Constellation Detection

Intermediate report

Yan Lyu; Lijiang Xu; Shuangyu Zhao

Abstract

Constellations, defined by clusters of bright stars within specific regions of the sky, are easily recognizable with spatial and temporal information. However, without such data, identifying these patterns against the vast expanse of the night sky or within a digital image is challenging. While modern technology, particularly smartphone applications, offers convenience in identifying constellations, there is still a significant demand for understanding and detecting constellations without digital assistance for survival and navigation, astronomy and research, cultural and historical significance, and reducing light pollution. Our research addresses this challenge, focusing on the identification of constellations from images without relying on any specific spatial or temporal information. This work holds significant value for places where digital devices fail, knowledge of the stars can be a literal lifesaver. The night sky can serve as a reliable compass.

In this project, there are six constellations (Ursa Major, Cassiopeia, Gemini, Perseus, Orion, Summer Triangle), which could be observed during this season, were chosen to classify. Various techniques were utilized in the acquisition of sky images, leading to the introduction of diverse noises and the creation of a comprehensive dataset. The dataset comprises four types of images: those captured with and without atmospheric interference, and photographs taken during both cloudy afternoons and clear night skies. For each type of image source, specific hyperparameters were selected for binary thresholding, median filtering, and dilation kernel processes to enhance star outlines. The project's core objective is to train both CNN and MLP models for constellation detection, benchmarking their performance against pre-existing CNN models. To diversify the dataset further, rotation and scaling were employed on each pre-processed image.

1. Data preprocessing

1.1. Dataset preparation requirement

To meet the specific data input requirements of different models, we developed two scripts for data preparation.

- CNN:

  The first script was designed for CNN data preparation, where the code directly read the preprocessed grayscale images with the same size in the NumPy array files and stored the constellation names.
  https://github.com/shaunzhao666/CSE881_project/blob/main/script/prepare_CNN.ipynb

  For another Pre-trained CNN model, the same images inputted in the CNN model were converted to RGB channels in the NumPy array files.
  https://github.com/shaunzhao666/CSE881_project/blob/main/script/prepare_pretrainedCNN.ipynb

- MLP:

For the second script, tailored to MLP data preparation, we took a different approach. In each image, the coordinates, and areas of the top 9 brightest stars were extracted. In cases where the total number of stars detected in the image was fewer than 9, the remaining coordinates and areas were set to 0, which ensured the MLP model received the samples with the same number of attributes. The links jumping to these two scripts are provided below.
https://github.com/shaunzhao666/CSE881_project/blob/main/script/prepare_MLP.ipynb

1.2. Data collection

The majority of the original constellation images are gathered from the Stellarium website (https://stellarium-web.org) by screenshotting with a size of 770 by 500. The six constellations (Ursa Major, Cassiopeia, Gemini, Perseus, Orion, Summer Triangle) were chosen for classification, which could be observed during this season as shown in Figure 1[1].

