広視野可視光望遠鏡 WIDGET に使用する CCD カメラ接続プレートの設計および衛星自動追尾プログラムの改良

埼玉大学 理学部 物理学科 田代研究室 学籍番号 03PR029
増野 圭輔

指導教員：田代 信 助教授
浦田 裕次 日本学術振興会特別研究員

2 月 15 日 (木)

概 要

広視野可視光望遠鏡 WIDGET は、ガンマ線バーストの可視光閃光を検出することを目的として理研と埼玉大学が中心となって開発した地上望遠鏡である。ガンマ線バーストとは宇宙最大規模の突発的な爆発現象で、衛星軌道上のガンマ線バースト観測衛星による常時観測によって検出される。ガンマ線バーストの発生直前直後における可視光閃光の観測はその突発性などから非常に困難であり、現在までに成功した例がない。この状況を打開するため、WIDGET は複数の CCD カメラと広角レンズを用いた広い観測視野と、専用 PC での自律制御によるガンマ線バースト観測衛星 Swift の同時追尾機能による観測を行っている。広い観測視野をもつことによって、いつどこで起きるか分からないガンマ線バーストをとらえる可能性が高くなる。また、Swift の視野を同時観測することにより、ガンマ線バーストが検出された前後における可視広閃光の有無を観測することが可能になる。本研究では、観測に使用する CCD カメラを Canon EF35mm 3 台から Canon EF50mm 4 台へと増設することに伴って、新たなカメラ接続プレートを設計、製作した。これにより、WIDGET の観測視野を $64° \times 64°$、分解能 0.94 arcmin とすることができた。また、望遠鏡の Swift 自動追尾プログラムを改良し、現在 Swift が観測している方向だけでなく次の観測予定方向も読み込むようにした。この改良によって観測方向の最適化を行い、Swift がバーストをとらえた場合の早期可視光閃光の検出を目指す。

目 次

図 目 次

表 目 次

第1章　ガンマ線バースト

1.1　ガンマ線バースト とは

ガンマ線バーストとは、宇宙のどこかから前触れもなくやってくる宇宙最大規模の爆発現象である。ガンマ線バーストは 1967 年に初めて観測され、現在までにその原因を突き止めようとさまざまな方法によって観測されてきた。しかし発見から 40 年経った現在もなお、その発生機構などについて謎の部分が多く、より詳細な観測をするためのアプローチがなされている。

等方性　1991 年に打ち上げられた NASA の Compton ガンマ線観測衛星は Burst And Transient Source Experiment (BATSE) という検出器を搭載し、それまで謎だったガンマ線バーストの発生位置をつきとめることに成功した。コンプトン衛星は 9 年間に及ぶ観測期間のあいだにガンマ線バーストの観測例数を飛躍的に増やすことに成功し、合計 2500 を超えるバーストを検出した。この観測により、ガンマ線バーストは一日に 1 回から 2 回程度発生する宇宙ではありふれた現象であり、さらに宇宙全体から等方的にやってきていることが示された (図 1.1)。

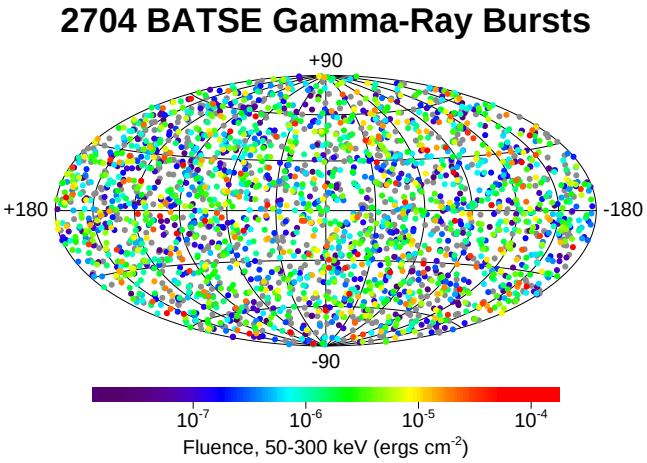

図 1.1: BATSE がとらえた GRB の分布図

減光のようす　ガンマ線バーストでは、バーストが検出されたのち数秒から数時間にわたってガンマ線が放射され、徐々にべき型で減光していくようすがみられる。この減光時間の違いによってロングバーストとショートバーストと呼ばれる 2 種類のバーストに分類されている。放射が 2 秒以上続く長時間型のバーストについては、GRB011121 におけるハッブル宇宙望遠鏡での観測にお

いて、超新星のルミノシティ変化モデルとの一致が見られた。また、GRB030329 ではすばる望遠鏡が残光の可視光分光観測に成功し、Ic 型超新星とよく似たスペクトルをもつことが確認された。これらの成果によって長期型バーストは超新星爆発が起源の現象ではないかと考えられている。放射が 2 秒以下 (ミリ秒単位) の短期型バーストは、その継続時間の短さから詳細な観測が困難であった。しかし、ガンマ線バースト観測衛星 Swift は GRB050509B の観測で世界で初めてショートバーストの詳細な位置決定をすることに成功した。このバーストは継続時間 40 ミリ秒で、地球から 300 万光年離れた楕円銀河の周辺部に位置していたことが分かり、中性子星やブラックホールの合体現象を起源とするのではないかと言われている。

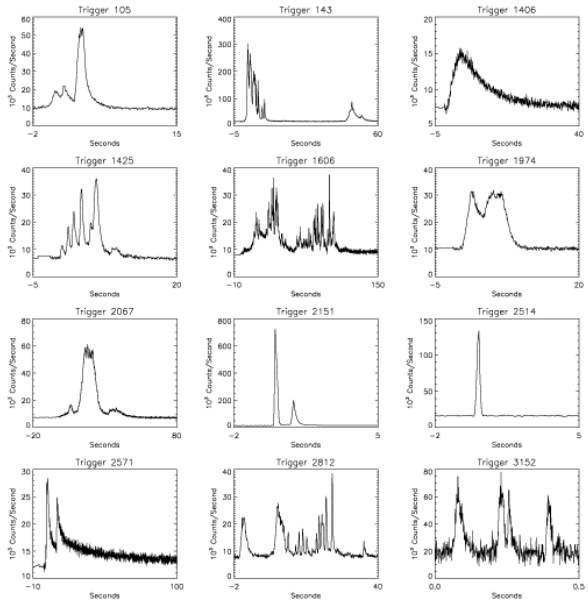

図 1.2: さまざまなガンマ線バーストの減光の様子。

1.2 ガンマ線バースト可視光同時観測の意義

1999 年 1 月 23 日、アメリカの可視光検出器 ROTSE が初めてガンマ線バーストと同時に起こった可視光閃光の観測に成功した。このバーストはガンマ線領域で 2 つのピークをもち、全体的にガンマ線バーストに典型的な時間のべき乗で減光する様子をみせている。ROTSE がとらえた可視光でのライトカーブはバースト発生後にも上昇を続けてバースト発生後 57 秒後にピークを迎え、そののち急速に減光していった。バースト開始後ピーク時の等級は 8.86 ± 0.02 等にまで達している (図 1.3)。

この観測例の後、可視光閃光を捉えるための地上観測が世界中で追随しはじめたが、ガンマ線バーストの可視光観測は今までに*例程度しか成功していない**??**。また、地上での可視光観測の多くがガンマ線バースト観測衛星のフォローアップ観測であるため、バースト発生直前直後の観測には成功していない。このようにガンマ線バーストの可視光同時観測に関してのデータはまだまだ少なく、可視光検出器による同時観測が非常に重要となっている。

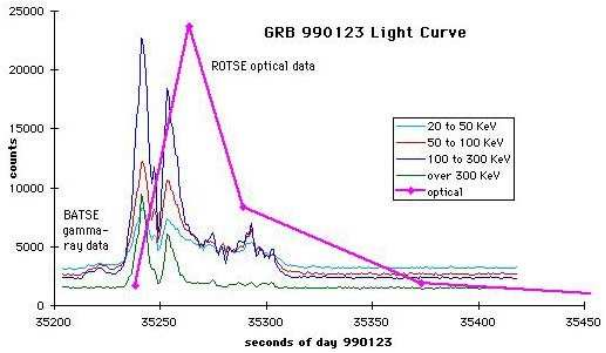

図 1.3: GRB990123 の可視光およびガンマ線でのライトカーブ

1.3 ガンマ線バースト観測衛星 Swift

Swift は日米欧で共同開発され、2004 年 11 月に NASA によって打ち上げられたガンマ線バースト観測専用衛星である。この衛星は Burst Alert Telescope (BAT)、X-Ray Telescope (XRT)、Ultraviolet and Optical Telescope (UVOT) という 3 つの検出器を搭載し、従来のガンマ線バースト観測衛星に比べて広い範囲を観測しながらガンマ線バーストを検出する。平時の Swift はオペレーションチームによって決められた観測予定により、既に起こった GRB の残光観測やその他の天体現象を観測している。このあいだに、$120^\circ \times 90^\circ$ という広い視野を持った BAT が全天の 6 分の 1 をモニターし、突発的なガンマ線バーストに備える。バーストが検出されたら、10 秒以内に衛星上でその位置を決定し、GCN として地上への通報を行う。それと同時に衛星自身の姿勢を制御してバーストの方向へ機体を向け、バーストの残光観測を行う。このように自身がバーストの方向を向くことで、XRT と UVOT という 2 つの検出器によってバースト直後の広帯域にわたる残光の様子を捉えることができる。

Swift は観測方向に地球が入り、観測を妨げてしまうことを防ぐため、衛星軌道を周回する間その視野をさまざまな方向へ向ける。この方法を取ることによって常に検出器を使った観測が可能となり、より効率的にガンマ線バーストを捉えることができる。

