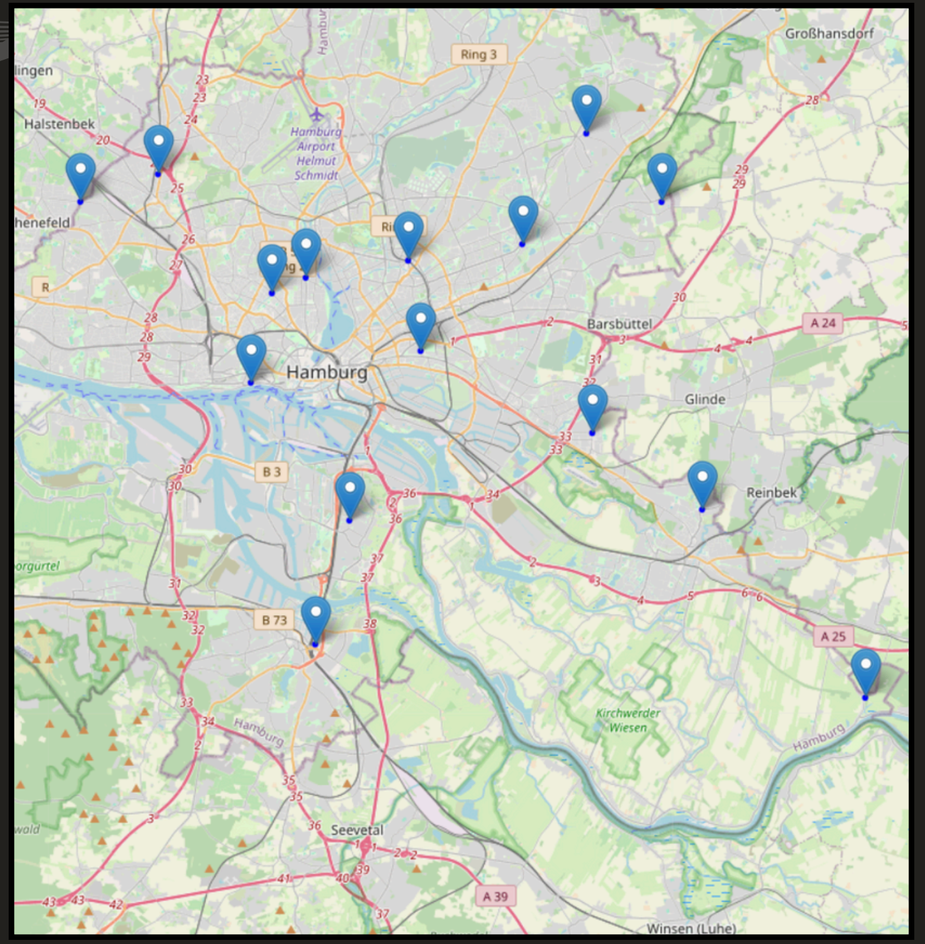
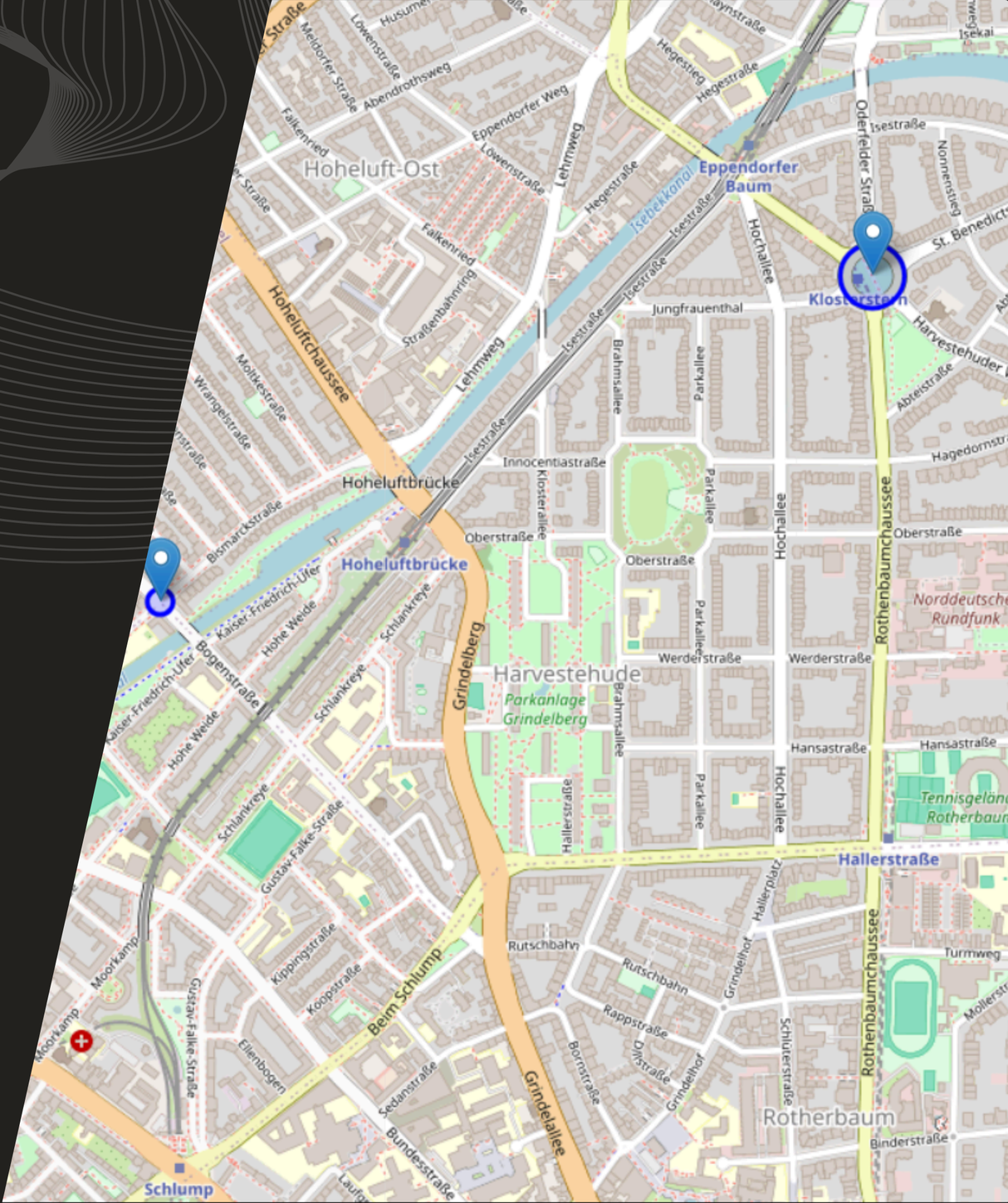
HERE
Cairo
Hackathon

