# Reverse search in Mandelbrot set

Kristijonas Silius, Julio Torres Fernández

Vilnius University

December 18, 2024

# Overview

# Mandelbrot set

The Mandelbrot set is a two-dimensional set described by the iteration of a simple function on the complex plane.

Within this set, there are convergent and divergent values, with the former laying inside of the set while the latter outside of it.

$$\text{Mandelbrot set: } f_c(z) = z^2 + c$$

$$\text{Iteration: } f_c(0),\ f_c(f_c(0)),\ f_c(f_c(f_c(0)))...$$

# Mandelbrot set

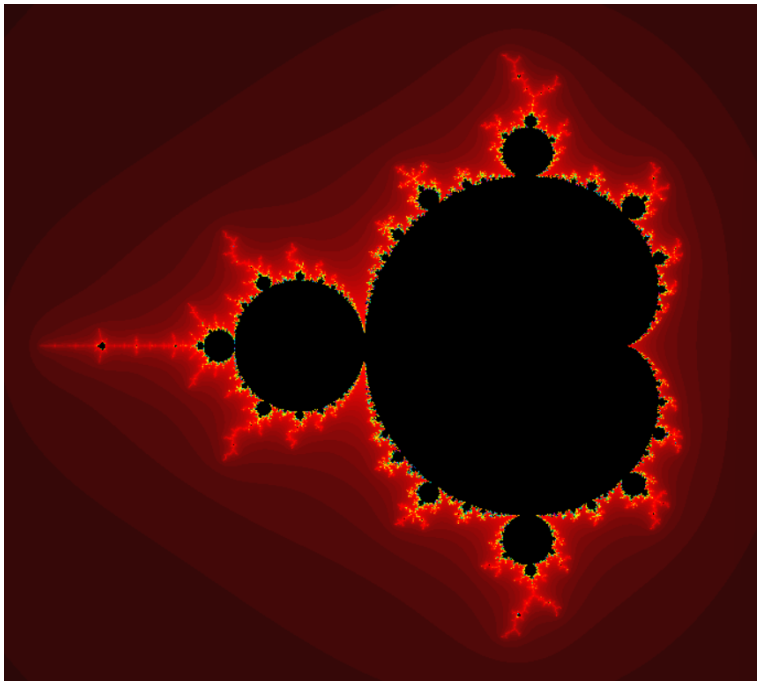

Figure: Mandelbrot set.

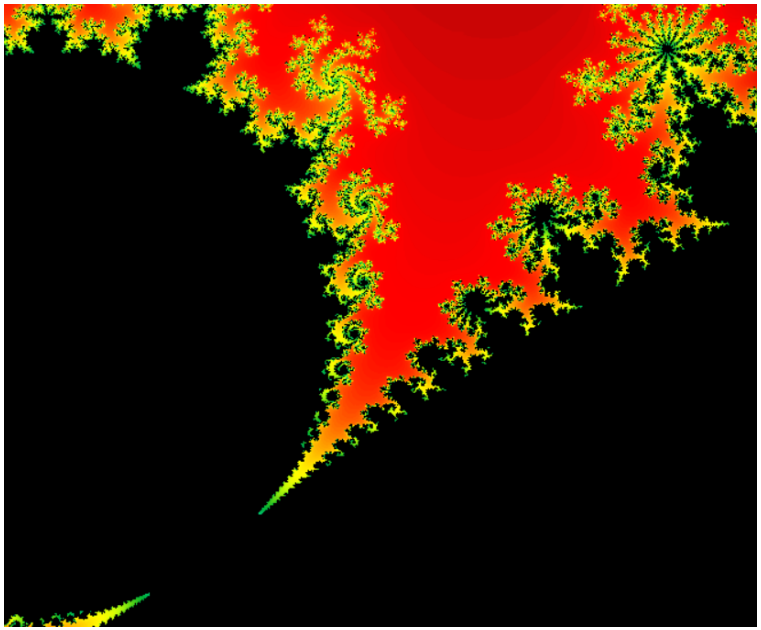

Figure: Mandelbrot fragment and fractal curves.

