

情報理工学特別研究報告書

題目

全天を分割して撮影した高画質な天体写真から
360度パノラマ画像を合成する研究

学生証番号 2054272

氏名 松原 光汰

提出日 令和6年1月26日

指導教員 蚊野 浩

京都産業大学
情報理工学部

要約

Ricoh THETA や GoPro Max など, 360 度パノラマ画像を簡単に撮影できる機器が普及している. これに伴い, 誰でも簡単に 360 度の天体写真を撮影することが可能になった. しかし, 360 度カメラは 2 つの超広角のレンズで全ての方向を撮影するため, 望遠レンズや望遠鏡を用いて撮影した通常为天体写真と比べて画質が劣る. 本研究では, 通常为天体写真と同等の画質を保ちつつ, 360 度のパノラマ画像を生成することを目指す. 具体的には, アマチュア天体写真家の平川俊夫氏から提供された, 八ヶ岳とオーストラリアで撮影した合計 1010 枚の天体写真を用いて, 360 度の天体写真を生成する. ここで, 原画像と共にそれぞれの画像がどの場所でどの方角に向けて撮影したものであるかを示したデータが提供されており, それも併せて利用した. 画像の変換には C++17 と OpenCV を用いた.

プログラムを作成し実行した結果, デジタル一眼カメラで撮影された天体写真から, 正距円筒図法の 360 度の天体写真を生成することができた. 当初は, 平面射影変換を用いて原画像を台形状に変形した後, 画像を貼り合わせて合成した. しかし, 合成後の 360 度の天体写真を確認したところ, 画像間のずれが目立った. そこで, 天球を円筒面に投影する変換を行うように改良し, より高精度の 360 度の天体写真を生成した. また, 開発のイテレーションを高めるために, すべての処理を同時並行で実行できるように工夫し, 処理速度を向上させた.

しかし, いくつかの課題が残った. 撮影された原画像の赤経・赤緯は提供されたデータから明らかになっているものの, 人間が手作業で撮影したため, 実際の写真は提供されたデータと比べて多少角度がずれている. この角度のずれによる位置合わせ精度の低下が目立った. また, 北極点と南極点を撮影した天頂部分の画像は, 他の画像と違い, 1 つの画像に全方角の星が記録されている. よって, これらの画像を 360 度画像に変換するには, 他の画像とは違う処理を行う必要があるが, その処理を作ることはできなかった.

目次

1 章 序論	．．． 1
2 章 ソフトウェアの作成に必要な情報	．．． 2
2.1 正距円筒図法	．．． 2
2.2 提供された画像	．．． 3
2.3 開発に使用したライブラリ	．．． 4
3 章 天体写真から 360 度パノラマ画像を合成するツールの開発	．．． 5
3.1 天体写真を全天画像に貼り合わせる方法	．．． 5
3.2 円筒面投影を台形変換で近似する場合の合成	．．． 7
3.3 天球を円筒面に投影する変換を用いた場合の合成	．．． 10
4 章 正距円筒画像の生成と比較	．．． 11
4.1 台形変換画像を順次上書きして貼り合わせた場合	．．． 11
4.2 台形変換画像から中央部を切り出し貼り合わせた場合	．．． 12
4.3 円筒面投影を行い, 中央部を切り出し貼り合わせた場合	．．． 14
5 章 結論	．．． 16
謝辞	．．． 17
付録	．．． 18

1 章 序論

有史以来, 人類は星々を高精度に記録する方法を模索してきた. 近代に入り望遠鏡や写真技術の進歩によって, 星々や銀河の高精度写真を撮影することが可能になった. 現代においては, スマートフォンやデジタル一眼カメラが普及したことによって, 誰もが手軽に星や銀河を撮影し, 記録することが可能になった. しかし, 広い天の川銀河や, 全天の星座同士の位置関係を 1 枚の写真に収めるためには, 広い空全体を撮影する必要がある, それはスマートフォンやデジタル一眼カメラでは難しい. しかし, 近年普及した, Ricoh THETA や GoPro Max などによって全天を一枚の写真に収めることが可能になった. これらの機器には 2 つの超広角レンズが備わっており, 同時に 2 枚の写真を撮影して合成することで, 360 度の写真を撮影することが可能である. しかし, この手法で撮影した画像は, デジタル一眼カメラで撮影した通常为天体写真と比べて, カメラの画素数に対して記録する範囲が広い. よって, 等級が低い星が映らない, 隣接する小さな 2 つの星が重なって別々の星であると認識できなくなるといったデメリットがある. また, 北天と南天の夜空を同時に撮影することはもちろんできない.

本研究では, デジタル一眼カメラで撮影した 1010 枚の天体写真を貼り合わせ, 1 枚の 360 度のパノラマ画像を生成することを目的としている. この手法により, デジタル一眼カメラの高い画質や分解能を保ちつつ, 天球全体を観察できる広い視野を実現することができる. そのために, C++17 と OpenCV を用いて複数の星の画像を合成するシステムを開発した. 原画像はアマチュア天体写真家の平川氏から提供された, 1010 枚の天体写真を用いた. これらの写真は, 八ヶ岳とオーストラリアで撮影されており, 天球全体を分割して写真に収めている. また, それぞれの写真を撮影した方角に関するデータも提供されており, それも併せて利用した.