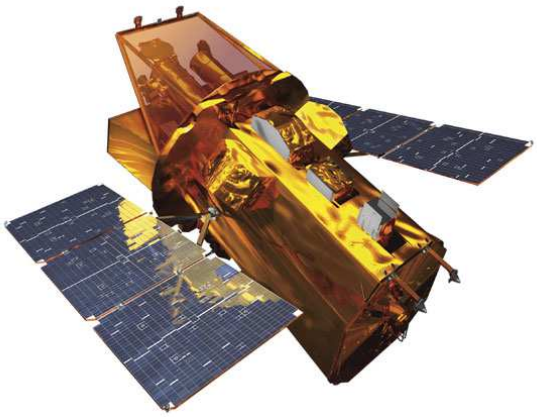

図 1.4: Swift 衛星

第2章　超広視野可視光望遠鏡 WIDGET

2.1　概要

広視野可視光望遠鏡 WIDGET (WIDe-field telescope for GRB Early Timing) はガンマ線バースト観測衛星との可視光同期観測を実現するため、理化学研究所牧島宇宙放射線研究室と埼玉大学田代研究室が中心となって開発し、2004 年 6 月に運用を開始した。

同期観測の対象にはガンマ線バースト観測衛星 Swift を用いている。WIDGET の検出器は視野角 $32.0° \times 32.0°$ の広角レンズを取り付けた CCD カメラ 4 台で、観測視野の合計範囲は最高で $64° \times 64°$ である。これは全天の約 9 分の 1、Swift/BAT 検出器の観測視野 $120° \times 90°$ の視野中心部約 3 分の 1 をモニターすることができる。

現在システムは長野県木曽郡木曽町の東京大学木曽観測所内に設置され、完全自律運用されている。また、運用ステータスが 15 分おきに World Wide Web 上に用意された専用の Quick Look Page にアップデートされる。チームメンバーはこのページから、いつ、どこからでも状況が確認できるようになっている。

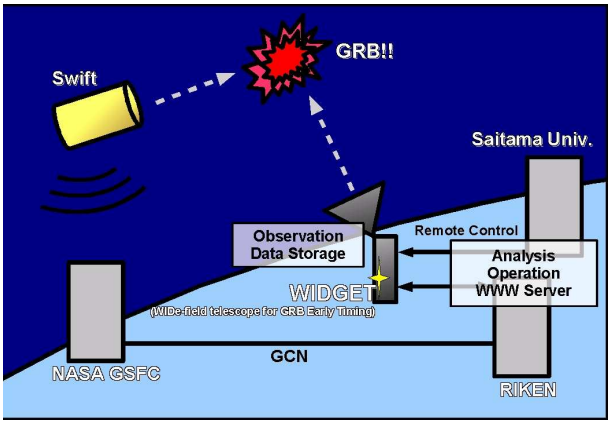

図 2.1: WIDGET の概略

2.2　観測システム

2.2.1　WIDGET のシステム構成

WIDGET は主として観測装置系、保守装置系、制御装置系に大別される。これらがそれぞれシステム内ネットワークで接続され、データのやりとりを行なうことによって WIDGET システム全体がなりたっている。

観測装置系　4 台の CCD カメラと光学系からなる検出器はそれぞれ 1 台ずつ割り振られた PC と USB 接続され、専用のソフトにより撮像とデータの読みだしが行われる。われわれは、5 秒間の露光と 5 秒間の読みだしによって一枚の画像を取得している。このサイクルを一晩中繰り返しつつ、常に夜空を撮影している。こうして得られたデータは、システム内ネットワークを通じてデータ保存用の PC に蓄積された後、外付ハードディスクドライブに保存される。また、検出器を載せる赤道儀は専用 PC とシリアル接続され、Linux 上のプログラムで制御している。プログラムは観測予定データをリアルタイムで読み込んで、赤道儀の向く方向を決定する。

保守装置系　システム全体は観測小屋に収められて、風雨や降雪などから守られている。さらに小屋の内外には監視カメラと気象センサーが配置され、常に周囲の状況をモニターしている。気象センサーのひとつである雨滴センサーと小屋の開閉システムはリレーボードを通じて直結し、雨滴を検出したら即座に小屋の屋根が閉まるようになっている。

制御装置系　これらの機器類は専用の Linux PC *台によりわれわれが独自にアレンジした制御プログラム群によって運用されている。また、WIDGET システムは World Wide Web 上での固定 IP アドレスを取得しており、外部ネットワークからのリモートアクセスも可能である。

2.2.2　赤道儀

我々は観測対象となる天体の運行を精確に追うため、CCD カメラの架台として高橋製作所 赤道儀 NJP Temma2 を用いている。カタログスペックを表 2.1 に示す。

赤道儀とは、あらかじめ地球の極軸方向と並行になるような固定軸を設定し、その上で赤経方向に回転する軸 (赤経軸) と赤緯方向に回転する軸 (赤緯軸) によって天体の日周運動を追尾する装置である。Swift 衛星の観測情報は観測対象の天球座標 (赤経、赤緯) によって与えられるので、この値を赤道儀に入力することで衛星との同期観測ができる。これらの手順は赤道儀を Linux PC とシリアル接続することにより、コマンドライン上でも行うことができる。

図 2.2: 赤道儀

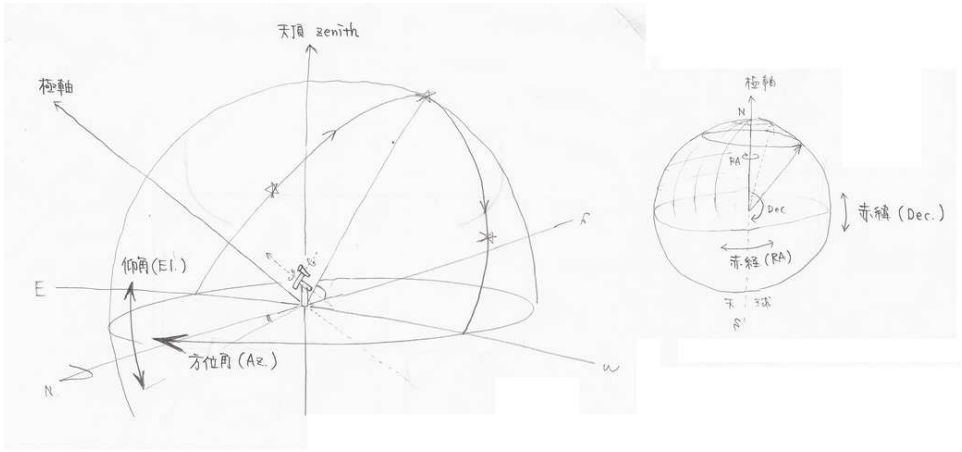

図 2.3: 赤道儀の視野方向

表 2.1: 高橋製作所 NJP Temma2 カタログスペック

形式	２軸モーター外付、ドイツ式赤道儀
赤経微動	ウオームホイル全周微動 (減速比240：１) ステッピングモーターによる電動駆動 (手動は不可) ハイスピード、ノーマル駆動モード切替可能
赤緯微動	ウオームホイル全周微動 (減速比144：１) ステッピングモーターによる電動駆動 (手動は不可) ハイスピード、ノーマル駆動モード切替可能
方位微動	ダブルスクリュー方式、可動範囲±５゜
傾斜角微動	スクリュー式、可動範囲　高度 25゜～48゜
目盛環	赤経：最小目盛 10 分赤緯：最小目盛２゜
搭載質量	約 30kg
本体質量	約 24.5kg（バランスウエイト別)
極軸望遠鏡	固定内蔵式、据付精度　約 2’＊北極星用歳差補正目盛（2015 年まで対応）＊南極星八分儀座 σ 星用歳差補正目盛 (2015 年まで対応)
駆動方式	両軸駆動、水晶発振制御、回転誤差± 0.05 % (対恒星時)
駆動周波数	約 200PPS
高速駆動	電源ＯＮ時の電源電圧により、最高速を自動選択
DC24V 時	赤経方向：約 350 倍速赤緯方向：約± 5250 秒角／ sec.
DC12V 時	赤経方向：約 175 倍速赤緯方向：約± 2625 秒角／ sec.
補正駆動	赤経方向：0.1～1.9 倍　(対恒星時 0.1 倍刻み) 赤緯方向：± 1.5～13.5 秒角／ sec.（1.5 秒角刻み) ハンドボックスのボタン操作により設定
電源電圧	定格 DC12V、または DC24V *WIDGET では DC12V を使用
消費電流	0.8～2.1 A（使用電圧により変化）
オプション	パソコン接続ケーブル、THC ハンドコントローラー

2.2.3 冷却 CCD および光学系

可視光観測には、CCD カメラによる撮像を用いる。CCD (Charge Coupled Device) を用いる利点として、CMOS センサーなどと比べて感度が良いことが挙げられる。また、長時間露光に起因するノイズを減らすため冷却 CCD を利用している。

可視光観測機器では、バックグラウンドノイズと紛れることなく有意に検出できる明るさの限界のことを、限界等級という。限界等級はさまざまな要因によって決まるが、検出器の空間分解能の高さがそのひとつである。空間分解能とは検出器が判別できる 2 点間の距離のことで、これが高い場合 1 画素中に切り取られる空間領域が狭くなり、必然的に入ってくるバックグラウンドの光量が少なくなる。すると、これと比較して有意に検出できる星の明るさを低くすることができるようになる。つまり、空間分解能をあげることで限界等級を高めることができるのである。

ここで、レンズの焦点距離 (f) と CCD 受光面の辺の長さ (d)、画像の辺画角 (θ) の間には

$$\theta = 2 \times tan^{-1}\left(\frac{d}{2f}\right) \tag{2.1}$$

という関係式がある。この式と CCD 受光面の画素数 (P) を用いると、空間分解能 (R) は

$$R[deg^2/pix] = \frac{\theta^2}{P} \propto \frac{1}{f \times P} \tag{2.2}$$

と書けるから、空間分解能を小さくするためにはレンズの焦点距離を長くするか、受光面の画素数を増やせばよい。

また、レンズには像の明るさを決める値として F 値というものがある。これはレンズの口径を 1 としたときの焦点距離の長さを表す値で、この値が小さいほど受光面に対するレンズの口径が大きくなるため、撮影した画像は明るくなる。

Swift/BAT 検出器は 120° × 90° という広い検出域をもつため、同時観測望遠鏡である WIDGET としては、検出時の限界等級はできるだけおとさずになおかつ広視野をカバーしたい。以上の点を踏まえ、WIDGET では CCD カメラには Apogee Alta U10、カメラレンズには Canon EF50mmF1.4 USM および EF35mm F1.4L USM を使用している。