# Mandelbrot set - Assignment.

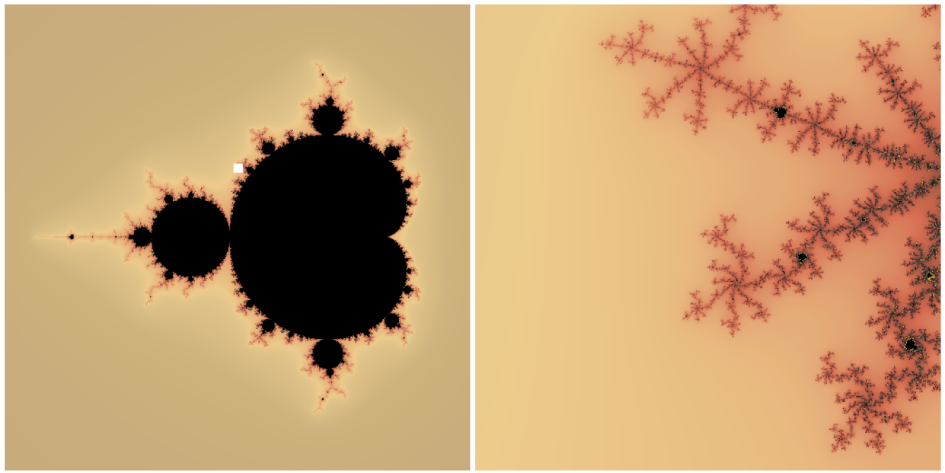

Figure: Assignment example.

- Convolutional Neural Network.
- Featuring matching approach.

# Convolution Neural Network.

In CNN, all neurons share the same weights and biases $\rightarrow$ All neurons can identify the same features.
CNN can have multiple layers $\rightarrow$ each layer can focus on a different feature.

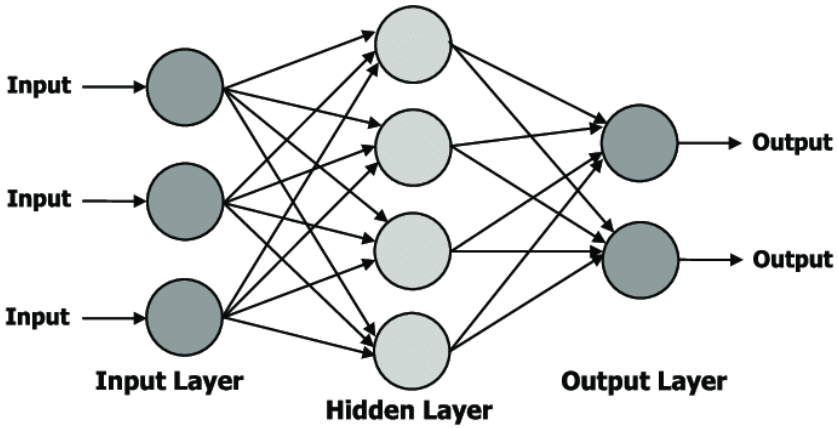


Figure: Typical Neural Network.

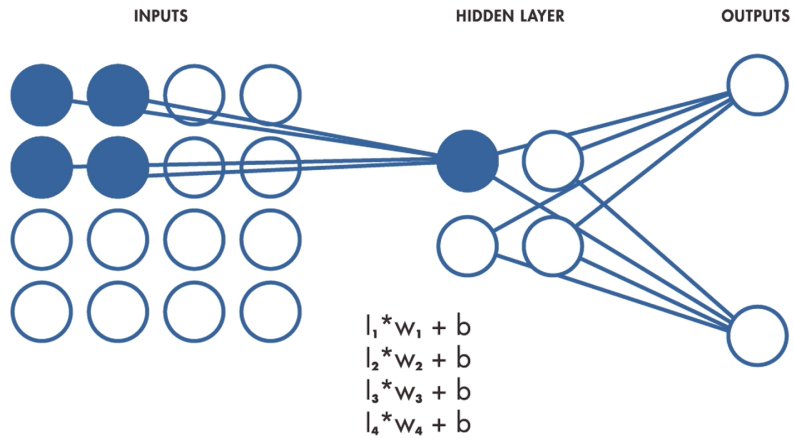


Figure: Convolution neural network.

# Convolution Neural Network.

Activation and pooling are further measures to reduce the learning workload in terms of dimensions and feature identification.