GEO TENSORS
PROJECT



Overview

- About Us
- The Problem
- Ideas
- Tabular Approach
- CV Approach
- CV Second Approach



Who Are We?

Shehab Diab LinkedIn GitHub

Omar Elgammal LinkedIn GitHub

Zeyad Shahin LinkedIn GitHub

Youssef Mohamed LinkedIn GitHub

The Problem

Identify roundabouts using HERE Probe Data

Since first time we saw data

Tabular Approach

After cleaning the data and excluding impossible speeds during rotation, we could detect the roundabout within a certain radius by performing a logistic regression analysis to predict whether a data point falls within a specific region (circle) based on several features.

Computer Vision

We thought about how to use computer vision to detect roundabouts in the data by first plotting the roundabouts and then applying image processing techniques to them.

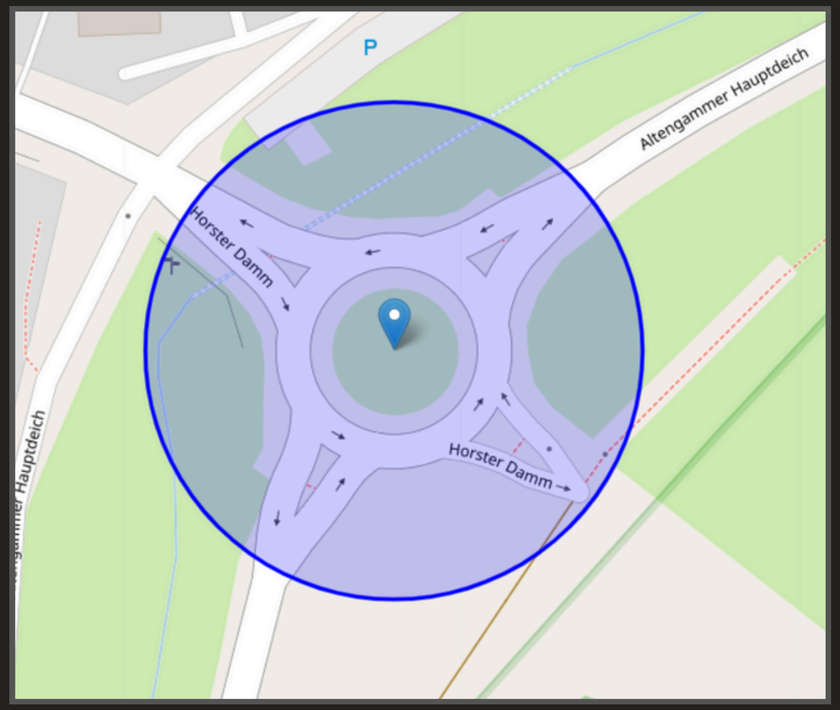
CV Another Approach

The data is chunked by longitude and latitude, then aggregated into a 500x500 grid. After removing outliers, applying Gaussian blur, and logarithmic transformations, the data is converted into an image. Computer vision techniques like binarization and contour detection are used to identify circular shapes based on circularity scores.