Apogee U10 は CCD イメージセンサに TH7889 (総画素数 2048 × 2048、検出エリア 28.6 × 28.6mm) を採用している。表 2.2.3 にその基本スペックを示す。画像取得は露光に 5sec 、読みだしに 5sec の合計 10sec で行う。Canon EF35mm および 50mm は焦点距離 35mm、50mm の広角カメラレンズで、基本スペックを表 2.2.3 に示す。

これらの CCD とレンズのセットによる視野角は式 2.1 により、32.0°×32.0° (50mm)、44.5°×44.5° (35mm) となる。WIDGET ではこれらの検出器を用いることにより、現在までに限界等級を 14.7 等まで深めることに成功している [1]。

また、周辺機器の LED 光などを排除するため、レンズのまわりに遮光用のフードを製作し取り付けている。

2.2.4 観測小屋

屋外での天体観測においては、風雨への対策が重要である。WIDGET システムでは観測装置と制御装置を木曽観測所の施設とは独立した観測小屋に納めている。観測装置の広い視野を最大限に生かすためにヒューマンコム (株) と共同開発した専用の観測小屋 (2m × 3m × 2m) を利用している。観測時には一方向へスライディングルーフが移動し、それ以外の三方へは広く視界がとれるように設計されている。また、おもりによって屋根が閉められる補助機構を備え、落雷などによる突然の停電でも無停電で閉鎖できるようになっている。

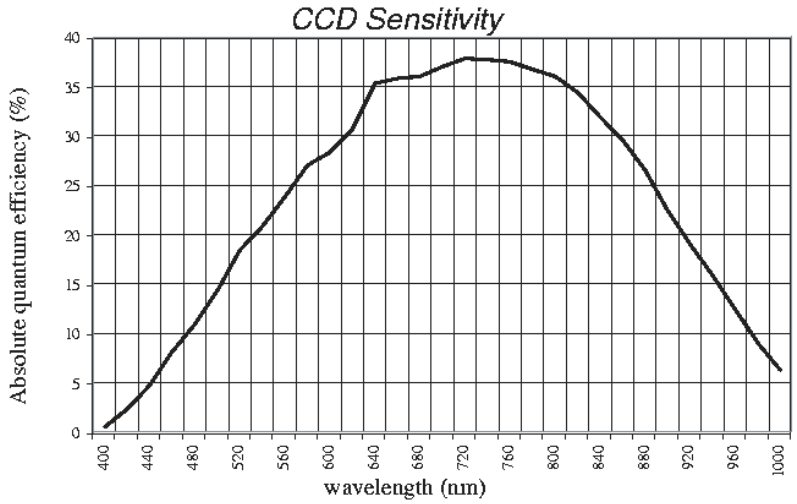

図 2.4: U10/TH7889 の波長に対する検出効率

表 2.2: Apogee Alta U10 のカタログ値

CCD	TH7889
画素数	2048 × 2048 (419 万画素)
画素サイズ	14μ
検出エリア	28.6 × 28.6mm
フルウェル	270.000 (e-)
ダイナミックレンジ	80dB
ピーク QE	38%@720nm
冷却 (対環境温度)	-45c
マウント	2inch

表 2.3: Canon EF35mm/EF50mm のカタログ値

	EF35mm F1.4L	EF50mm F1.4L
レンズ構成	6 群 7 枚	9 群 11 枚
撮影距離範囲	0.45m〜∞	0.3m〜∞
最大撮影倍率	0.15 倍	0.18 倍
フィルター径	58mm	72mm
最大径×長さ	φ 73.8 × 50.5mm	φ 79 × 86mm
質量	290g	580g
型番	EF5014U	EF3514L

2.2.5 自動観測システム

可視光での天体観測作業は当然夜間に行われ、明け方に終了するので、日没、日の出にあわせた小屋の開閉、赤道儀、データ取得の制御が必要になる。これらに加え、十数分単位で観測対象の変化する衛星の運用データに対応しなければならないことから、WIDGET ではさまざまな自律

図 2.5: 観測小屋の全景

運用システムを採用している。これにより日々の技術的な運用作業を減らすことができる。

雨センサー

観測小屋が設置されている周囲 3 箇所に雨滴センサーを設置し、現地の降雨状況を監視している。このセンサーによって雨天時にも装置類が濡れることのないように、屋根の自動開閉を行うことができる。

観測小屋の自動開閉

観測小屋のスライディングルーフは、制御 PC で実行されている専用のプログラムによって自動開閉される。このプログラムは以下のような動作を行う。[4]

- 小屋に設置されたリレーボードにソケットでコネクションを確立し、小屋の開閉コマンドを送ったり、センサーの情報を受け取る。
- 太陽が沈む時刻に小屋を開け、日の出時刻に小屋を閉める。
- 小屋の運用に問題が発生した際にチーム全体にアラート メールを送信する。

赤道儀の自動運用

赤道儀も制御 PC で実行されるプログラムによって自動運用されている。このプログラムは小屋開閉プログラムとは独立に毎日 17 時から実行され、日没時刻から衛星の追尾を開始して日の出の時刻に運用を終えるようになっている。また、視野が小屋の壁や周囲の森林などによって視界をさえぎられる範囲や、観測装置が赤道儀自身と接触して壊れるおそれのある範囲に動かないように制御するはたらきもしている。

無停電電源

WIDGET システムには、停電時の対策として有効出力電力 1400VA の無停電電源 (UPS) が導入されている。運用に死活的に重要である PC やネットワーク関連機器などは UPS に接続され、停電時にも最低限安全に運用停止できるまでの電源が確保できるようになっている。

データストレージ

CCD カメラによって撮影された画像はシステム内ネットワークを通じ、データストレージ用の PC に転送される。これらの画像データは観測中一時的に PC 内部に保管され、観測終了後に外部接続された専用のハードディスクへと保存される。一晩で得られるデータは CCD 一台で 3GB 程度、4 台運用した場合 12GB にも及ぶため、予め 1TB 程度の大容量ハードディスクを接続しておき、適当な時期に回収して理研や埼大でのデータ解析に供される。

2.2.6 遠隔監視システム

観測データの限界等級を下げる条件のひとつとして、観測場所周囲の市街光の影響を無視することはできない。この条件を満たす観測場所として東京大学木曽観測所が選ばれた。常時観測所に滞在して運用作業をすることが困難なため、埼玉大学や理化学研究所などから遠隔操作で状況を監視できるように以下のシステムが導入されている。

監視カメラ群

小屋内部に 2 台、外部に 2 台の Web カメラが設置されている。これらのカメラからの映像は PC に静止画像としてキャプチャされ、任意の時間帯の画像を確認することができる。内部視点は赤道儀や CCD の状況確認に、外部視点は小屋の状況確認や天候の確認に特に有効である。

気象センサー群

WIDGET では気圧計、温度計、湿度計、風力計によって現在の気象情報をリアルタイムに取得する。

Web を通じてのステータス確認

すでに述べたように現在のシステム全体のステータスをどこからでも確認できるように、理研の Web サーバ上にチーム専用のステータス確認ページが作成されている。このページでは小屋の開閉情報、CCD の状況、天候、ポインティングプロット、PC のディスク容量、監視カメラ画像などが 15 分間隔でアップデートされる。また、毎日午前 9 時には前日一日分の運用データがまとめられ、メンバーがシフト制で異常がないかを確認する。

第3章　WIDGETの改良

WIDGET システムは現在までにもガンマ線バースト観測衛星との同時観測に成功している。しかしいまだ可視光での有意な観測結果は得られておらず、システム全体にわたって改良の余地がある状態である。そこでわれわれは 2006 年 9 月から現在にかけて、WIDGET1.5 から WIDGET2 への大幅なシステムの改良作業を進めている。本章では主に私が関わってきた改良点を取り上げる。

3.1　WIDGET2への改良のポイント

実際に CCD カメラで星空を追尾するためには、赤道儀と観測機器類を接続しなければならない。2.2.3 で述べたように、われわれのシステムでは広範囲の星空を観測するために複数の CCD カメラを用いているので、観測機器類の配置に独自の方法を用いる必要がある。今回 WIDGET2 での新たな観測体制に対応するため、CCD カメラを搭載するためのプレートの設計と発注を行った。これを 3.2 節で述べる。

これまで WIDGET システムが追尾しているガンマ線バースト観測衛星は、HETE-2 と Swift の 2 台であった。しかし HETE-2 の経年劣化による観測効率の低下に伴い、2006 年 12 月 14 日より Swift のみを追尾している。今回の改良では Swift がこれから観測する方向に予め視野を向ける観測方法を追加した。これによりその視野内でガンマ線バーストが起こった際に、バースト検出以前の観測実現を目指す。これを 3.3 節で述べる。

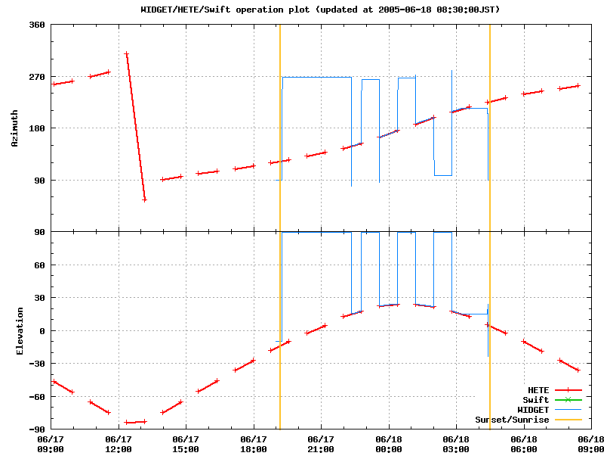

図 3.1: HETE-2 観測位置情報

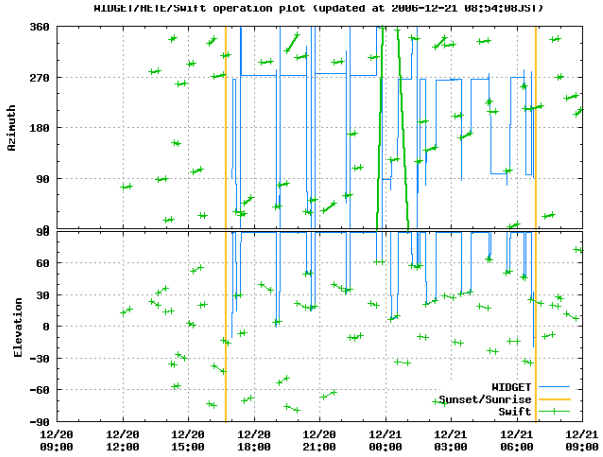

図 3.2: Swift 観測位置情報

3.2 CCD カメラ搭載プレートの設計

WIDGET1.5 では、3 台の CCD カメラと焦点距離 35mm のレンズを使用しておおよそ 89.1° × 89.1° × 3 の範囲を観測していた。WIDGET2 ではレンズを焦点距離 50mm のものに換え、さらに CCD を一台追加投入することによって広視野のまま高い感度でモニターすることが目標である。また、今まで使用してきた 3 種類のレンズ (Canon EF 24mm/35mm/50mm) を継続して使用することも考え、CCD カメラ 4 台と各レンズの任意に組み合わせに対応できる体制が望ましい。そのためには CCD カメラを設置する接続プレートの形状が重要になってくる。適切な視野方向を確保しつつ、赤道儀や小屋など他の観測装置と干渉しないような形状を考え、シーアイ工業 (株) との共同製作を行った。