以下, 2 章ではプログラム作成に必要な情報に関して, 3 章では天体写真から 360 度パノラマ画像を合成するツールの開発について, 4 章では実験結果について記述する. 5 章では結論について記述する.

2章 ソフトウェアの作成に必要な情報

2章では, 正距円筒図法の説明と, 提供された 1010 枚の天体写真とデータの説明, 開発に使用したライブラリの説明を記述する.

2.1 正距円筒図法

正距円筒図法は球体表面を平面に投影するための地図投影法の 1 つである. 今回はこの方式を使って, 天球をパノラマ画像に変換した. ここで言う天球とは, 地球や観察者を中心として広がる無限大の半径を持った球体のことである. 天体観測とは, 天球の表面に配置された星々を, 球体の中心から外側に向かって観察することと仮想的に考えることができる. なお, この天球上に配置された星々の位置を表すために, 赤道座標系がよく用いられるが, これは, 例えば経度方向に 60 度, 緯度方向に 80 度進んだ場所を, 赤経 4h0m0s 赤緯 5h20m0s と表現する手法である. しかし, この論文ではより直感的に星の位置を表すために, 特に断りがなければ赤経 60 度赤緯 80 度と度数法を用いて記述する.

画像が貼り付けられた球体から, 正距円筒図法に基づくパノラマ画像を作成する手順は以下の通りである. 最初に, 緯度の線に従って切り出した画像を平面に配置する. その例を図 1 に示す. ここで空いた隙間は横方向に引き延ばすことで埋める. 横に伸ばす倍率は緯度の角度 θ に対して $1/\cos \theta$ である. 正距円筒画像の縦方向は緯度に比率する. しかし, 横方向は, 画像の上部と下部に向かうにつれて引き伸ばされる. 正距円筒図法の拡大率を表した画像を図 2 に示す. なお, 球面全体を展開した正距円筒画像は横の長さが縦の 2 倍になる.

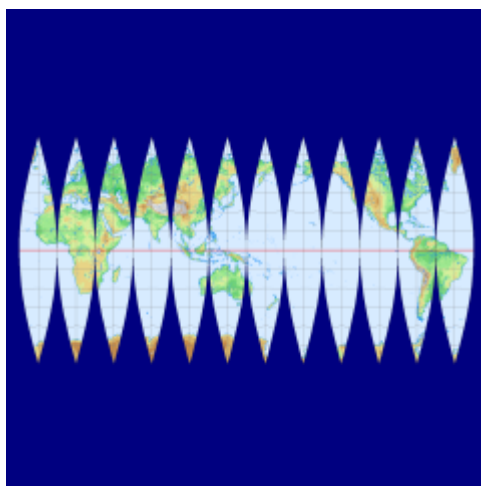


図 1 緯度の線に従って切り出した画像を平面に配置した例

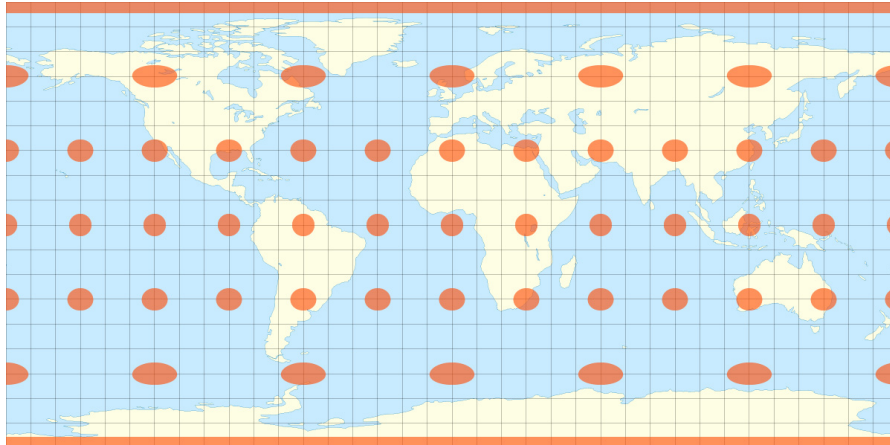


図2 正距円筒図法の拡大率を表した画像

2.2 提供された画像

本研究で使用する 1010 枚の天体写真は、アマチュア天体写真家の平川俊夫氏から提供された画像である。これらの写真は、長野県八ヶ岳とオーストラリアのノーザンテリトリー・アリスプリングスで撮影されたものである。撮影機材はニコン D810A のカメラとニコンの 105mm のレンズで、赤緯 0 度周りの全周を 80 分割、赤緯方向には 10 度刻みで撮影されている。なお、天球の極に向かうほど全周の分割数は小さくなる。例えば赤緯 60 度地点では 48 枚、赤緯 90 度地点では 1 枚のみ撮影されている。赤緯と画像の枚数の関係を示した表を表 1 に示す。