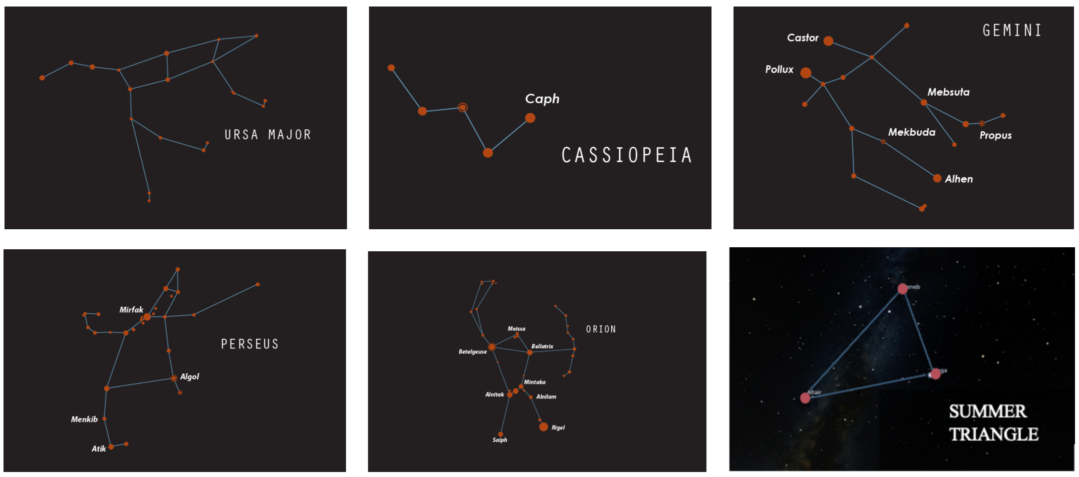

Figure 1. Selected six constellations

Leveraging the website's special effects capabilities, we configured the atmosphere and time setting to consider different sky conditions and deliberately cropped some stars out to include different scenarios of missing stars for each constellation. Besides, our group member Lijiang, who is the master of stars in the team, used her iPhone to take photos to enrich our dataset. But due to the cloudy weather recently, it still needs some time to complete. Therefore, for each constellation, there are 4 screenshots with atmosphere, 4 screenshots without atmosphere, and 4 photographs with or without clouds, as shown in Figure 2. For training the CNN and MLP model, we need to prepare numerous images with the same image size and fetch the location and area information of the stars from the images. To enrich our dataset, each image is scaled and rotated during the image preprocessing described in Part 1.3, and thus there are a total of 24 images generated for each screenshot and photo. After applying the affine transformation, 288 images are generated for each constellation. Therefore, the dataset includes 1728 images for all these 6 constellations under different conditions.

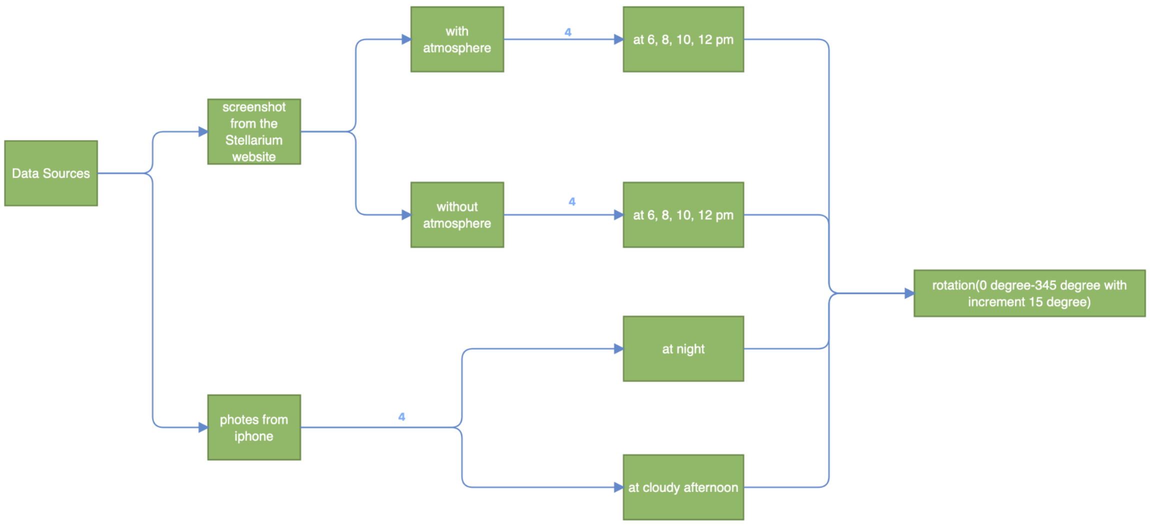

Figure 2. Schematic of the Dataset

1.3. Image preprocess

There are seven steps in preprocessing the collected data and different hyperparameters were chosen in binary threshold and medium filter for images from different gathering sources in order to extract clear star contours. The star extraction workflow is shown here.