3.2.1 各 CCD カメラの視野の重ね具合

ここで観測地点を原点とした極座標系を考えて、CCD カメラの視野を半径一定の球面に投影すれば、球面上における面積が求められる。この単位を平方度といい、天球上の範囲が容易に指定できる。さて、焦点距離 50mm のレンズの視野角は 32.0° × 32.0° (平方度) であることから、各カメラの視野が重ならないように、図**??** のように並べたい。いま観測地点における各カメラの位置のずれは観測対象となる天球までの距離に対して無視できる大きさであるから、原点から 32.0° × 32.0° の広がりをもった視野角を 4 つ並べると考えればよい。このとき単位球面上にある各カメラの視野中心ベクトル $\vec{r}_i(\theta_i, \phi_i)$ は、検出器全体の視野中心ベクトルの座標 $\vec{r}_0(\theta_0, \phi_0)$ を用いて

$$\vec{r}_i(\theta_i, \phi_i) = \vec{r}_i(\theta_0 \pm 10.75^\circ, \phi_0 \pm 10.75^\circ), \quad (i = 1, 2, 3, 4) \tag{3.1}$$

と表せる。したがって、カメラの視野を重ねずに配置するためには各カメラの視野方向をプレートに固定されたある基準方向から (θ, ϕ) 方向にそれぞれ 16.0° ずつ回転させて配置すればよい。
われわれが用いている Apogee U10 CCD カメラはカメラ本体底部のネジ穴 1 点および、カメラマウント部に接続した L 字型金具のネジ穴 2 点によって面上に固定する。この構造を利用して、上記の経緯度方向への回転は、

- 接続面上にそったカメラ本体の回転
- プレート底面に対するカメラ接続面の回転

の 2 つの方法によって実現した。
この方法により、オプションとして全てのカメラを同一方向へ向けることも可能である。

3.2.2 重量および強度

プレートに搭載する機器類は CCD カメラ 4 台、レンズ 4 台で、これらの総重量が約 **kg となる。また、NJP Temma2 の積載重量は表 2.1 より約 30kg が限度であることから、プレート本体は **kg 以内に収める必要がある。また搭載する装置へぶれが伝わったり、どのような姿勢になっても装置が落下したりすることがないように、十分な強度をもった構造でなければならない。
以上を満たす構造材として、アルミニウム合金 A5052 9mm を選択した。以下にその物理性質をあげる。また、可視光検出器におよぼす影響を減らすため、表面には黒塗り仕上げを施した。

3.2.3　周辺機器との干渉

できるだけたくさんの衛星の観測対象を追尾するためには、観測装置ができるだけ自由に運動できることが望ましい。プレートの形状によって赤道儀の動きに制限がかからないようにプレート全体をなるべくコンパクトに収めることも重要である。

以上の条件を考慮した結果、次のようなプレートを設計した (表 3.1)。

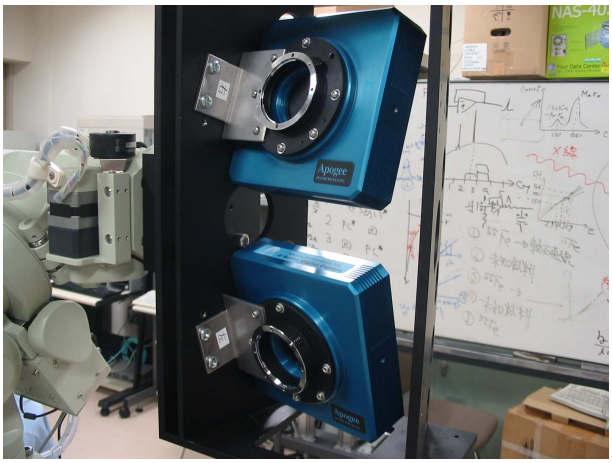

図 3.3: CCD を 15° 開いて取り付けた状態

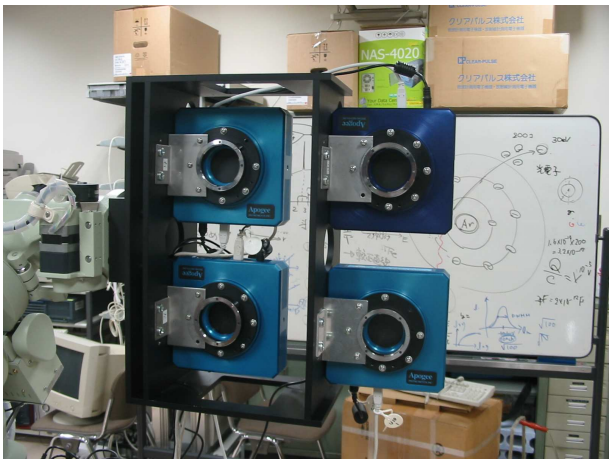

図 3.4: プレートと 4 台の CCD カメラ

図 3.5: 赤道儀と接続した CCD プレート

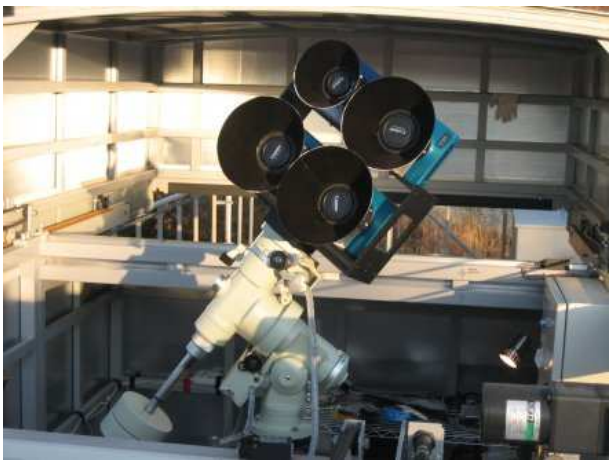

図 3.6: 実働状態のプレート、CCD、赤道儀

表 3.1: プレート設計基本スペック

構造	2 段 2 列 横板可動式 (底板に対し 0 ° /15 ° 傾斜)
サイズ (W × D × H mm)	416 × 155 × 250
材質	A5052 9mm
塗装	つや消し黒塗装

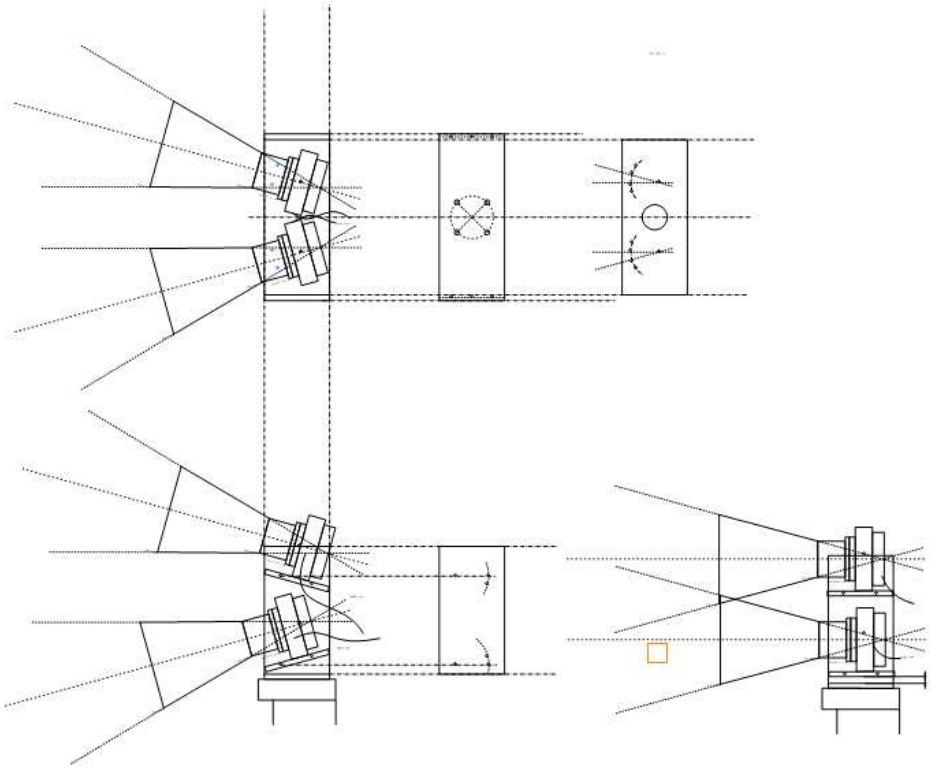

図 3.7: CCD プレート設計図

3.3 Swift 衛星追尾プログラム

3.3.1 Swift 衛星の観測予定情報

現在 WIDGET2 が追尾している Swift 衛星は、運用チームによって決められた Observation Schedule[] によってありとあらゆる方向に視野を向けることが特徴である (図 3.8)。Swift 衛星の運用スケジュールには GRB 残光観測以外のものが多数含まれる。このうち現在の WIDGET システムで同時観測可能な観測対象は 3 分の 1 程度に過ぎない (図 3.9、表 3.2) 。今回はできる限り GRB を観測する機会を増やすために、Swift 観測位置を予め先取りして観測するようにプログラムを改良した。