表 1 赤緯と画像の枚数の関係

赤緯	枚数	赤緯	枚数
+90度	1枚	-90度	1枚
+80度	24枚	-80度	24枚
+70度	36枚	-70度	36枚
+60度	48枚	-60度	48枚
+50度	60枚	-50度	60枚
+40度	64枚	-40度	64枚
+30度	72枚	-30度	72枚
+20度	80枚	-20度	80枚
+10度	80枚	-10度	80枚
+0度	80枚		

画素数は、赤経+90度と赤経-90度の2枚は 6158x4922 で、その他の画像は 3935x4922

である。全ての画像は、撮影した RAW 画像を AstroArts 社の StellaImage8 でダーク画像 + フラット画像を加えて処理したものである。なお、本研究では画像処理を高速化するために原画像を縮小した。縮小後の画素数は赤経+90 度と赤経-90 度の 2 枚は 626x500 で、その他の画像は 400x500 とした。これ以降、原画像について言及する時は、この縮小後の天体写真を指す。

隣り合う原画像はかなりの重なりを持っている。図 3 は赤緯 50 度, 赤経 318 度に向けて撮影した原画像で、図 4 はその隣の、赤緯 50 度, 赤経 324 度に向けて撮影した原画像である。赤い矢印はデネブを指している。図 3 で右端にあったデネブが図 4 では中心近くにあることから、原画像は互いに広い範囲で重なり合っていることがわかる。



図 3 赤緯 50 度, 赤経 318 度を向いた原画像 図 4 赤緯 50 度, 赤経 324 度を向いた原画像

2.3 開発に使用したライブラリ

開発には OpenCV と cxxopts という 2 つのライブラリを使用した。OpenCV は画像の分割処理やヒストグラムの計算, パターン認識などの画像に関する処理を実行できるライブラリである。cxxopts は C++ 向けの引数解析ライブラリである。1 つのヘッダファイルにすべての処理が記述されているので手軽に利用することができる。通常、このライブラリはアプリケーションを実行する際に指定するコマンドライン引数を解析するために使用する。しかし、今回のソフトウェアの開発においては、起動後に画像をロードする、変換処理を実行するといった命令や参照する画像の親フォルダの位置を変えるとといった変数を変える命令を実行するために利用した。具体的な利用箇所については付録に記

載する.

3章 天体写真から360度パノラマ画像を合成するツールの開発

デジタル一眼カメラで撮影した高品質の天体写真から360度のパノラマ画像を生成するためのソフトウェアを開発した. 開発環境は以下の通りである.

IDE : VisualStudio2022

プログラミング言語 : C++17

使用したライブラリ : OpenCV, cxxopts

3.1 天体写真を全天画像に貼り合わせる方法

図5を用いて, 天体写真を正距円筒画像面で貼り合わせるための変形について説明する. まず, 図で天球の周囲に配置した円筒は, 正距円筒図法で用いる平面を天の赤道で接するように巻きつけたものである. 天体写真は天球の一部を, 図のように平面上に透視投影したものである. 図5が示すように, この天体写真を, 一旦, 球面上に逆投影し, その球面画像を円筒面に投影する, という幾何学的な変換によって, 元の天体写真を正距円筒面で貼り合わせる画像に変換する. ここで, 元の天体写真の画角が小さい場合, 球面投影による変形はほとんど無視することができるので, 本研究では元の天体写真を, 直接, 円筒面に投影する場合について研究した.

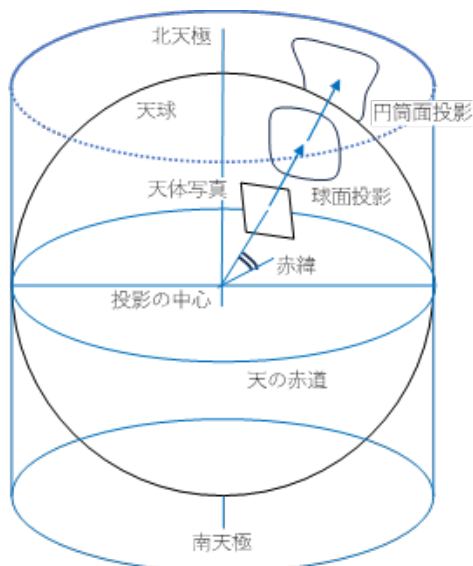


図5 360度パノラマ画像を合成するのに必要な幾何学的変換

天体写真を正距円筒画像で貼り合わせるためには, 個々の天体写真の赤緯・赤経が必要になる. その情報は表1から計算するのであるが, それらの位置と画像の変形の様子を図6に示す. 図6の最下行に赤緯0度の80枚の画像の並びを示す. 赤緯0度の場合,

円筒面に投影することによる変形はほとんど生じない. その一枚目の画像の中心が赤経 0 度であるとする. 横に 80 枚並べ、81 枚目が一枚目の画像になるので、赤緯 0 度における画像間の角度は 4 度である. 同様の計算で、赤緯 30 度の画像間角度は 5 度、赤緯 40 度は 5.625 度、赤緯 50 度は 6 度、赤緯 60 度は 7.5 度、赤緯 70 度は 10 度、赤緯 80 度は 15 度となる.