Tabular Approach

Circle Detection

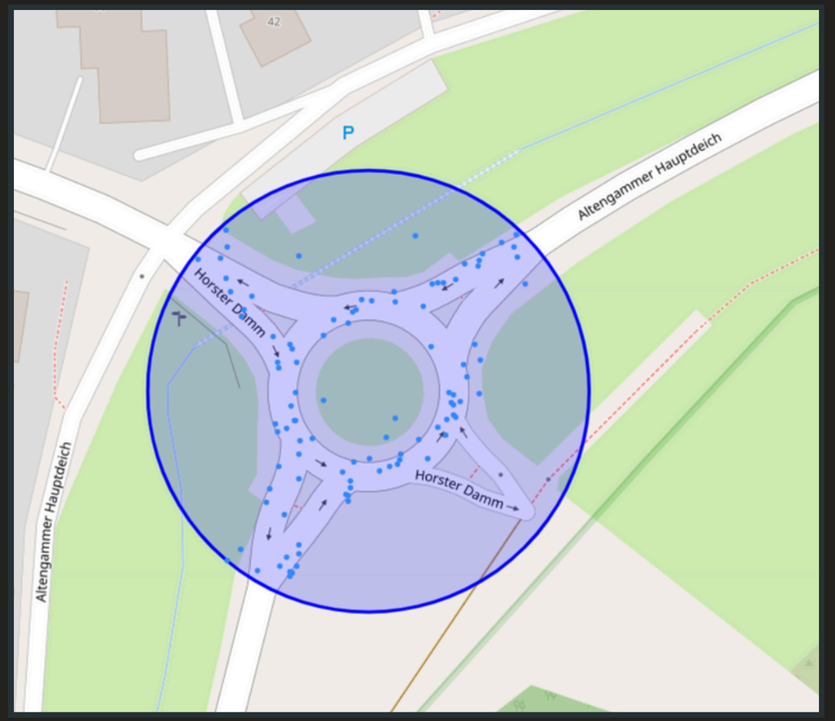
- Determine if a given point lies within any predefined circular region.
- Define circular areas using their centers (latitude and longitude) and radii.
- For each point, calculate the distance to the center of each circle.
- If the point is within the radius of any circle, it is considered inside the circle.



Tabular Approach

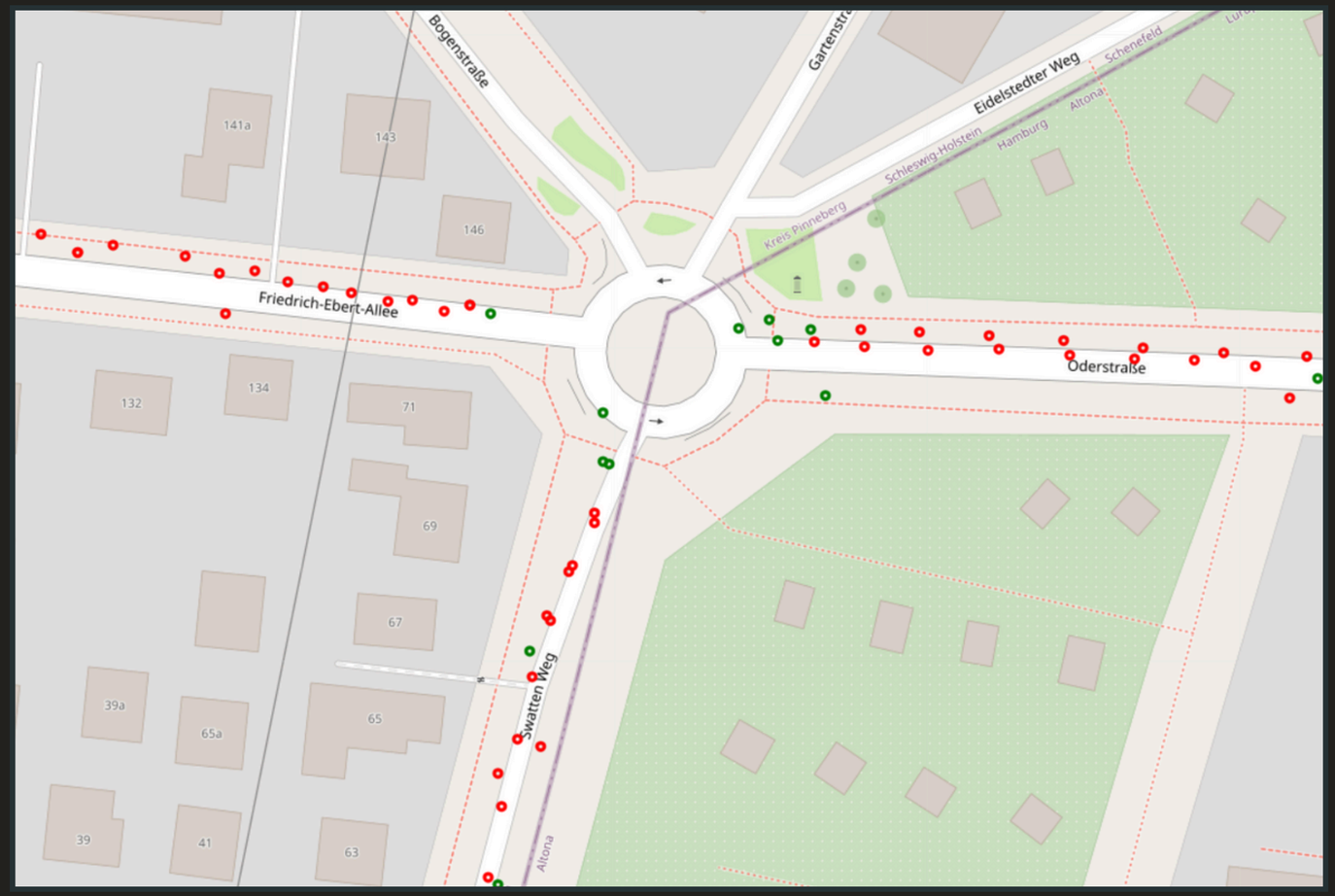
Headings Difference Calculation

- Capture how the heading (direction of movement) changes between consecutive points.
- Data points are grouped by their trace (a series of points that belong to the same path), and the difference in headings between each consecutive point is calculated.
- This helps identify sharp turns or changes in direction, which can indicate the presence of circular movements, like those found in roundabouts.