表 3.2: Swift 観測位置情報

WIDGET 視野内の個数	22824
WIDGET 視野外の個数	6593
合計	29417
その中でとらえた GRB の数	209

3.3.2 観測データ変換スクリプト

Swift 観測予定情報は、WWW 上の http://www.swift.psu.edu/operations/obsSchedule.php で提供され、予め 1 日から 2 日程度先の観測予定が分かるようになっている。この情報を取得して

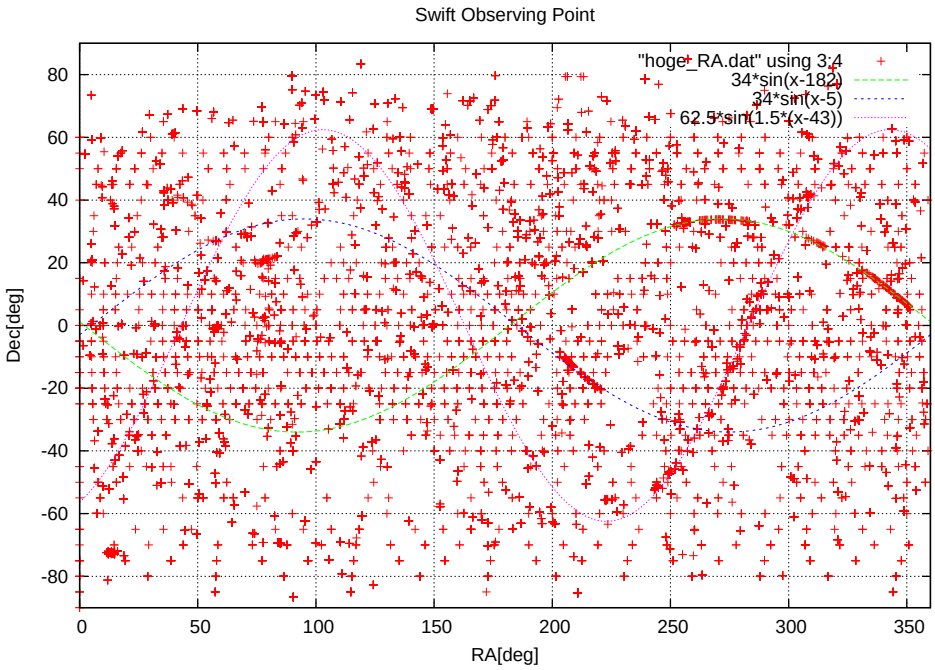

図 3.8: Swift 衛星の観測位置情報 (2005.4.4-2007.1.20) の分布図

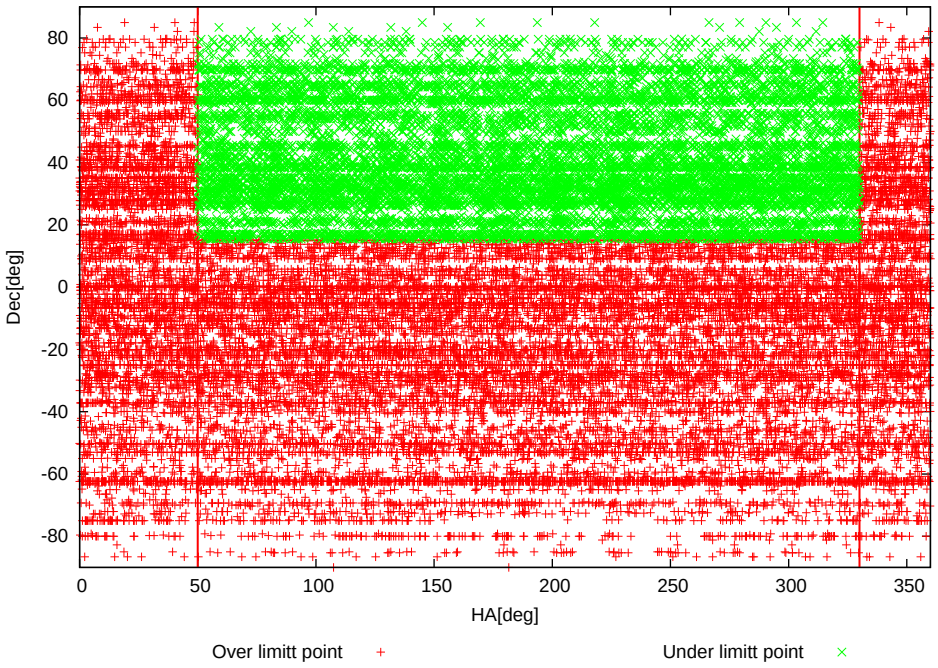

図 3.9: WIDGET の観測可能視野と Swift 衛星の観測位置情報の関係

WIDGET の赤道儀制御プログラムで利用できる形式にするために、次のようなデータ変換スクリプトを作成した (付録 A.4)。

Swiftcat2.pl

- 現在の日時から、この先一日の観測データファイルの URL を計算する。

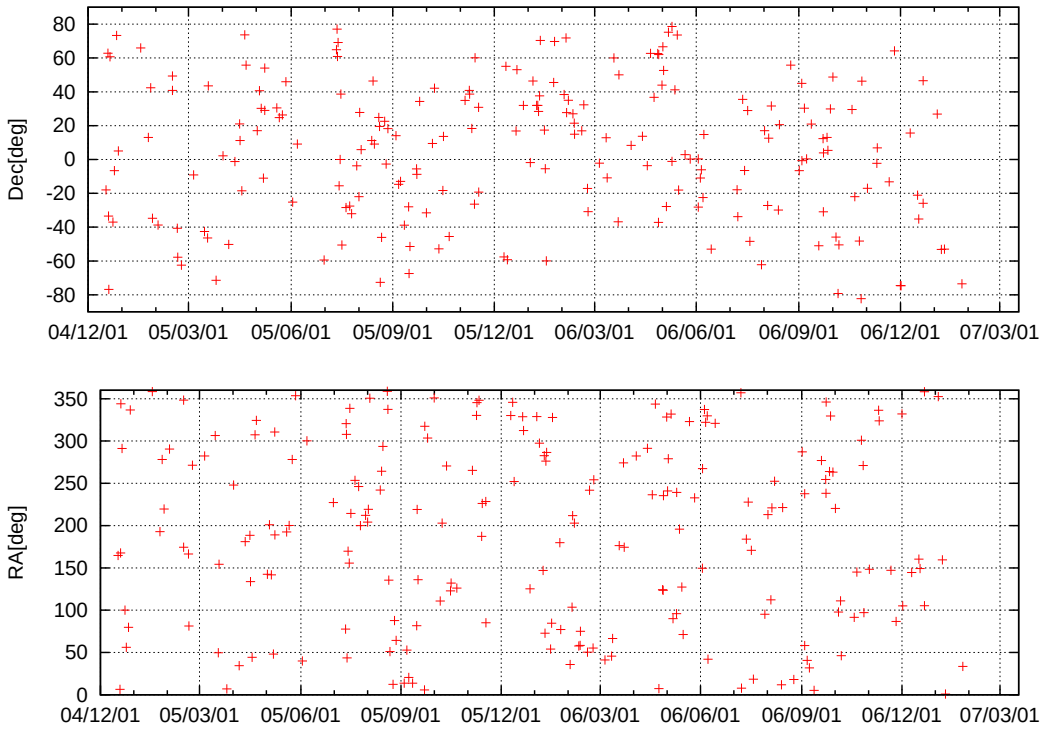

図 3.10: Swift で観測されたガンマ線バーストを時系列に並べた一覧図

- この URL にアクセスし、html ファイルを取得する。

Swiftcat3.pl

- 受信した html ファイルを読み込んで、テキストファイルに変換。
- 観測データは UT (国際標準時) 及び年間通算日でかかれているため、これを JST (日本標準時) の年月日スタイルに変換。
- 赤道儀制御プログラムに必要な情報のみを取り出し、整形して出力する。

Swift_info.csh

- Swiftcat2.pl と Swiftcat3.pl を一連の流れとして実行し、Swift_data.txt というファイルに Swift 観測予定情報を出力する。

これらの変換スクリプトは同じディレクトリに置いておく。また、http を用いて html ファイルを取得するための getHTTP.pl という外部スクリプトを使用するため、これも同じディレクトリに用意する。もし他のディレクトリあるいは他のマシンで使用する場合は、Swift_info.csh 中のディレクトリ情報を書き換えればよい。以上の準備が済み、実際に Swift 観測予定情報を読み込むには以下のコマンドを実行する。

スクリプトの置いてあるディレクトリで

```
./Swift_info.csh
```

3.3.3 赤道儀制御プログラム

赤道儀制御プログラムの衛星追尾システムは、前節のデータ変換スクリプトで作成されたデータファイルをファイルを読み込み、次にどの方向を向くとよいかを判断して WIDGET 観測方向を決定し、そののち実際に赤道儀を制御する。

今回プログラムを書き換えるにあたって、Swift 観測予定位置が WIDGET 観測方向に適切であるかどうかを判断するアルゴリズムを独立したモジュールとして作成した (付録 A.8。今後他の衛星を追尾することになったり、それ以外にも WIDGET の観測方向を決定するための新たなパラメータが増えた場合は、これと同様に独立したモジュールを追加していけばよいだろう。

hoge2.c

Swift 観測予定情報を読み込んで、WIDGET で観測可能かどうかを判断する。

関数

`int read_swift(double *, double *)`
引数: 現在の赤道儀座標 (RA, Dec) ポインタへのアドレス
返り値: Swift_status = 0 or 1 or -1

`int swift_point_status(double *, double *)`
引数: 判断する Swift 観測位置座標 (RA, Dec) ポインタへのアドレス
返り値: swift_point_status = 0 or 1

アルゴリズム

- Swift_data.txt を読み込む。読み込めない場合、LOGFILE にエラーメッセージを書き出して、Swift_status に -1 を返して終了。
- ファイルから一行ずつデータラインを読み込むループを開始。
- 読み込んだデータラインを分割して個別のデータに変換。
- 現在時刻が観測時間内にマッチしたら、観測位置情報 (RA, Dec, status) を記憶してフラグを立てる。
- フラグが立ったシークエンスで次の観測位置情報 (RA', Dec', status') を記憶してループから抜ける。
- 観測位置情報の status を用いて条件分岐し、
  1. 現在の Swift 観測位置が WIDGET で観測可能
  2. 現在の Swift 観測位置が WIDGET で観測不能で、
     次の Swift 観測位置が WIDGET で観測可能
  3. それ以外