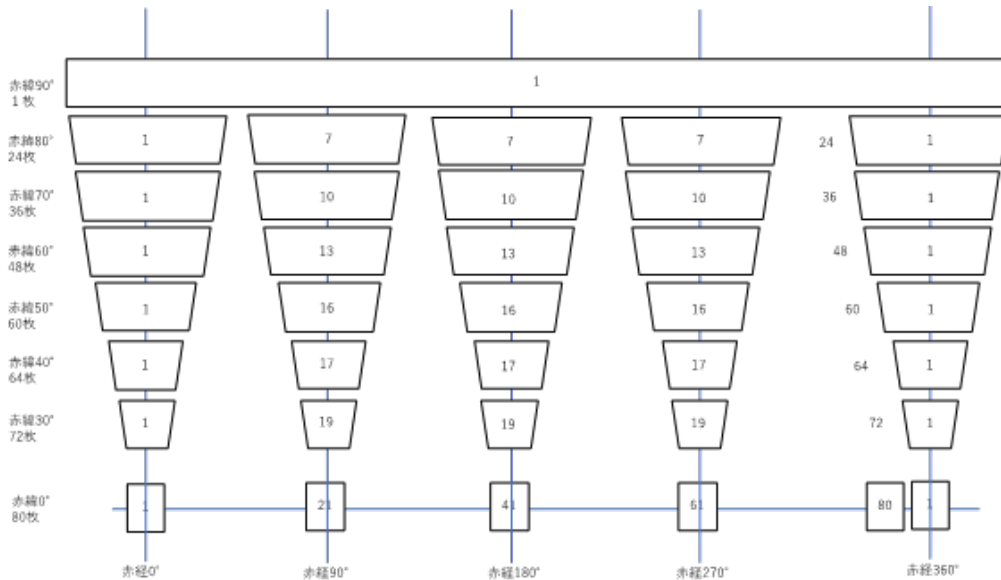


図 6 正距円筒画像の北半球部分で個々の天体写真の配置

本研究では、元の天体写真は平面画像であるが、これを天球面の近似であると考え、画像座標と赤道座標が相似であるとする. つまり、図 7 が天体写真画像であり、画像の中心を原点として水平方向を x 軸、垂直方向を y 軸にとり、 x 軸が赤経方向、 y 軸が赤緯方向、原点が赤経 α 度、赤緯 β 度、一画素あたりの角度が γ 度であるとき、画素位置 (x, y) の赤経は $\alpha + \gamma x$ 、赤緯は $\beta + \gamma y$ となる.

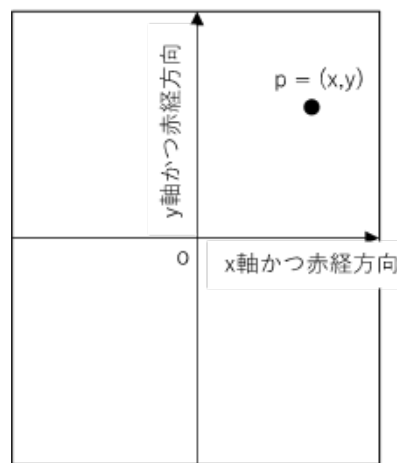


図 7 天体写真画像の画像座標系

3.2 円筒面投影を台形変換で近似する場合の合成

天体写真を円筒面に投影する変換は、画素位置の赤緯 θ に応じて、画像を赤経方向に $1/\cos \theta$ 倍に引き伸ばす処理である。この変換処理は少し複雑になるため、本研究では、これを台形変換で近似することを考えた。

台形変換は平面射影変換で実装することができる。平面射影変換は、任意の四角形 A を別の四角形 B に変形することができる変換である。今回は原画像の左上, 右上, 左下, 右下の 4 点によって構成される四角形を四角形 A とした。四角形 B を求めるには原画像の四隅に位置する画素が、天球のどの赤緯の部分でサンプリングしたのかを求める必要がある。原画像の中心の赤緯は原画像と共に提供されたデータに記載されている。その値から、原画像の四隅の画素の赤緯を求めるためには、画素位置が 1 ピクセル移動する毎に赤緯や赤経がどの程度変わるのかを求める必要がある。これ以降この値をピクセル毎角度と呼ぶ。

ピクセル毎角度を求めるために、図 8 と図 9 に示す原画像を用いる。なお、赤い矢印はデネブを指している。図 8 の原画像のデネブは原画像の中心から数えて 160 ピクセル下に位置している。図 9 の原画像のデネブは原画像の中心から数えて 205 ピクセル上に位置している。これらの関係を図 10 と図 11 に示す。なお、それぞれの図にある o の文字は、原画像の中心を指している。