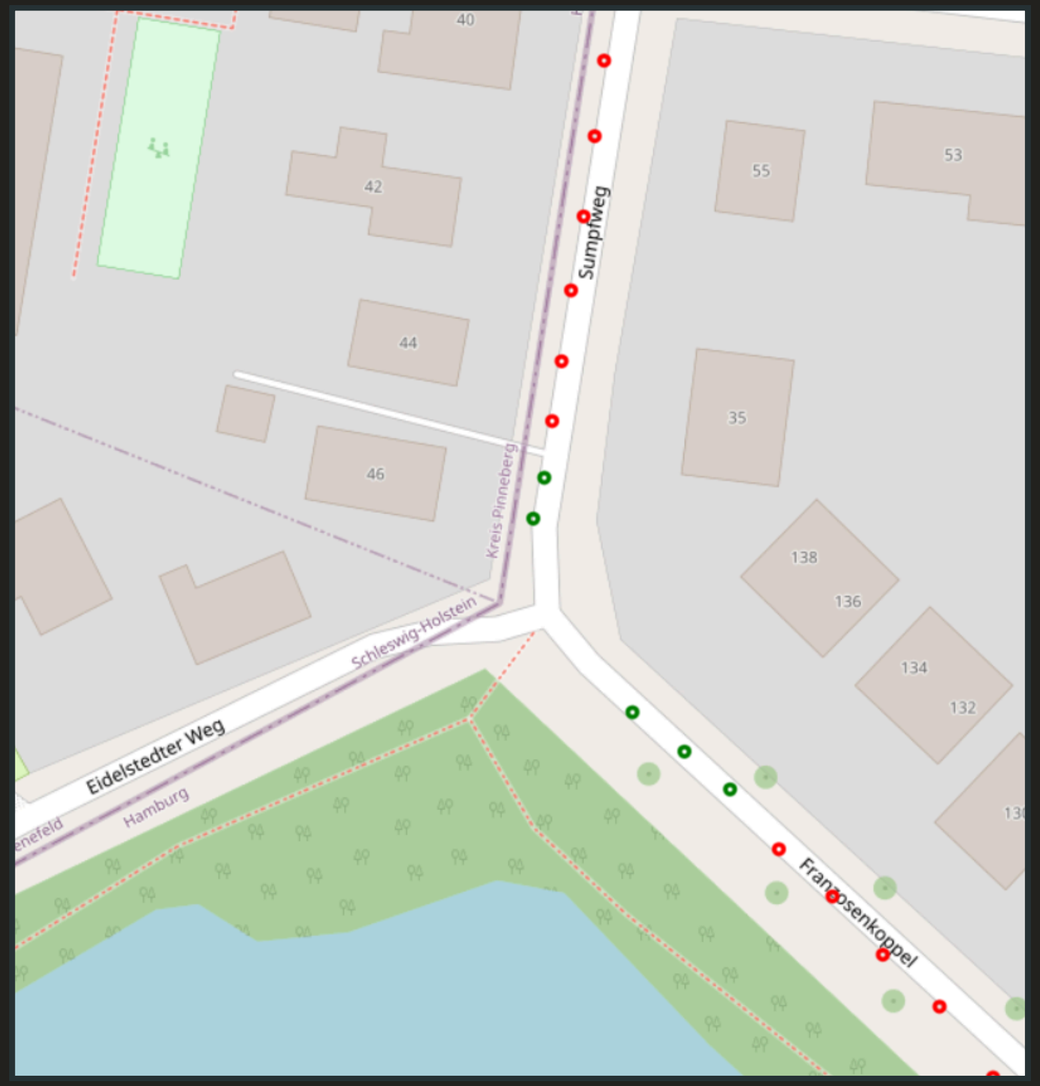


Tabular Approach

Tabular Approach -Testing



Tabular Approach -Errors



Tabular Approach -Results

Get Accurcay score

```
from sklearn.metrics import accuracy_score

accuracy = accuracy_score(day3_data['within_circle'], testing_data['predictions'])
print(f'Accuracy: {accuracy}')
```

[100] ✓ 0.0s Python

```
Accuracy: 0.8284422436373756
```

```
from sklearn.metrics import confusion_matrix

tn, fp, fn, tp = confusion_matrix(day3_data['within_circle'], testing_data['predictions']).ravel()
false_positive_rate = fp / (fp + tn)
print(f'False Positive Rate: {false_positive_rate}')
```