- **Apply grayscale.**
  The grayscale was applied to the cropped images with a resolution of 770 by 500 since a 1-channel gray image was always easier to deal with than a 3-channel colored one.

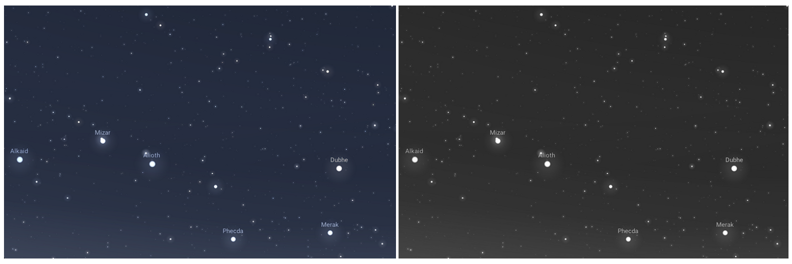

Figure 2. Ursa Major without atmosphere before (left) and after (right) applying grayscale

- **Experiment on binary thresholds, median filters, and dilatation kernels to produce contours.**
  To filter out all the non-star objects and reduce the possible noise, the grayscale images were preprocessed by three steps, binary thresholding, applying a median filter, and applying a dilatation kernel.
  1) Threshold
  Since we only require the stars to appear in the image, we binarize the image using an appropriate threshold to keep the majority of stars in the image. There were 3 thresholding values that were evaluated for images obtained from various sources. With a large threshold, more stars were removed. While if the threshold was too low, the comments on the stars were still there. The selected threshold was set conservatively to ensure the clarity of the stars. If some of the stars got removed, it is another way to increase our datasets by introducing patterns with missing stars.

2) Median filter
Two median filters (3 by 3 and 5 by 5) were compared to remove remaining local noises like labels. The selected median filter was set conservatively and made sure all label noises were removed.

3) Dilatation kernel
Finally, a dilatation kernel with size (2 by 2 or 3 by 3) was applied to emphasize the brightness of stars.

After applying these filters to the grayscale images, the contours of the stars were detected and all stars with zero area contours were removed.

The dataset had 4 categories about the source of images, which were images with or without atmosphere, and photos at night and in the afternoon with blue sky and clouds. Therefore, we should handle different scenarios using different thresholds, median filters, and dilatation kernels.

- Images without atmosphere (NAP in the image folder)
  The thresholds 190 were chosen from 150, 190, and 210. Then the two median filters and dilatation were compared. Finally, the thresholding value 190, 3 by 3 median filter, and 2 by 2 dilatation kernel were selected.

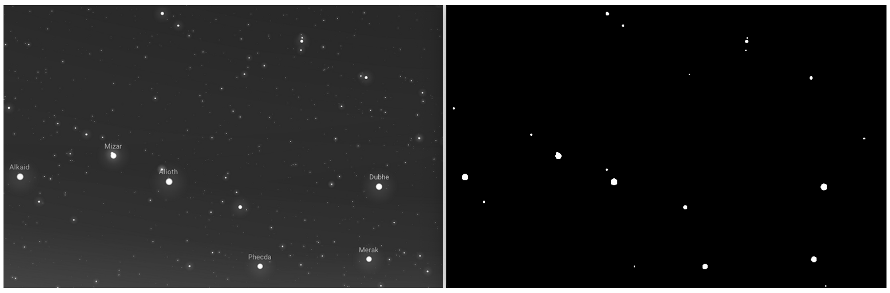

Figure 3. Grayscale Ursa Major (left) and preprocessed image with threshold 190, 3 by 3 median filter, and 2 by 2 dilatation (right).