図 8 赤緯 50 度, 赤経 324 度を向いた原画

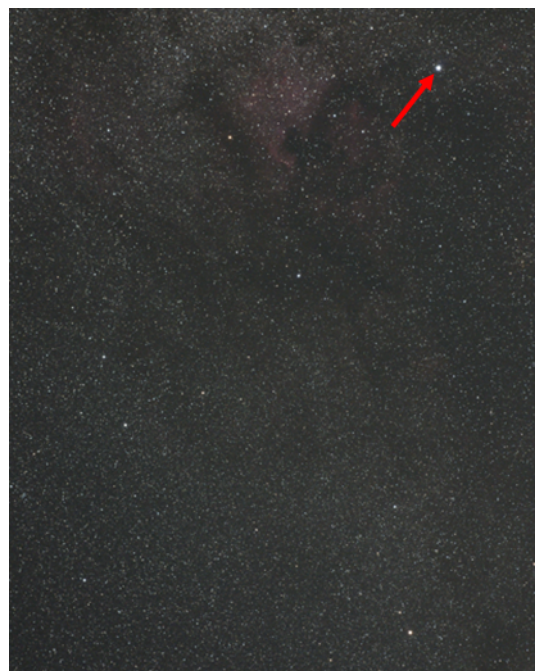


図 9 赤緯 40 度, 赤経 315 度を向いた原画像

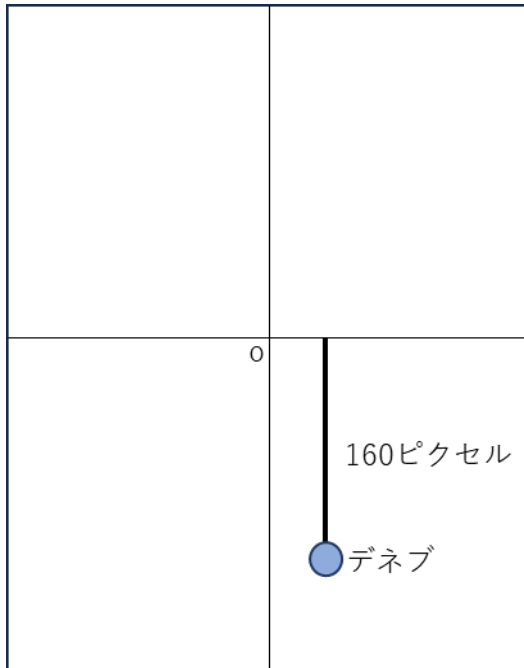


図 10 図 8 のデネブの位置

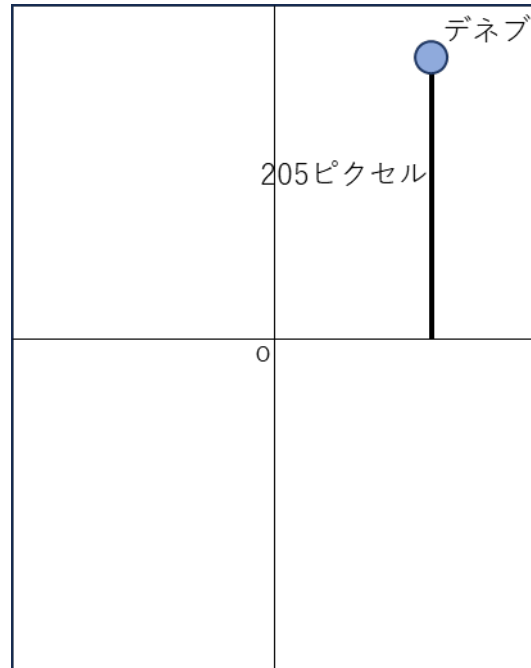


図 11 図 9 のデネブの位置

これらの情報から、縦に 160+205 画素移動すれば、赤緯が 10 度移動すると考えることができる。これを式に表したものを式(1)に示す。なお、 x はピクセル毎角度である。計算するとピクセル毎角度は 0.02739726027 度になった。この研究ではこの値をピクセル毎角度として使用する。

$$10 : 160 + 205 = x : 1$$

$$x = 0.02739726027 \quad \dots(1)$$

ピクセル毎角度を求めたことによって、平面射影変換を用いて原画像を台形に変形することができる。図 8 を用いて例を示す。初めに、原画像の最上部と最下部の赤緯を求める。図 8 の場合、中心の赤緯が 50 度であるから、式は式(2)と式(3)の通りである。なお、

式中の 500 という値は原画像の縦の画素数を示している。

$$50 + \left(\frac{500}{2}\right) \cdot 0.02739726027 \quad \dots(2)$$

$$50 - \left(\frac{500}{2}\right) \cdot 0.02739726027 \quad \dots(3)$$

求めた値を θ として、2.1 節で示した $1/\cos \theta$ の式に代入し、拡大率を得る。得られた

拡大率を用いて平面射影変換を実行する. 図8の原画像に対して実行した結果を図12に示す.