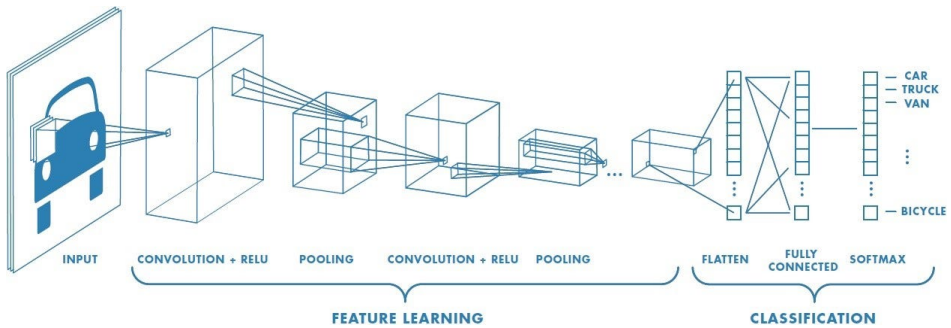


Figure: Reduction of dimensionality at CNN with activation and pooling.

**Let's create and train the model**

1. Create and train the dataset.
2. Load and train the model.
3. Search and zoom functions.
4. Output (GIF)

# CNN - Create Dataset.

- Define Mandelbrot. $f_c(z) = z^2 + c$
- Generate random sections of it:
  - x_min = np . random . uniform ( -2 , -0.8)
  - x_max = np . random . uniform ( x_min + 0.1 , x_min + 0.6)
  - y_min = np . random . uniform ( -1.4 , 1.4 - ( x_max - x_min ) )
  - y_max = y_min + ( x_max - x_min )
- Generate 1000 images. (Image size = 480*480 pixels. Approx. 30min.)

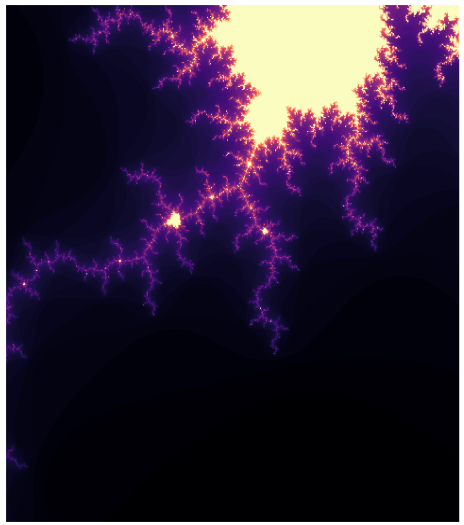

Figure: Example of image generated.

# CNN - Model

1. Load the data.
2. Train the model.
3. Store the model.

**Characteristics.**

1. Image size.
   - 64*64 pixels
   - 480*480 pixels
2. 3 layers.
3. Activation and pooling. (Number of filters and reduction of spatial dimentions by 2).
4. Dropouts (10 or 20 %)

# CNN - Zoom and output.

**Zoom**

- Overlap of 30 images
- Each image is created by Mandelbrot function, closing in the coordinates defined by the model.
- Generate GIF as an output.

**Output**

- Zoom functions runs the model.
- Identifies the coordinates and approximates the location.

# CNN - Results

2 models trained on the same dataset:

- Updated model: load_data(480pixels) and dropouts.
  - Model: load_data(64pixels) and no dropouts.

| | Updated model | Model |
|---|---|---|
| Training Mae | 0.7521 | 0.1867 |
| Validation Mae | 0.7393 | 0.3359 |
| Training Loss | 0.7624 | 0.0800 |
| Validation Loss | 0.7286 | 0.1974 |

Table: Final training and validation metrics.

# CNN - Results (Training Metrics)

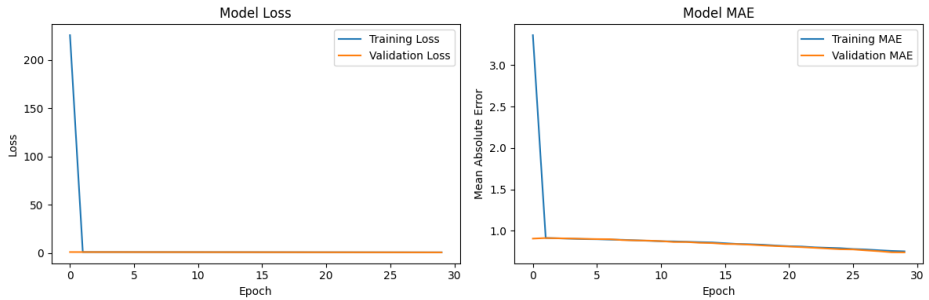


Figure: Training metrics updated model.