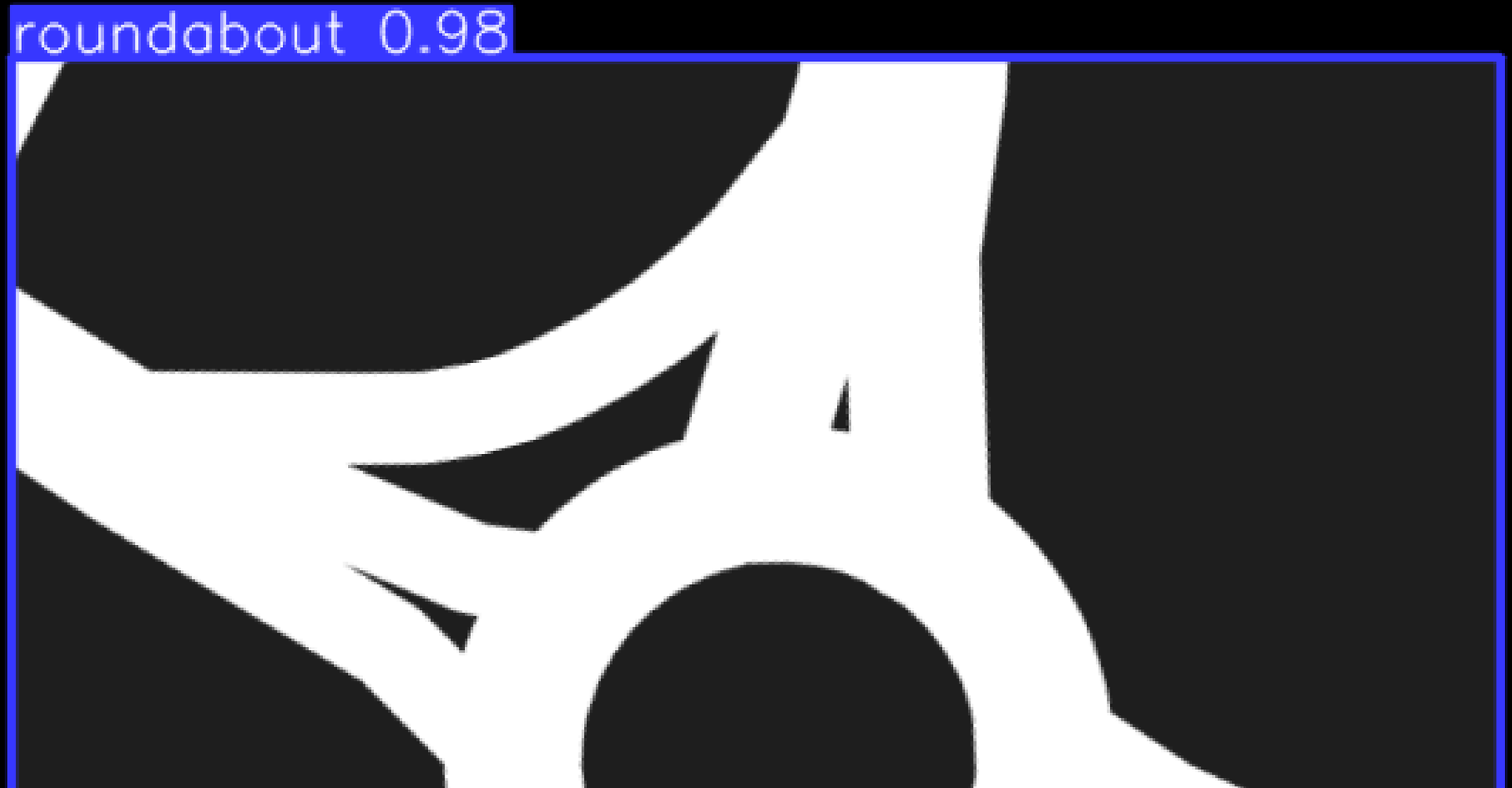
[101] ✓ 0.0s Python

```
False Positive Rate: 0.1660046444309398
```

Tabular Approach Improvements

- Features:
  - Rotation
  - steering angle
  - Wheel Base
- Neural networks, model selection, and fine-tuning
- Handling data as time-series
  - Using RNNs and LSTMs
- detect user activity
- Generate accurate labels for roundabouts
- Determine what to do with yield signs, traffic lights, stop signs, road links, and navstreets.
- feedback model in case in U-turns

Computer Vision Approach



Data Gathering

- Challenge: Lack of clear, top-view images of roundabouts.
- Initial Attempt: Generated images using generative AI but did not meet quality requirements.

Image Capture

- Utilized Snazzy Maps and Google Maps styling to capture roundabouts.
- Applied specific color stylings to resemble heatmap data from probe sources.

Image Augmentation

- Used OpenCV for transformations, including rotation and flipping
- Randomly placed roundabouts at different sizes and positions on blank canvases

Data Preparation

- Created bounding boxes around roundabouts in the augmented images
- Prepared data for training the model on roundabout detection and classification

Model Training/Prediciton

- **Model Selection**: Chose YOLOv8 Nano for its lightweight architecture and speed.

Training

- Utilized the prepared dataset of transformed and augmented roundabout images.
- Included diverse scenarios by varying size, rotation, and placement of roundabouts.
- Included other examples on other road parts that are not roundabouts to train on.