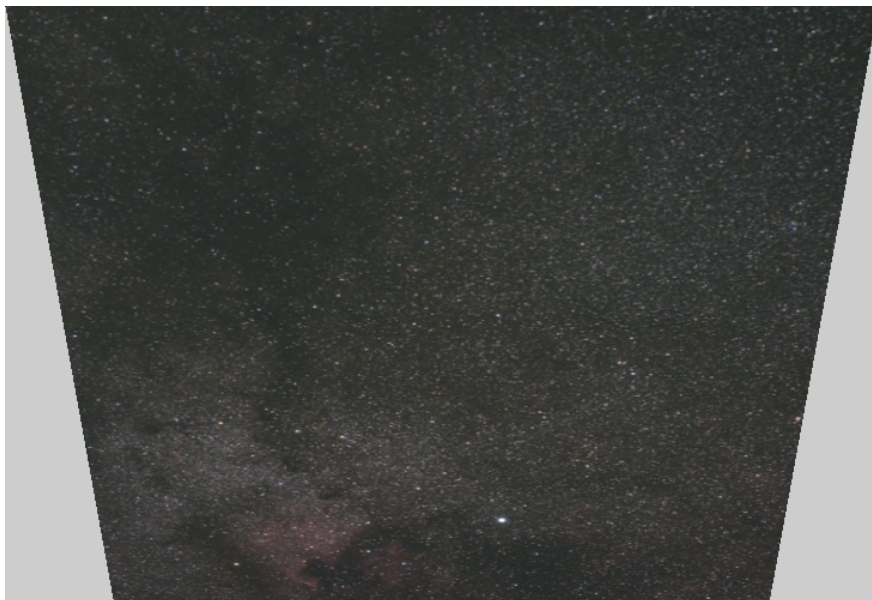


図12 図8に平面射影変換を行った結果

同じ変換をすべての画像に行い, 変換した画像を正距円筒画像の適切な位置に書き込んだ. 出力する画像の大きさは, 2.1節に示した通り, 横の長さを縦の2倍にする必要がある. 他に留意すべき点はないので, この時点では, 最終的な正距円筒画像の画素数を10000x5000にした. 後は原画像の赤緯と赤経と合う場所を, 出力する画像の赤緯と赤経から見つけて書き込めば良い. 図2に示した通り, 正距円筒画像の赤緯と赤経は, それぞれ垂直と水平になっているので位置合わせは容易に行える. 以降, この段階のソフトウェアをバージョン1と呼ぶ.

次に, 変形した原画像から必要な範囲を切り出し, それらを隙間なく縦横に並べて, 正距円筒画像を作る方針で進めた. 赤緯0度の画像を例として, 切り取る大きさについて説明する. 説明を簡略にするために横方向の画素数だけを考える. 正距円筒画像の赤経の角度を全て合わせると360度になる. 赤緯0度の原画像は合計で80枚なので, 360を80で割った値である赤経4.5度分を画像から切り取る必要がある. ピクセル毎角度を用いて赤経4.5度分の画素数を求めると, $4.5/0.02739726027$ で164.250000024となる. 小数点以下が含まれる画素数はありえないので, 小数点以下を繰り上げた165ピクセルが赤緯0度の原画像における横方向の切り取る大きさとなる. 同じように縦方向についても計算することで, 切り取る大きさを求めることができる.

正距円筒画像の画素数についても再度検討する. 表1に示した通り, 画像を横に並べる枚数は80, 72, 64, 60, 48, 36, 24の7通りある. これらの枚数の最小公倍数を求める

と 2880 となる. 正距円筒画像の横の画素数を 2880 の倍数にすれば, 画像を並べる際に整数倍で拡大するだけで配置できるので都合がいい. また, ピクセル毎角度を用いて赤経 360 度分の画素数を求めると, $360/0.02739726027$ が 13140.0000019 なので 13141 ピクセルとなる. よって, 2880 の倍数かつ, 13141 に近い値である 14400 ピクセルを正距円筒画像の横の画素数として選んだ. 縦の画素数に関しては, 正距円筒画像の縦の長さは横の $1/2$ 倍になるので, 7200 ピクセルとなる. 以降, この段階のソフトウェアをバージョン 2 と呼ぶ.

3.3 天球を円筒面に投影する変換を用いた場合の合成

より高精度な画像を得るために, 3.1 節で簡易的に実装した台形変換を, 円筒面に投影する変換に変更する. 具体的には原画像の全てのピクセルに対して赤緯を求め, 横方向に拡大するような変換を行う. このような処理は OpenCV の標準的な関数だけでは実行できない. よって, 以下の手順に従って画像を変形した. 400x500 の原画像を例とする. 初めに, 原画像を 500 個の 400x1 の画像に分解する. 次に, 分解した 400x1 の画像における赤緯を求める. ここで得た赤緯を θ として, 2.1 節に示した $1/\cos \theta$ の式に代入し拡大率を計算する. 最後にその拡大率を使って分解した 400x1 の画像を拡大してから, それぞれの画像を連結する. 図 8 にこの処理を行った結果を図 13 に示す. 図 13 を見ると, 図 8 に平面射影変換を行った図 12 とは違って完全な台形にはならず, 原画像のそれぞれのピクセルにおける赤緯の値によって, 原画像が滑らかに拡大されていることがわかる. 以降, この段階のソフトウェアをバージョン 3 と呼ぶ.