  のそれぞれについて、Swift_status に 0, 1, -1 を返して終了。

hogemain.c

main 関数を含むプログラム。

関数

`int main(int, char[] *)`
引数: オプション
返り値: -

アルゴリズム

- 今日の日暮れ、日没、夜明け、日の出の時刻を計算。
- NJP デバイスを開く。
- ホームポジション (Az, El) = (90.0, -10.0) へ向ける。
- 日没までホームポジションで待機。
- 日没時刻になったら、観測ループ開始
  - 日の出時刻になっていたら、観測終了。
  - Swift status と Swift pointing の読み込み
  - Swift pointing が次に移ったかどうかを判断。
  - メインコントロールルーチン
    NJP の status で switch
    * ホームポジションもしくは invalid ポジションの場合
      · if(Swift_status = 0) : 現在の Swift pointing へ向ける
      · if(Swift_status = 1) : 次の Swift pointing へ向ける
      · if(Swift_status = -1) : 天頂へ向ける
    * Swift pointing を観測している場合
      · if(Swift_status = 0) : そのまま現在の Swift pointing を追う
      · if(Swift_status = 1) : そのまま次の Swift pointing を追う
      · if(Swift_status = -1) : その場で statndby-on 、status = -1
    * 天頂を向いている場合
      · if(Swift_status = 0) : 現在の Swift pointing へ向ける
      · if(Swift_status = 1) : 次の Swift pointing へ向ける
      · if(Swift_status = -1) : そのまま天頂を追う
    * その他の場合 その場で待機し、status = -1
  - 現在の観測位置を表示して、30 秒待機
- 観測終了後、ホームポジションへ向ける。
- standby-on にして、デバイスを閉じて終了。

3.3.4 改良後の動作評価

operaton plot。追尾時間の増加率。↑期待できる GRB 検出の増加率。

第4章　まとめと今後の課題

今回新たにCCDプレートを設計、製作したことによってCCDカメラ4台同時観測体制がとれるようになり、これと Canon EF50mm レンズを組み合わせることによって観測視野を 64° × 64°、分解能を 0.94arcmin とすることができるようになった。これにより、広い範囲の観測と観測精度の両立を実現した。またプレートの横板とカメラ接続面を可動式とすることで、2台以上のカメラでの同方向比較観測も可能になった。

Swift追尾プログラムの改良によって、今までは天頂を向いていた状況でもSwiftの観測方向を先取りして観測することができるようになり、Swiftがバーストを検出したときの早期可視広閃光を検出できる可能性が大きくなった。また、赤道儀の操作プログラムからSwift観測位置のステータス判断アルゴリズムを別離することにより、今後新たな視野方向の判断基準を導入しやすくするようにした。

今後の課題としては、Swiftの観測予定情報の情報源が WWW 上の Operation Schedule のみの場合、Swiftのバースト発生後の変更スケジュールや GCN のバーストアラートに対応できないことが挙げられる。これに対応するためには、今までの Alart Mail による Swift Pointing データの読み込みを同時に行うことが必要になる。方針としては Swift_data.txt ファイルを WWW 上の Operation Schedule 用と Alart Mail 用の2種類用意し、それらが異なる際には常に Alart Mail から得た観測予定情報を優先するプログラムにすればよいと考えられる。そのためには hoge2.c に2種類の Swift_data ファイルを比較させるモジュールを付け加えたい。これにより WIDGET2 への実装が可能となる。

また、現在 WIDGET の視野判定は視野中心座標のみで行っているため、例えば視野中心が skyline よりもわずかに下回る場合でも天頂 (もしくは次の Swift 観測予定方向) へと向くようになってしまう。このため視野の半分程度がSwiftと同時観測できていても、観測不能として捨ててしまっている場合がある。これを改良できれば、Swiftとの同時追尾時間がさらに増加し、バーストの瞬間をとらえる可能性がさらに増える。対象天体が WIDGET の視野内に入っているかどうかを判定する 吾妻スクリプト を改造し視野判定モジュールに組み込むことができれば、視野の蹴られ具合に閾値を定めて WIDGET の視野判定をすることも可能になるはずである。これも今後の課題である。

関連図書

[1] 小高卒論 (2007)

[2] 恩田卒論 (2005)

[3] 恩田修論 (2007)

[4] 玉川徹, 白井文彦, WIDGET オペレーションマニュアル, 2006

[5] Akerlof, C., et al., 1999 Nature, 398, 400

[6] http://cweb.canon.jp/camera/ef/description/knowledge/index.html

[7] http://www.ne.jp/asahi/starfactory/home/tips/index.html

[8] http://www.nao.ac.jp/nao_topics/data/000103.html

[9] http://www.astroarts.co.jp/news/2005/05/18short_gamma-ray_burst/index-j.shtml

[10] http://www.astroarts.co.jp/news/2006/09/05xray_flash/index-j.shtml

[11] http://www.astro.isas.jaxa.jp/ takahasi/swift.html

[12] http://www.subarutelescope.org/Pressrelease/2003/06/j_index.html

[13] http://ja.wikipedia.org/

謝辞

本研究をすすめるにあたって、WIDGET チームメンバーを中心に沢山の方々にご指導をいただきました。理研の玉川徹先生には、問題につきあたって立ち止まっているときに的確な助言をいただき、幾度となく目からウロコが落ちました。玉川先生と出会わなかったら WIDGET への道をすすんでいなかったと思います。本当にお世話になりました。PD の浦田裕次さんには常に身近に、天文、物理、レンズ、PC、論文の書き方やその他にもいろいろと時に厳しく時に丁寧に教えて頂きました。研究の方向に対するアドバイスを日々いただけたことで、本研究をすすめていくことができました。ありがとうございました。D3 の阿部圭一さんは私の心の師匠です。WIDGET のプロジェクトに関わらず、研究室生活における大先輩でした。阿部さんから暗に陽に学んだことは数知れません。真夜中に二人で食べに行った不二家のケーキ美味しかったです。M1 の恩田香織さんには最も身近な先輩として私の初歩的な質問にも丁寧に答えていただき、ありがとうございました。特に星や望遠鏡に関しては全くの素人だった私に対し、恩田さん自身の貴重な体験談を踏まえつつ、天文学への扉を開いてくださりました。東京理科大で理研の研修生である M1 のくわはらまことさんとは、WIDGET という大学の垣根を越えたチームプロジェクトを通じて沢山の貴重な体験を共有できました。楽しかったです。くわはらさんの明るい雰囲気と「一緒に頑張ろう!」という言葉があったからこそ、WIDGET の仕事を何とかこなせてきた気がします。また、小高さんとは同じ卒研生として、ゼロから一緒に学んできました。同じ学年の戦友は心強かったです。ありがとう。

私が主に関わった赤道儀プログラムの改良は、宇宙研の臼井文彦さんのお仕事を引き継ぐかたちですすめて参りました。直接お会いする機会はありませんでしたが、そのシステムを通じてたくさんの勉強をさせていただきました。ありがとうございました。

最後になりましたが、指導教官である田代信先生にはここには書き尽くせない程お世話になりました。そもそも先生のお人柄に触れ、田代研究室を卒研の研究室として選択したことで、一年間大変貴重な経験をすることができたと思います。この一年間卒研生として田代研に所属し、WIDGET チームで卒業研究ができたことを誇りに思います。

周囲の方々には多大なご迷惑とご心配をおかけしつつ、なんとか無事卒業論文の完成そして大学卒業へとたどりつくことができました。一年間という短い間ではありましたが、みなさま本当にありがとうございました。

付 録A　プログラムソース

A.1　calc_skyline.c

```
/*********************************************************************/
/*                                                                   */
/*  WIDGET --- WIDe-field telescope for GRB Early Timing             */
/*  Calculate limit of elevation from skyline.dat file               */
/*  Written by Masuno Keisuke - 2006/10/06 Ver 1.0                   */
/*                                                                   */
/*********************************************************************/
#include "common.h"

/*********************************************************************/
/**           skyline.dat を読み込んで、el_min を返す関数                **/
/*********************************************************************/
double calc_el_min(double *, double []);

double calc_skyline(az)
    double *az;
{
  double el_min = 30.000 ; // 危険回避のため、初期値は 30 ° に設定
  char string[256];

  /** skyline の直線式を出すため、az, el それぞれ２点を配列にする **/

  static double a[4] = {0,0,0,0}; //[0]:az1, [1]:el1, [2]:az2, [3]:el2
  FILE *fp;

  /** (skyline.dat が開けなかったとき、el_min 30 を返しておしまい) **/
  if ((fp = fopen("skyline.dat", "r")) == NULL){
    return 30.000;
  }

  while((fgets(string, 256 - 1, fp)) != NULL){

    sscanf(string, "%lf %lf", &a[0], &a[1]);

    /** az1 が az より大きくなったとき el_min を計算 **/
    if (*az <= a[0]){
      el_min = calc_el_min(az, a);
      return el_min;
    }

    /* それ以外のとき、az1,el1 を az2,el2 に置き換える */
    else {
      a[2] = a[0];
      a[3] = a[1];
    }
  }
}

/*********************************************************************/
/*     (az1, el1), (az2, el2) から直線の式を求めて el_min を計算する    */
/*********************************************************************/
double calc_el_min(double *x, double a[])
{
  double az = (*x);

  double az1 = *(&a[0]);
  double el1 = *(&a[0] + 1);
  double az2 = *(&a[0] + 2);
  double el2 = *(&a[0] + 3);  // 引数として与えられた配列の値を名付け直した
```

```
  /* 直線の傾きと、直線の式。az から el_min を求めておわり。*/
  double dydx = (el2 - el1) / (az2 - az1) ;
  double el_min = dydx * (az - az1) + el1;
  return el_min;
}
```

A.2 RADec2azel.csh

```
#! /usr/bin/perl
use constant PI
while(<>){
    @a = split(" ");
    for($i=1;$i<=2;$i++){
@b = split(/:/,"$a[i]");
$HA = ( $b[0] * 15 + $b[1] * 0.25 + $b[2] * 0.00417 )*PI/180;
$sin_h = sin(36)*sin($a[0])+cos(36)*cos($a[0])*cos($HA);
$h = asin($sin_h);
$cos_A = (cos(36)*sin($a[0])-sin(36)*cos($a[0])*cos($HA))/cos($h);
$A = acos($cos_A);
    printf("$h $A\n");
    }

}
```