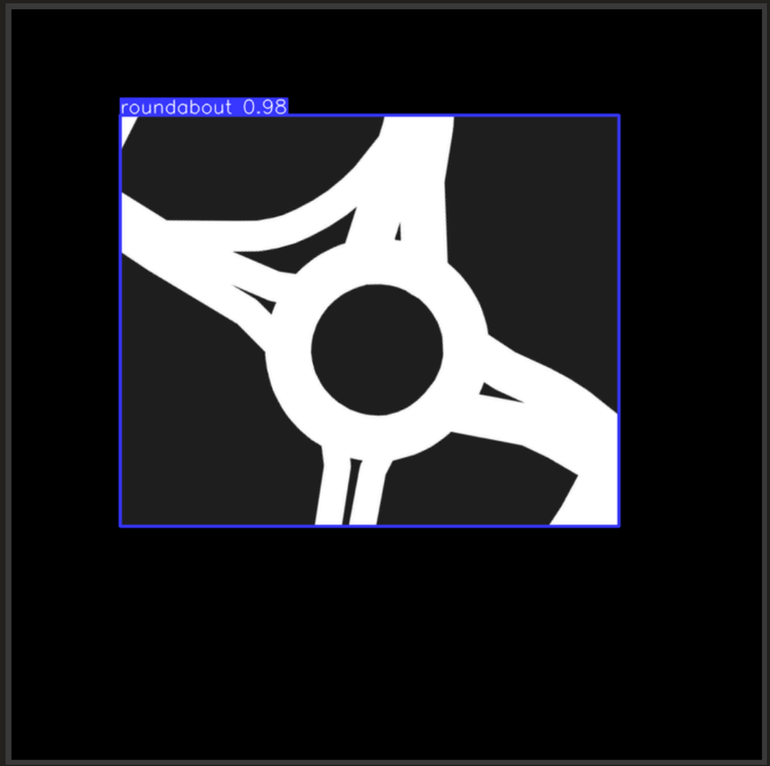
Prediction Examples

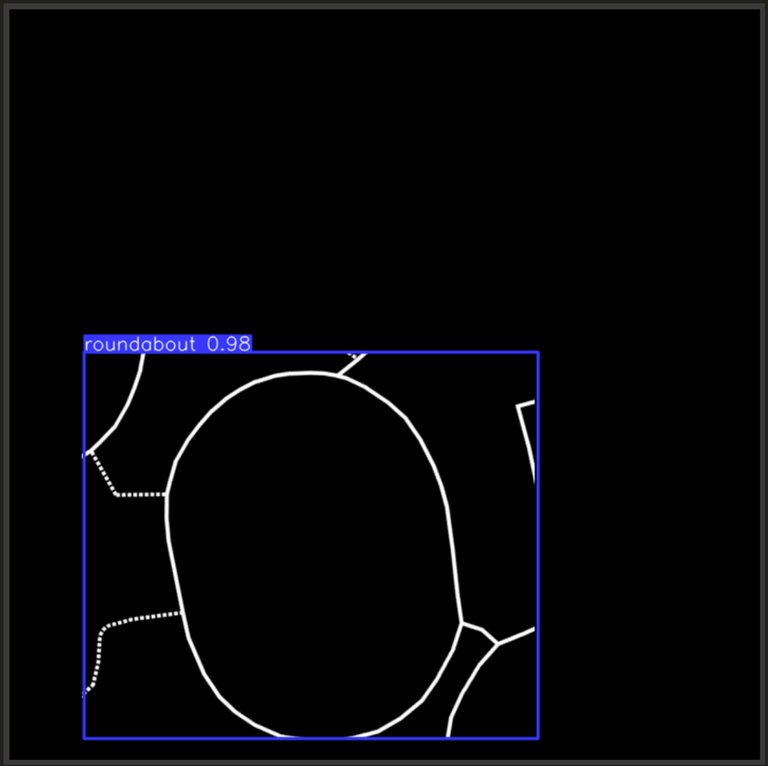
- Displaying model predictions on a subset of the dataset in the next slide
- Illustrates the model's ability to detect and classify roundabouts accurately

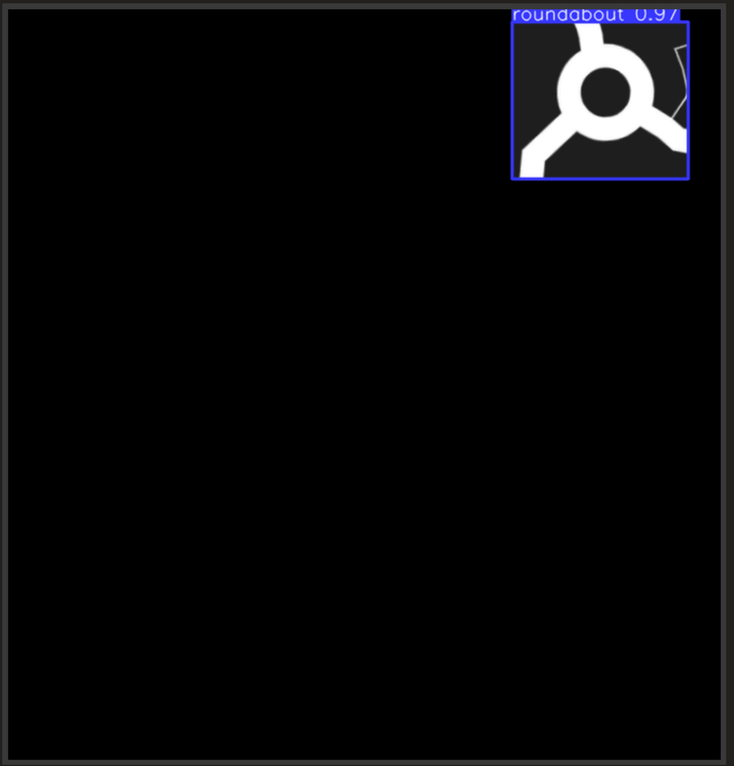
Key Results

- Fast inference suitable for real-time applications
- Demonstrated high accuracy on synthetic dataset

Prediction Examples







Improvements

- **Model Selection**: Chose YOLOv8 Nano for its lightweight architecture and speed.