図 13 図 8 に天球を円筒面に投影する変換を実行した結果

4章 正距円筒画像の生成と比較

この章では3章で開発したソフトウェアの各バージョンを使って、正距円筒画像を生成した結果を示す。

4.1 台形変換画像を順次上書きして貼り合わせた場合

バージョン1のソフトウェアを実行し、出力された画像を図14に示す。このバージョンでは台形変換を行った後、必要な範囲を切り出さず、隣接する画像を上書きすることで、正距円筒画像を生成する。なお、出力する画像がかなり大きいので、この実験では北半球のみ処理を行った。図14を拡大した画像を図15と図16に示す。なお、赤い矢印はデネブを指している。

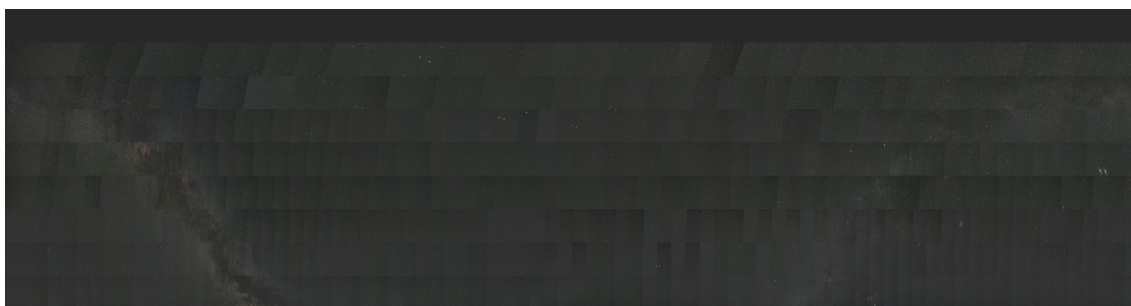


図14 出力された画像



図15 図14を拡大した画像



図 16 図 14 を拡大した画像

図 15 を見ると画像中心にある星雲が画像の切れ目によって分断されていることがわかる。また、図 16 ではアンドロメダ銀河が 2 つ見えてしまっている。また、図 15 と図 16 における、すべて画像の切れ目が斜めになっている。台形の斜めの部分が見えるという事は、原画像の中心部分以外の場所が使用されている事に他ならない。カメラで撮影した写真は中心から端に向かうにつれて、レンズの歪みが発生する。そのため、歪みを減らすためには画像の中心部分を使うことが望ましい。しかし、この手法の場合、画像同士の重なりにより、画像の端部が使用される。また、この手法では、最終的に出力する画像に書き込む原画像の順番を守る必要があった。例えば、左から A, B, C と隣り合わせで並んでいる同じ赤緯の原画像があったとする。原画像 B を書き込んだ後に A と C を書き込むと、B は左右から上書きされることになる。図 3 と図 4 で示した通り、隣り合う原画像はかなり重なった部分があるので、A と C を B の後に書き込むと B の画像は隠れてしまっ表示されない。そのために、赤緯や赤経の数値が小さい順に書き込む必要があった。よって、この書き込む処理はマルチスレッドにすることができず、その後のバージョンと比べてソフトウェアの動作は遅かった。

4.2 台形変換画像から中央部を切り出し貼り合わせた場合

バージョン 2 のソフトウェアを実行し、出力された画像を図 17 に示す。このバージョンでは台形変換を行った後、画像中心部の適切な範囲を長方形領域として切り出し、正距円筒画像上で隙間なく並べた。また、3 章に示した通り正距円筒画像の画素数についても再度検討を行い、バージョン 1 と比べてより都合がいい値になるよう計算した。なお、出力する画像がかなり大きいので、この実験では北半球のみ処理を行った。図 17 を

拡大した画像を図 18 と図 19 に示す. なお, 赤い矢印はデネブを指している. 図 18 を見ると画像中心にある星雲の上部に切れ目が見えている. しかし, 図 15 と比べると誤差は少なくなっている. また, 図 16 では 2 つに見えていたアンドロメダ銀河が, 図 19 では 1 つに修正された. しかし, アンドロメダ銀河の端が画像の切れ目によって分断されてしまった.

図 18 と図 19 における, すべて画像の切れ目が直角になっている. よって, 図 15 と図 16 に比べて, 原画像の中心部分が使用されている事だと考えた. なお, 必要な範囲を切り出して合成する手法に変更したことから, バージョン 1 で問題となったマルチスレッドで処理できない問題を解消することが出来た. よって, バージョン 1 と比べてソフトウェアの動作は早くなった.



図 17 出力された画像



図 18 図 17 を拡大した画像



図 19 図 17 を拡大した画像

4.3 円筒面投影を行い, 中央部を切り出し貼り合わせた場合

バージョン 3 のソフトウェアを実行し, 出力された画像を図 20 に示す. このバージョンでは, 円筒面に投影する変換を行った後, 画像中心部の適切な範囲を長方形領域として切り出し, 正距円筒画像上で隙間なく並べた. 図 20 を拡大した画像を図 21 と図 22 に示す. なお, 赤い矢印はデネブを指している. 図 21 を見ると画像中心にある星雲は, 図 18 と比べると誤差が少なくなっている. しかしずれが無いとは言い切れない. 星雲の上部は僅かではあるが, 画像の切れ目によって分断されている様子が確認できた. なお, 図 19 ではアンドロメダ銀河の端は分断されていたが, 図 22 ではその問題も解消された.