- Images with atmosphere (AP in the image folder)
  The thresholds 90 were chosen from 90, 110, and 130. The lower thresholds than the ones with atmosphere resulted from the smaller star size and less noise in the original image. Then the two median filters and dilatation kernels were compared. Finally, the thresholding value 90, 3 by 3 median filter, and 3 by 3 dilatation kernel were selected. The increasing dilatation filter made the stars more prominent which would be beneficial for the following contour detection procedure.

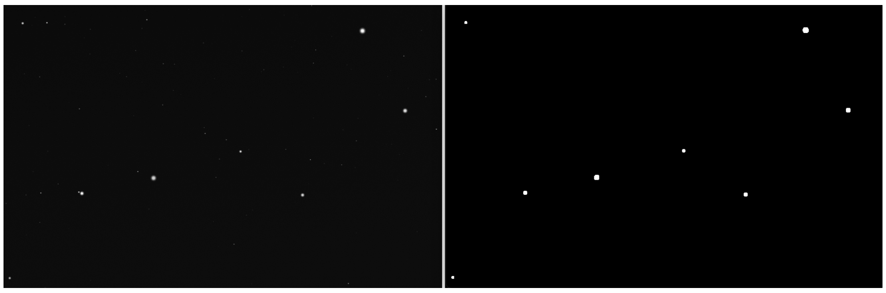

Figure 4. Grayscale Ursa Major (left) and preprocessed image with threshold 90, 3 by 3 median filter, and 3 by 3 dilatation (right).

There were some special cases in the category of images with atmosphere. Since the stars in Gemini, and Perseus are much dimmer, we removed the median filter to maintain the crucial stars and increased the threshold a little bit from 90 to 110. Besides, there are two labels of the two brightest stars in Orion. In order to remove the labels, the 3 by 3 median filter was applied before thresholding, then the threshold was increased to 130 followed by the 3 by 3 median filter to ensure the labels were removed.

- Photos taken at night

  The thresholds 110 were chosen from 90, 110, and 130, which was similar to the images with atmosphere. Then the two median filters and dilatation kernels were compared. Finally, the thresholding value 110, no median filter, and 2 by 2 dilatation kernel were selected. The median filter was removed to keep more stars in the sky. And the dilatation filter made the stars more obvious which would be beneficial for the following contour detection procedure.

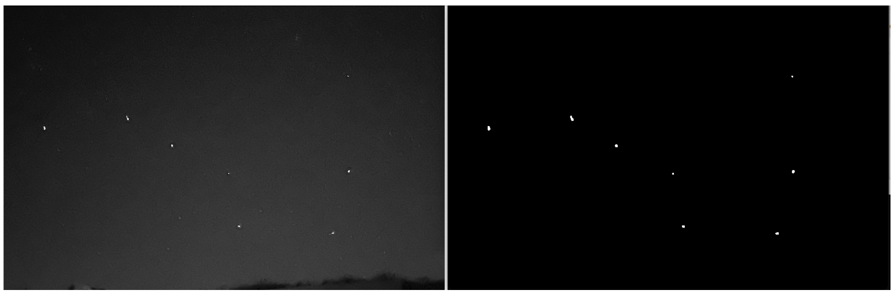
Figure 5. Grayscale Ursa Major photo (left) and preprocessed image with threshold 110, and 2 by 2 dilatation (right).

- Photos taken in the afternoon with blue sky and clouds

  The thresholds 190 were chosen from 150, 190, and 210, which was the same as the images without atmosphere due to more noise in the photo. Then the two median filters and dilatation kernels were compared. Finally, the thresholding value 190, no median filter, and 3 by 3 dilatation kernel were chosen. The thresholding is effective in removing clouds and the sky. Therefore, the median filter was removed to keep more stars in the sky. And the dilatation filter made the stars more prominent which would be beneficial for the following contour detection procedure.