A.3 Swift_info.csh

```
#! /bin/csh -f
set bindir = /home/masuno/widget/Swift/bin
set datadir  = /home/masuno/widget/Swift/dat
set htmlfile = ${datadir}/Swift_data.txt.html
set datafile = ${datadir}/Swift_data.txt
cd ${bindir}
./Swiftcat2.pl ${bindir} ${htmlfile}
./Swiftcat3.pl ${htmlfile} > ${datafile}
##rm ${htmlfile}
```

A.4 Swiftcat2.pl

```
#!/usr/bin/perl
##############################
use Math::Trig;
#use Astro::SLA;
my $R2D = 180.0 / pi;
my $D2R = pi / 180.0;
$bindir = $ARGV[0];
$htmlfile = $ARGV[1];
require($bindir.'/getHTTP.pl');
##############################
```

```
#現在時刻の取得
($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) = localtime(time);
$year += 1900;
$mon += 1;
$tommorow = $yday + 1;
##############################
#データ URL の整形
$hostname = "www.swift.psu.edu";
$datafile = sprintf("operations/PPST/%4d%03d/PPST_%4d%03d0000_%4d%03d0000_00.html",$year, $yday
$data_URL = "http://" . $hostname . "/" . $datafile;
##############################
#HTTP でデータを取得し、ファイルへ書き出す
#$datadir = "/home/masuno/widget/Swift/dat";
#$htmlfile = "Swift_data.txt.html";

($http_response, $errorMessage) = &getHTTP('URL'      => ${data_URL});
open(OUTPUT, ">$htmlfile") or die "cannot open $htmlfile, $!";
print OUTPUT $http_response ;
close(OUTPUT);
_END_
```

A.5 Swiftcat3.pl

```
#!/usr/bin/perl
################################
use Time::Local;
use Math::Trig;
use Astro::SLA;
my $R2D = 180.0 / pi;
my $D2R = pi / 180.0;
################################
##ファイルから各行の読み込み
##HTML タグを除去して配列に代入
while($line = <>){
    $line =~ s/<.*?>/ /g;
    @data = split(/\s{2,}/, $line);
############################
#観測データの整形
    if($data[1] =~ /20[0-9]{2}\-/){
@start_data = split(/\-/, $data[1]);
@end_data = split(/\-/, $data[2]);
#[0]:西暦 [1]:経過日数 [2]:24 時 hh/mm/ss
@start_time_HMS = split(/:/, $start_data[2]);
@end_time_HMS = split(/:/, $end_data[2]);
#[0]:HH [1]:MM [2]:SS
######################
# yday -> mday
# datetime -> mjd
#starttime
&calc_monday(@start_data,@start_time_HMS);
$start_date = sprintf("%4d/%02d/%02d",$year+1900, $mon+1, $mday);
$start_time = sprintf("%02d:%02d:%02d",$hour,$sec,$min);
$start_mjd  = &datetime_to_mjd($start_date, $start_time);
#endtime
```

```
&calc_monday(@end_data,@end_time_HMS);

$end_date = sprintf("%4d/%02d/%02d",$year+1900, $mon+1, $mday);
$end_time = sprintf("%02d:%02d:%02d",$hour,$sec,$min);
$end_mjd  = &datetime_to_mjd($start_date, $start_time);

#RA,Dec

$RA       = sprintf("%.6f",$data[4]*$D2R);
$Dec      = sprintf("%.6f",$data[5]*$D2R);
my $RA_string  = &RA_to_string($RA);
my $Dec_string = &Dec_to_string($Dec);

print("$start_date $start_time $start_mjd $end_date $end_time $end_mjd $RA_string $Dec_string
    }
}
#    printf("%s\n", &time_stamp());

######################
######################
#経過日数から月日を計算する。
#timelocal() 関数で当該年 1 月 1 日の time を計算

sub calc_monday{
    ($year, $yday, $time_hms, $HH, $MM, $SS) = @_ ;
    $mon = 0;
    $mday = 1;
    $hour = 0;
    $min   = 0;
    $sec   = 0;
    $time = timelocal($sec, $min, $hour, $mday, $mon, $year);

#1 月 1 日 0 時 0 分 0 秒からの経過日時を秒に換算して $time に足す。
#UT -> JST  9 時間を足す。
#改めて localtime() 関数を用いる。 -> ssmmhhDDMMYYYY

    $time += $yday*86400 + $HH*3600 + $MM*60 + $SS ;
    $time += 9*3600;
    ($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) = localtime($time);
}

##########################################################################
##########################################################################
sub datetime_to_mjd {
    my ($date, $time) = @_;

    my ($yy, $mm, $dd) = split(/\//, $date);
    my ($HH, $MM, $SS) = split(/:/, $time);

    slaCaldj($yy, $mm, $dd, my $mjd, my $status);
    if ($status != 0){
print "MJD calculation error\n";
exit (1);
    }

    slaDtf2d($HH, $MM, $SS, my $mjd_frac, $status);

    if ($status != 0){
print "MJD calculation error\n";
exit (1);
    }

    return $mjd+$mjd_frac;
}

##########################################################################

sub mjd_to_datetime {
    my $mjd = shift;
    my @ihmsf= ();

    slaDjcl($mjd, my $iy, my $im, my $id, my $frac, my $status);
    slaCd2tf(0, $frac, my $sign, @ihmsf);

    my $result = sprintf("%04d/%02d/%02d %02d:%02d:%02d",
```

```
 $iy, $im, $id,
 $ihmsf[0], $ihmsf[1], $ihmsf[2]);
    return $result;
}
##########################################################################
sub Dec_to_string {
    my $Dec = shift;
    my @idmsf = ();
    &slaDr2af(2, $Dec ,my $sign, \@idmsf);
    my $result = sprintf("%s%02d:%02d:%02d",
 $sign, $idmsf[0], $idmsf[1], $idmsf[2]);
    return $result;
}
##########################################################################
sub RA_to_string {
    my $RA = shift;
    my @ihmsf = ();
    &slaDr2tf(2, $RA, my $sign2, \@ihmsf);
    my $result = sprintf("%02d:%02d:%02d",
 $ihmsf[0], $ihmsf[1], $ihmsf[2]);
    return $result;
}
__END__
```

A.6 weather.csh

```
#!/bin/csh -f
set bindir = /home/masuno/widget/weather
set dataurl = http://www.jma.go.jp/jp/amedas_h/today-48531.html
set datafile = weather.txt
cd ${bindir}
/usr/bin/w3m -dump ${dataurl} > ${datafile}
${bindir}/weatherhtmlcat.pl ${datafile}
```

A.7 weatherhtmlcat.pl

```
#!/usr/bin/perl
    $status = 0;
    @a = ();

while($line = <>){
    @str = split(/\s+/, $line);
#    print("$str[0]\n");
    #print $line;
 #   print scalar(@str);
    if(scalar(@str) == 7 || scalar(@str) == 8){
@a = @str;
$status = 1;
# print("@a\n");
    }
    if($status == 1 && scalar(@str) == 2){
```

```
#   if($status == 1){
print "降雪量:$a[7]cm\n";
last;
    }
#    for($i = 0; $i < scalar(@str); $i++){
# if($str[$i] =~ /長野県/){
#
#     if($str[3] > 20){
# print("$str[0] $str[1] $str[2]\n");
#     }
# }
}
```

A.8 hoge2.c

```
\baselineskip=0.1in
\begin{verbatim}

#include "common.h"
#include "slalib.h"
#include "njp_control.h"
#include <math.h>
/*********************************************************/
/* Swift following judgement                             */
/*********************************************************/
int read_swift(RA, Dec)
    double *RA, *Dec;
{
  double el_min,HA_swift_max,HA_swift_min;
  double el,HA;
  int point_status[2];
/*********************************************************/
  FILE *fp;
  char info_file[] = "/home/masuno/widget/njp/log/Swift_data.txt";
  char string[256];
  char time_string[256];
  double mjd, mjd_frac;
  double start_mjd, end_mjd;
  char start_date[256], start_time[256], end_date[256], end_time[256];
  char RA_string[256], Dec_string[256];
  double RA_VALUE, DEC_VALUE;
  double ra_value[2], dec_value[2]; /* 初期化必要 */
  int j = 0;
  char hogera[256];
  int HIPERION;
  if((fp = fopen(info_file, "r")) == NULL){
    message(LOGFILE, "cannot open Swift Information File");
    return -1;
  }
/*********************************************************/

  calc_mjd(&mjd, &mjd_frac);
  calc_mjd_to_cal(mjd+mjd_frac, time_string);
  //  fprintf(fp_log_2, "%s %lf %lf\n",time_string, mjd, mjd_frac);
while((fgets(string, 256 - 1, fp)) != NULL){
  sscanf(string, "%s %s %lf %s %s %lf %lf %lf RA:%s Dec:%s\n", \
         start_date, start_time, &(start_mjd), end_date, end_time,
         &(end_mjd), &(RA_VALUE), &(DEC_VALUE), RA_string, Dec_string);
```

```
  //  printf("%s %s %lf %s %s %lf %lf %lf\n", start_date, start_time, start_mjd, end_date, end

  if (j==1){
    ra_value[1]  = RA_VALUE;
    dec_value[1] = DEC_VALUE;
    point_status[1] = swift_point_status(&ra_value[1],&dec_value[1]);
    break;
  }
  if ((start_mjd <= mjd+mjd_frac) && (end_mjd >= mjd+mjd_frac)){
    ra_value[0]  = RA_VALUE;
    dec_value[0] = DEC_VALUE;
    j = 1;
    point_status[0] = swift_point_status(&ra_value[0],&dec_value[0]);
  }
}

/*********************************************************/
  *RA = 0.0;
  *Dec = 0.0;

  sprintf(hogera,"%d%d", point_status[0], point_status[1]);
  //  fprintf(fp_log_2,"\nSwift point status: %s\n",hogera);
  //  fprintf(fp_log_2,"current ra:%lf dec:%lf\n",ra_value[0], dec_value[0]);
  //  fprintf(fp_log_2,"next    ra:%lf dec:%lf\n\n",ra_value[1], dec_value[1]);
  HIPERION = atoi(hogera);
  switch(HIPERION){
  case 10:
  case 11:    /*** go to current Swift point ***/
       *RA  = ra_value[0];
       *Dec = dec_value[0];
       return 0; break;
  case 01:    /*** go to next Swift point ***/
       *RA  = ra_value[1];
       *Dec = dec_value[1];
       return 1; break;
  case  00:
  default   :    /*** go to zenith ***/
       return -1; break;
  }
}
/*********************************************************/
/*********************************************************/
int swift_point_status(RA,Dec)
     double *RA, *Dec;
{
  double HA_swift_max = 330;
  double HA_swift_min = 50;
  double az,el,el_min,HA;
  el  = calc_AzEl(*RA, *Dec, "el")*_R2D;
  az  = calc_AzEl(*RA, *Dec, "az")*_R2D;
  el_min = calc_skyline(&az);
  HA  = calc_aHA(*RA)*_R2D;
  //  printf("el:%f az:%f el_min:%f HA:%f\n",el,az,el_min,HA);
  /* HA_swift_min/max は global にもらっておく (main 内) */
  if((el < el_min) || (HA > HA_swift_max) || (HA < HA_swift_min)){
   return  0;   }
  else{
   return  1;   }
}
/*********************************************************/
```