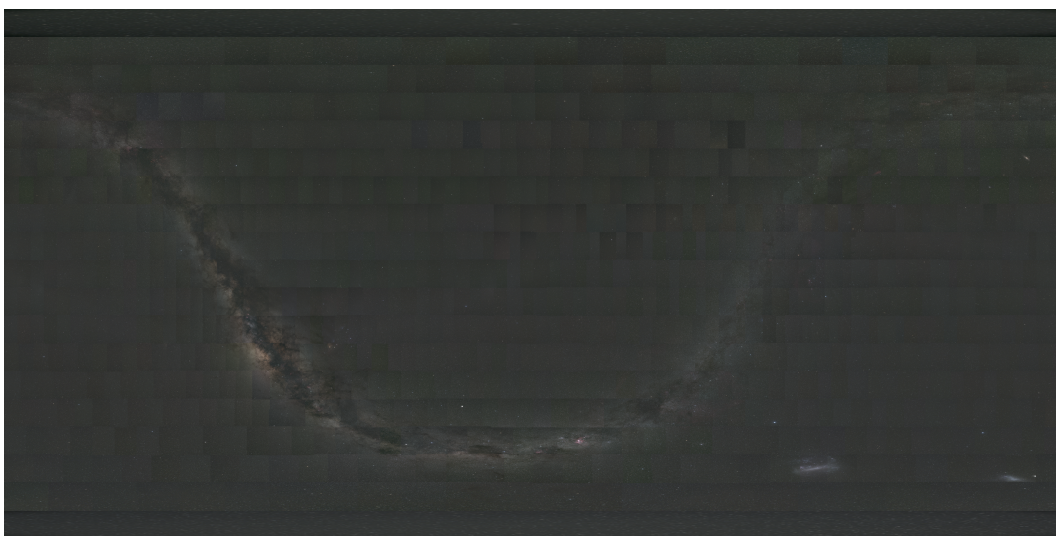


図 20 出力された画像

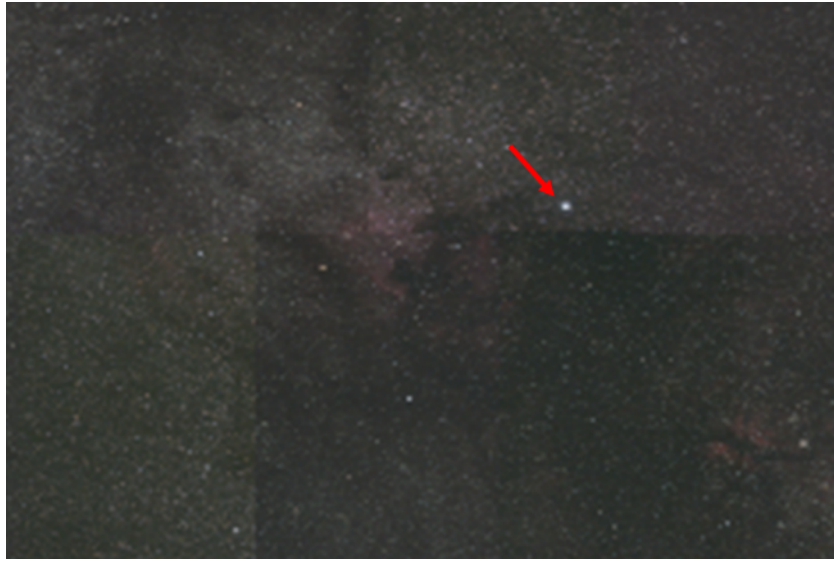


図 21 図 20 を拡大した画像



図 22 図 20 を拡大した画像

5章 結論

今回, 全天を分割して撮影した高画質な天体写真から 360 度パノラマ画像を合成する研究を行った. その結果, 開発したソフトウェアを使って, ある程度違和感が少ない正距円筒画像を生成することが出来た.

しかし, いくつかの課題が残った. まず, 出力画像に現れるずれを完全に修正することが出来なかった. 人間が手作業で撮影したことによって, 提供されたデータと実際に画像が向いている角度がずれている事が原因であったと考えられる. そのずれによって, 位置合わせ精度が低下し, 出力画像に現れたと推察した. この問題を解決するために, 画像がデータと比べてどの程度傾いているのかを調べるプログラムをパターンマッチングと平面射影変換を用いて開発したが, それを使って出力画像の精度を上げるプログラムを作成するのは間に合わなかった.

また, 北極点と南極点を撮影した天頂部分の画像をうまく変換することが出来なかった. これにより, すべての出力画像の上部と下部がうまく繋がっていない. 原因は, 北極点と南極点を撮影した天頂部分の画像が, 他の画像と違って, 1つの画像に全方向の星が記録されているからである. よって, これらの画像を 360 度画像に変換するには, 他の画像とは違う処理を行う必要があるが, その処理を作ることはできなかった. この問題を解決するためには, 北極点と南極点を撮影した天頂部分の画像を上下に分割し, それを横につなぎ合わせて 1つの画像に再構築する処理が必要なのではないかと考える.

謝辞

本論文を作成にあたり,丁寧な御指導を賜りました蚊野浩教授に感謝いたします.

ソフトウェア作成にあたり,天体写真を提供してくださった平川俊夫氏に感謝いたします.

付録 本研究で開発したプログラム

<https://github.com/NidoKota/StarCompositionSystem/>