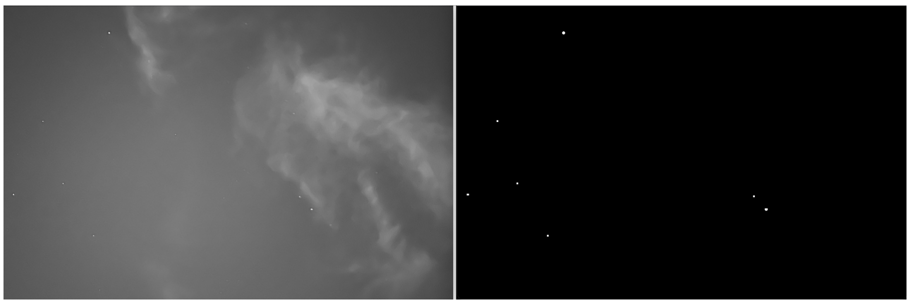

Figure 6. Grayscale Ursa Major photo (left) and preprocessed image with threshold 190, and 3 by 3 dilatation (right).

From Figure 3,4,5,6, we can see that only the stars with an area standing for its visibility magnitude were shown in the preprocessed contour images.

- **Find the star's location and area**

For rotating and scaling the stars in the next step, the location and area of the stars need to be derived in advance. As for the location and area of the stars, we can use image moments to calculate it.

The image moments summarize a shape-given image $I(x,\ y)$:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y)$$

Where $I(x,\ y)$ is the intensity of the pixel at location $(x,\ y)$. For a binary image, $I(x,\ y)$ can be either 0 (black) or 1 (white).

Therefore, the spatial moment M_{00} is the zeroth moment, which is equivalent to the area of the contour for a binary image:

$$M_{00} = \Sigma_x \Sigma_y I(x, y)$$

The spatial moments M_{10} and M_{01} are given by:

$$M_{10} = \Sigma_x \Sigma_y x \bullet I(x, y)$$

$$M_{01} = \Sigma_x \Sigma_y y \bullet I(x, y)$$

In simple terms, M_{10} gives the weighted sum of x-coordinates, and M_{01} gives the weighted sum of y-coordinates. When dividing them by M_{00} , the average x-coordinate and y-coordinate (weighted by pixel intensity) are calculated respectively, which is the centroid of the contour $(C_x,\ C_y)$. The formulas are given:

$$C_x = \frac{\Sigma_x \Sigma_y x \bullet I(x,y)}{\Sigma_x \Sigma_y I(x,y)} = \frac{M_{10}}{M_{00}}$$

$$C_y = \frac{\Sigma_x \Sigma_y y \bullet I(x,y)}{\Sigma_x \Sigma_y I(x,y)} = \frac{M_{10}}{M_{00}}$$

For a binary image, where each pixel in the contour has an intensity of 1, this essentially calculates the average x and y position of the contour, giving the centroid of the contour. After getting the centroid and area of these contours in a preprocessed image, these contours were ranked by area from large to small in decreasing order. If two stars have the same area, a small value of 0.0001 was added to the area of one of them to distinguish them.

- **Rotate and scale stars and draw an image with the desired output size.**

To generate multiple images from a single preprocessed one, we applied rotation and scaling to the star contours of the preprocessed image. The rotation spanned angles from 0 to 345 degrees in 15-degree increments, centered on the input image. For each rotation angle, contours were scaled using a distinct scale factor. This factor was determined by adjusting the bounding box of the rotated star contours to fit a fixed output image size of 256x256 pixels. To guarantee that all stars remained within the output image, we reduced the calculated scale factor by a safety margin of 0.1. Since area is proportional to the square of length, it was reduced by the square of the scale factor. To preserve the circular shape of the stars, we initially drew the rotated stars as figures before converting them into images. These images could be in grayscale for the CNN model or in color for the pre-trained CNN model.