Adding Noise to Data

- Include vertical or horizontal lines to simulate random roads.

Connect Roads

- Integrate roundabout images with road networks.
- This would require additional time for data labeling.

Utilize Heatmap Images

- Integrate roundabout images with road networks.
- This would require additional time for data labeling.

Explore Models

- Using larger YOLOv8 models, such as YOLOv8-Large or higher.
- Other YOLO versions like YOLOv10 as they made architectures with smaller/fast models but similar accuracy.
- Or alternative frameworks like Meta's Detectron.
- This is reasonable since extensive inference is not required.

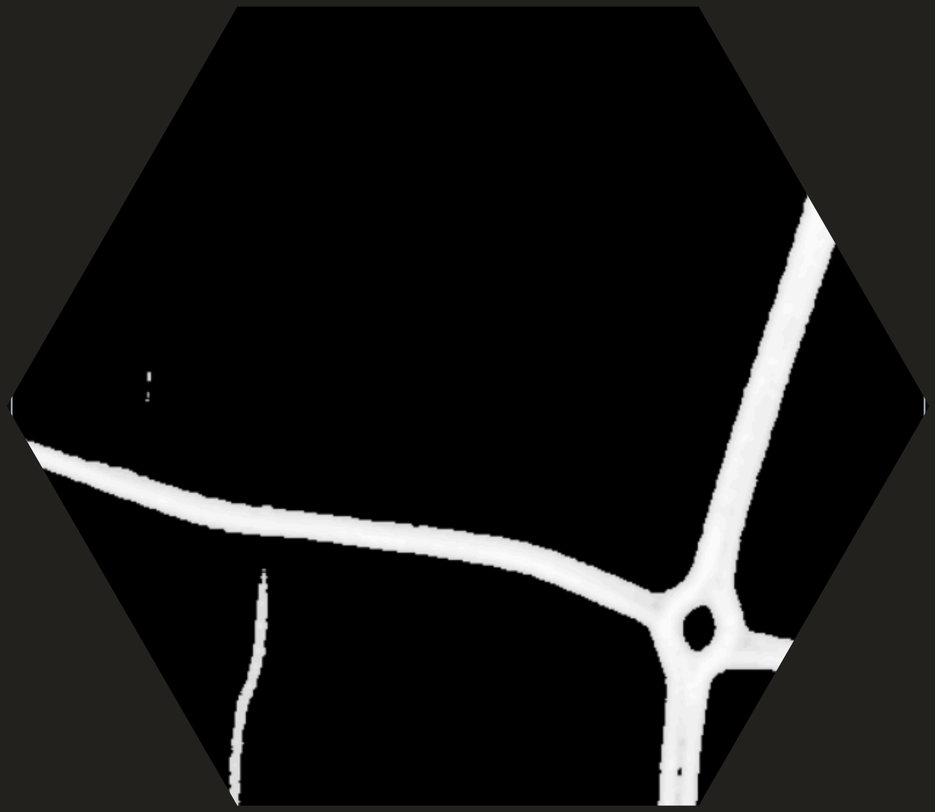
CV Second Approach

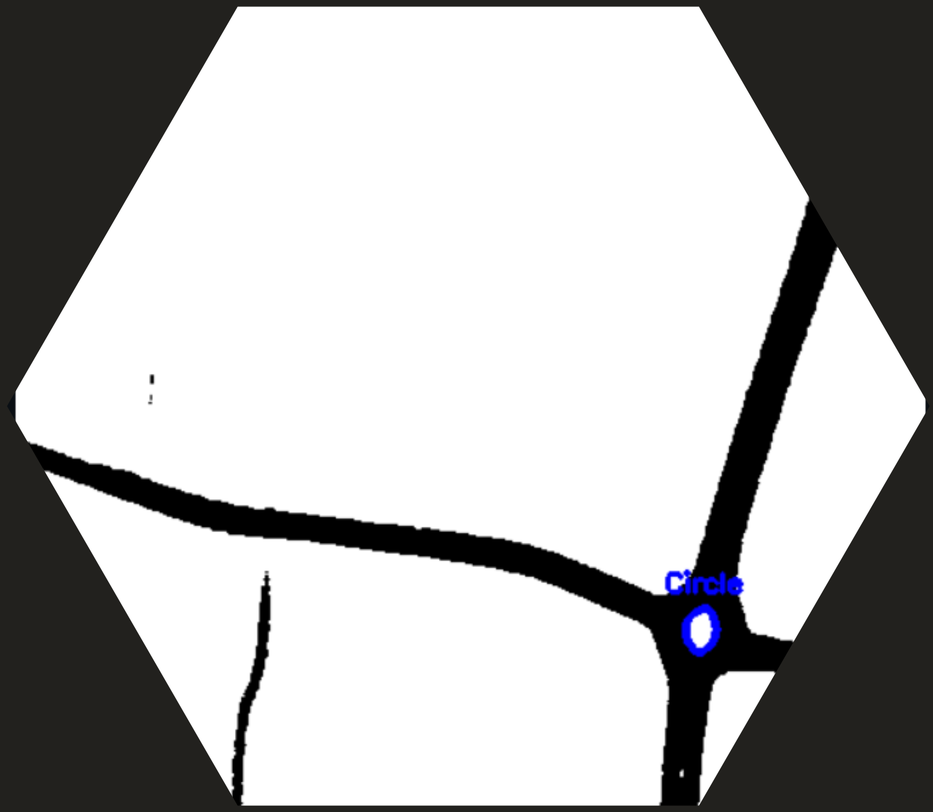
- Each folder of data is loaded in its entirety and subdivided into smaller chunks based on the longitude and latitude columns.
- A sliding window of size 2 is applied to the chunks, allowing for a focus on specific regions of the map rather than viewing it as a whole.
- Next, the chunks are aggregated into a grid with a resolution of 500 x 500, treating the dataset like an image.
- The aggregation is done by counting the number of data points projected onto each pixel of the grid.
- After aggregation, outliers are removed, followed by the application of Gaussian blur.