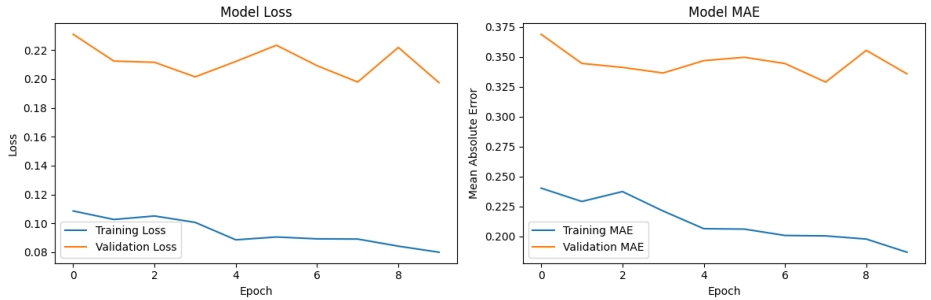


Figure: Training metric original model. (Overfitting)

# CNN - Results (Training Metrics)

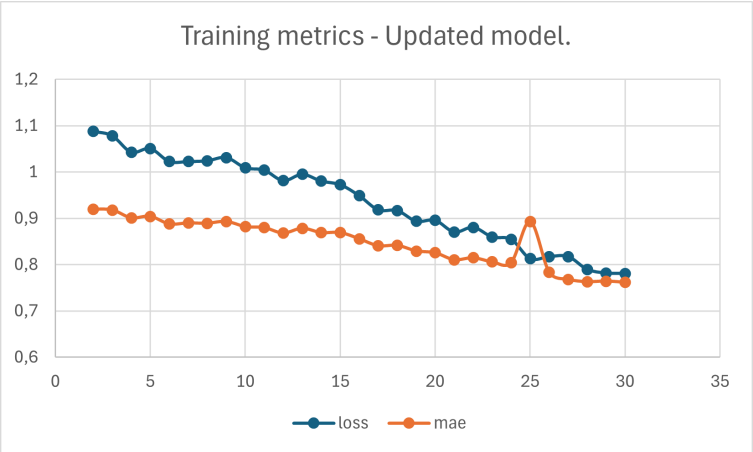


Figure: Training metrics excluding the first odd, epoch.

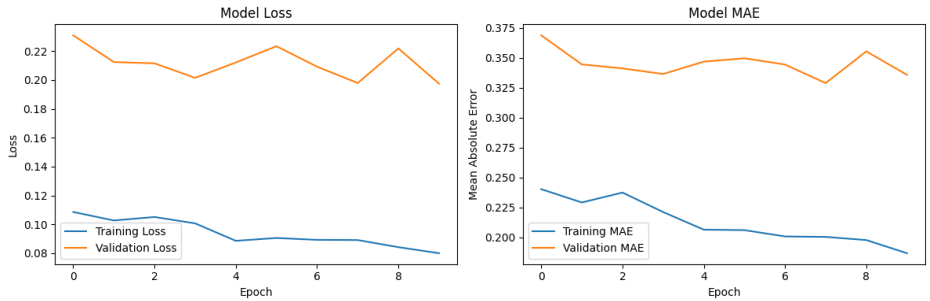


Figure: Training metric original model. (Overfitting).

# CNN - Results

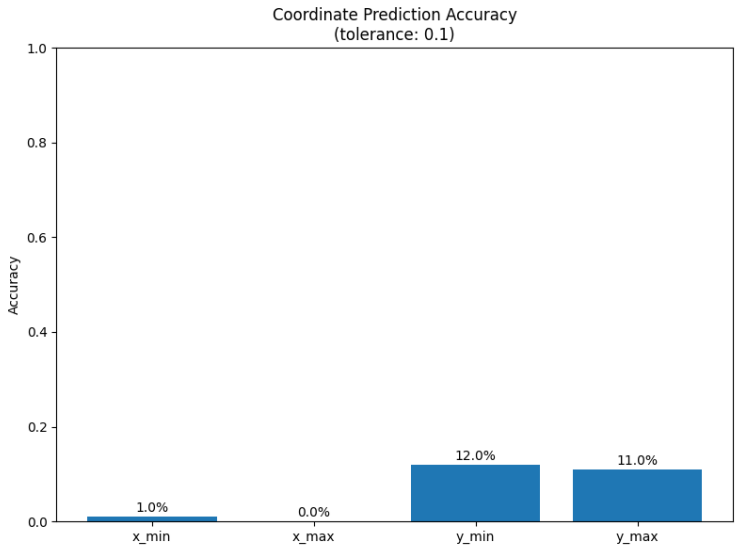


Figure: Updated model.