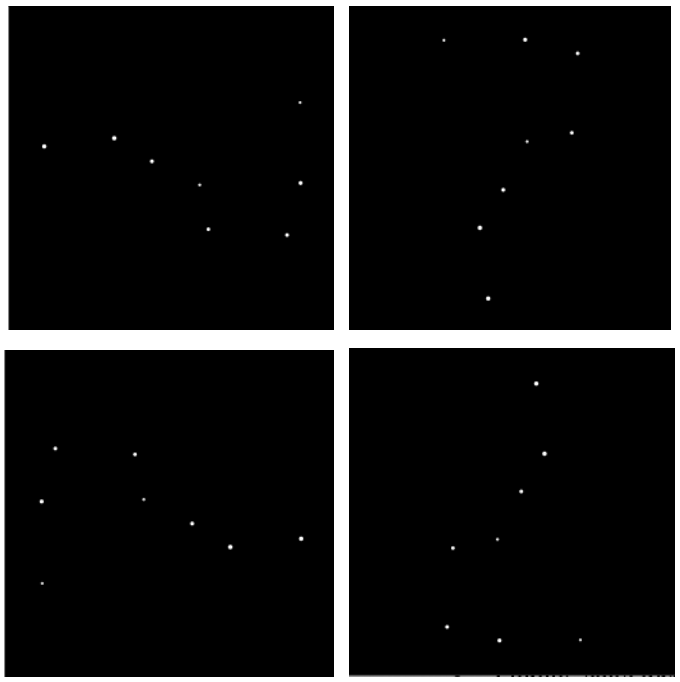
Figure 7. Preprocessed Ursa Major photo with rotation angle 0° (top left) 90° (top right), 180° (bottom left), 270° (bottom right).

- **Find the star's location and area for all the generated 24 images.**
  The coordinates and area of the stars of the 24 generated images were derived using the same image moments technique shown above.

2. Remaining tasks

The remaining work is about training multiple models to detect constellations. Three pre-trained CNN models are selected as baseline models. We will train our own MLP model and CNN model for constellation detection.

2.1. Pre-trained CNN models

Pre-trained CNN models are pre-trained by other image datasets. The corresponding functions in the Tensorflow package will be directly used.

- VGGnet
- ResNet
- GoogLeNet

2.2. MLP model

The multilayer perceptron is a supervised learning model using the backpropagation method during training. The major applications of MLP are pattern classification, recognition, prediction, and approximation. The Tensorflow package will be used to build the MLP model.

2.3. CNN model

The Convolutional Neural Network (CNN) is a standard model for image recognition that leverages linear algebra in its operations. The input data will be processed in grayscale images. The Tensorflow package will be used to build the CNN model.

3. Difficulty level

Moderate

Total level points: 4

	Level Points
Data Collection	1
Data preprocessing	2
Ground truth labels	0.5
Evaluation(compare against more than 2 baseline models)	0.5

4. Timeline

Time	Mission	Persons in Charge
09/21~09/26	Proposal	Lijiang Xu, Shuangyu Zhao
10/01~10/27	Data collection	Yan Lyu, Lijiang Xu
10/01~10/27	Python scripts for image preprocessing (image filtering and resizing, constellation rotation, star coordinates, and area extraction)	Lijiang Xu
10/01~10/27	Python scripts for data preparations (pre-trained CNN, MLP, and CNN)	Shuangyu Zhao

10/27~11/2	Intermediate report	Shuangyu Zhao, Lijiang Xu
11/2~12/1	MLP model	
11/2~12/1	Pre-trained CNN model	
11/2~12/1	CNN model	
Pending	Presentation	Lijiang Xu, Shuangyu Zhao, Yan Lyu
12/1~12/7	Final report	Lijiang Xu, Shuangyu Zhao, Yan Lyu

5. Reference

[1] The night sky – 88 constellations http://astronomyonline.org/Observation/Constellations.asp?Cate=Obse%20rvation&SubCate=MP07&SubCate2=MP0801

[2] Stellarium, an online planetarium, Matthew Gates, Barry Gerdes. https://stellarium-web.org/

6. Appendix

- Github link to our repository: https://github.com/shaunzhao666/CSE881_project/tree/main