CV Second Approach

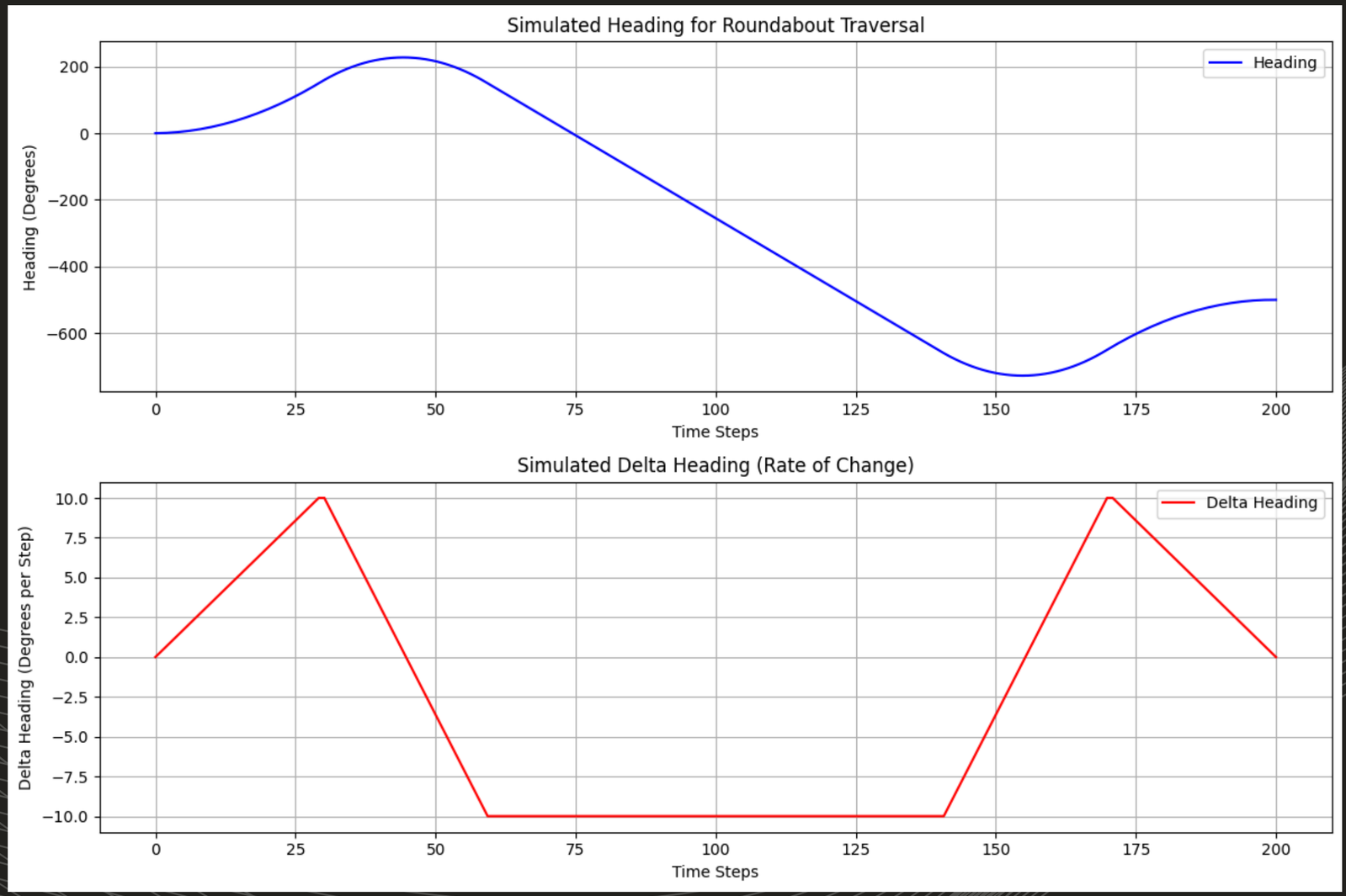
- A logarithmic transformation is then applied twice to the data to prevent skewing from large or unusual values.
- Once the data is converted into an image, computer vision techniques are applied.
- The image is binarized, inverted, and contour detection is performed.
- The detected contours are sorted based on circularity scores, and the contours representing circles are returned.

CV Second Approach





Pattern Recognition (Only an Idea)



Pattern Recognition (Only an Idea)