." );
        }
        Sleep( 100 );
        if( recv( sock, rbuf, 1024, 0 ) <= 0 ){
                AfxMessageBox( "Receiving null data in Socket Connection." );
        }
        closesocket( sock );
        sock=INVALID_SOCKET;

        return atoi( &rbuf[2] );

}

//
// Move Telescope (sending offset)
//
int CAutoGuiderDlg::SendTelescopeOffset( double dalp, double ddel ){
        TELEPARAM *offset;
        offset = (TELEPARAM *)malloc( sizeof( TELEPARAM ) );
        memset( (TELEPARAM *)offset, 0.0, sizeof( TELEPARAM) );

        SockInit();
        SockConnect();

        TGetOffset( offset );
        offset->ra = offset->ra - dalp;
    offset->dec = offset->dec + ddel;
        offset->ir1 = offset->ir1 * 3600;
        offset->ir2 = offset->ir2 * 3600;
        TOffset( offset );

        free( offset );

        closesocket( sock );
        sock=INVALID_SOCKET;

        return FALSE;
}

void CAutoGuiderDlg::OnBnClickedDraveStage()
{
        TransStage();
```