A.9 hogemain.c

```
/*************************************************************************/
/*                                                                       */
/*  WIDGET --- WIDe-field telescope for GRB Early Timing                 */
/*  TAKAHASHI SEISAKUSHO Ltd. NJP-Temma2 control program                 */
/*                                                                       */
/*  Written by Fumihiko Usui - 2004/06/13 Ver 1.0                        */
/*  Rearranged by FU         - 2005/05/05 Ver.2.0                        */
/*  Rearranged by Keiichi ABE- 2005/10/01 Ver.3.0                        */
/*  Rearranged by K.Masuno   - 2006/12/15 Ver.4.1                        */
/*  Rewrited   by K.Masuno   - 2007/01/29 Ver.5.0  for WIDGET2           */
/*                                                                       */
/*************************************************************************/

#include "common.h"
#include "njp_control.h"

/*************************************************************************/
/* Operation Limit */
/*************************************************************************/
//#define el_min 15
#define HA_min 30
#define HA_max 330
//#define HA_swift_min 50
//#define HA_swift_max 330

/*************************************************************************/
int main(int argc, char *argv[])
{
  char string[BUFFER_SIZE];
  char time_string[BUFFER_SIZE];
  char RA_string[BUFFER_SIZE], Dec_string[BUFFER_SIZE];
  double RA, Dec, HETE_RA_old, HETE_Dec_old, swift_RA_old, swift_Dec_old;
  double az, el, az_old, el_old, el_min;
  double HA;

  time_t stime;
  struct tm *sstime;

  double mjd, mjd_frac;

  double sunset_mjd, dusk_mjd, dawn_mjd, sunrise_mjd;
  double offset_time;

  int HETE_status = -1, Swift_status = -1, status = 0;
  // Swift_status = 0:now_pointing, 1:next_pointing, -1:invalid
  // status = 0:home position, 1:Swift 2:zenith -1:invalid

  sprintf(LOGFILE,  "log.txt");
  sprintf(STATFILE, "/home/widget/ops/config/MOUNT_STAT");

  /***********************************************************************/
  /* calculate today's schedule */
  /***********************************************************************/

  message(LOGFILE, "===== operation schedule =====");

  calc_mjd(&mjd, &mjd_frac);
  calc_mjd_to_cal(mjd+mjd_frac, time_string);
  sprintf(string,"now:     %s (mjd=%lf)", time_string, mjd+mjd_frac);
  message(LOGFILE, string);

  solar_schedule_time(&sunset_mjd, &dusk_mjd, &dawn_mjd, &sunrise_mjd);

  calc_mjd_to_cal(sunset_mjd, time_string);
  sprintf(string,"sunset:  %s (mjd=%lf)", time_string, sunset_mjd);
  message(LOGFILE, string);

  calc_mjd_to_cal(dusk_mjd, time_string);
  sprintf(string,"dusk:    %s (mjd=%lf)", time_string, dusk_mjd);
  message(LOGFILE, string);

  calc_mjd_to_cal(dawn_mjd, time_string);
  sprintf(string,"dawn:    %s (mjd=%lf)", time_string, dawn_mjd);
```

```
message(LOGFILE, string);

calc_mjd_to_cal(sunrise_mjd, time_string);
sprintf(string,"sunrise: %s (mjd=%lf)", time_string, sunrise_mjd);
message(LOGFILE, string);

offset_time = 60.0*5/86400.0;
sprintf(string,"offset time: %lf", offset_time);
message(LOGFILE, string);

//  exit(0);
/*************************************************************************/

message(LOGFILE, "===== NJP CONTROL start =====");

/*************************************************************************/
/***** Device initialize                                            *****/
/*************************************************************************/

message(LOGFILE, "===== device initialize =====");

open_NJP_device();
set_flags();

//  read_version();
send_obs_latitude();
read_obs_latitude();

read_status(string);
read_standby_status();

read_position(&RA, &Dec);
display_position(RA, Dec);

send_standby_off();

/*************************************************************************/
/**** set scope to home position                                      ****/
/*************************************************************************/

message(LOGFILE, "===== set to home position =====");
operation_status(STATFILE, "home");

send_LST();
azel_go_to(90.0, -10.0);

sleep(3);

/** sleep until sunset **/
  send_standby_on();

  while(1){
    operation_status(STATFILE, "home");

    message(LOGFILE, "standing-by for sunset");
    calc_mjd(&mjd, &mjd_frac);
    if ((mjd+mjd_frac) > (sunset_mjd + offset_time)) break;

    sleep(30);
  }

send_standby_off();

/*************************************************************************/
/*************************************************************************/
while(1){
  read_position(&RA, &Dec);
  el = calc_AzEl(RA, Dec, "el")*_R2D;
  az = calc_AzEl(RA, Dec, "az")*_R2D;
  el_min = calc_skyline(&az);

  calc_mjd(&mjd, &mjd_frac);

    /*********************************/
    /*****    check sunrise     ******/
    /*********************************/

      if ((mjd+mjd_frac) > (sunrise_mjd - offset_time)){
```

```
          message(LOGFILE, "sunrise");
          break;
          }

        else{

      Swift_status = read_swift(&RA, &Dec);

      /* Swift_status checking paragraph */

      sprintf(string, "Swift_status:%d",Swift_status);
      message(LOGFILE, string);
      sprintf(string, "latest Swift pointing RA:%lf Dec:%lf",RA ,Dec);
      message(LOGFILE, string);

      /*********************************/
      /***** check tracking point ******/
      /*********************************/

      if (RA != swift_RA_old || Dec != swift_Dec_old){
message(LOGFILE, "Swift pointing is old");
status = -1;
swift_RA_old = RA;
swift_Dec_old = Dec;
      }

      /*********************************/
      /***** main control routine  *****/
      /*********************************/

      switch(status){

/******** home position or *********/
/******** invalid position *********/
      case 0:
      case -1:

if( Swift_status == 0){
  message(LOGFILE, "go to current Swift Pointing");
  operation_status(STATFILE, "slew");
  send_LST();
  send_standby_off();
  go_to(RA,Dec);
  status = 1;       break;
}

if( Swift_status == 1){
  message(LOGFILE, "go to next Swift Pointing");
  operation_status(STATFILE, "slew");
  send_LST();
  send_standby_off();
  go_to(RA,Dec);
  status = 1;       break;
}

if( Swift_status == -1){
  message(LOGFILE, "go to zenith");
  operation_status(STATFILE, "slew");
  read_position(&RA, &Dec);
  az = calc_AzEl(RA, Dec, "az")*_R2D;
  send_standby_off();
  azel_go_to(az,90.0);
  send_standby_on();
  status = 2;       break;
}

/******** pointing at Swift *********/
      case 1:

if( Swift_status == 0){
  message(LOGFILE, "tracking at Swift Pointing");
  operation_status(STATFILE, "swft");
  send_standby_off();
  status = 1;       break;
```

```
}
if( Swift_status == 1){
  message(LOGFILE, "preparing for next Swift Pointing");
  operation_status(STATFILE, "slew");
  send_standby_off();
  status = 1;       break;
}
if( Swift_status == -1){
  message(LOGFILE, "POINTING HAZARD:Swift tracking overhanged");
  operation_status(STATFILE, "othr");
  send_standby_on();
  status = -1;       break;
}
/******** pointing at zenith *********/
      case 2:
if( Swift_status == 0){
  message(LOGFILE, "go to current Swift Pointing");
  operation_status(STATFILE, "slew");
  send_LST();
  send_standby_off();
  go_to(RA,Dec);
  status = 1;       break;
}
if( Swift_status == 1){
  message(LOGFILE, "go to next Swift Pointing");
  operation_status(STATFILE, "slew");
  send_LST();
  send_standby_off();
  go_to(RA,Dec);
  status = 1;       break;
}
if( Swift_status == -1){
  message(LOGFILE, "standby at zenith");
  operation_status(STATFILE, "zeni");
  send_standby_on();
  status = 2;       break;
}
/******** other case *********/
      default:
message(LOGFILE, "STATUS ERROR:impossible status");
operation_status(STATFILE, "othr");
send_standby_on();
status = -1;       break;

      }
      read_position(&RA, &Dec);
      display_position(RA, Dec);
      sleep(30);
  }
    /*************************************************************************/
    /**** set scope to home position                                      ****/
    /*************************************************************************/
    send_standby_off();
    message(LOGFILE, "===== set to home position =====");
    operation_status(STATFILE, "slew");
    send_LST();
    azel_go_to(90.0,-10.0);
    sleep(3);
```

```
  /*************************************************************************/
  /**** set standby mode                                                ****/
  /*************************************************************************/

  read_position(&RA, &Dec);
  display_position(RA, Dec);

  send_standby_on();

  operation_status(STATFILE, "home");

  /*************************************************************************/
  /**** device close                                                    ****/
  /*************************************************************************/

  message(LOGFILE, "===== device close =====");

  close_NJP_device();
  message(LOGFILE, "NJP CONTROL end");
}}
```