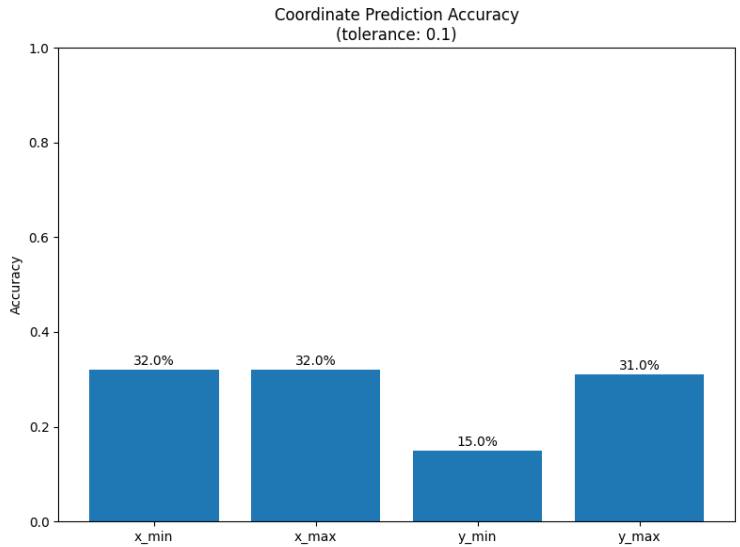


Figure: Model.

# CNN - Example in the three models.

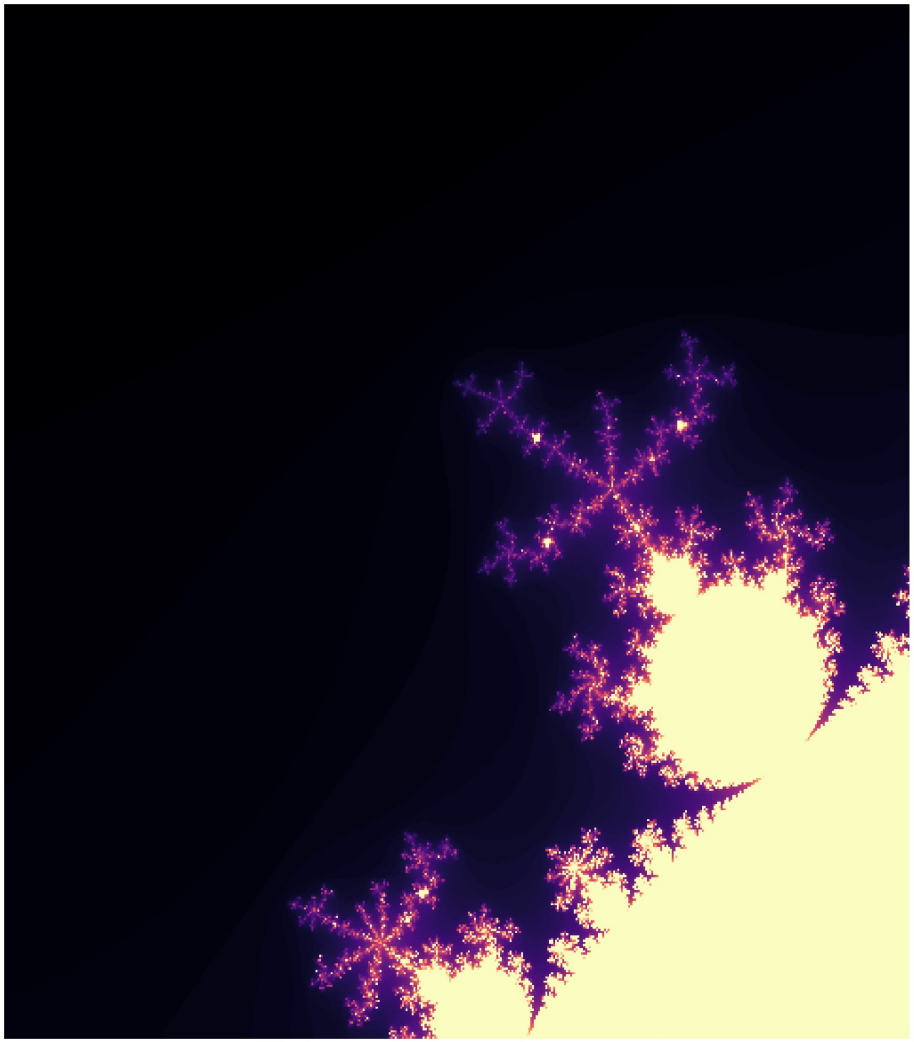

Figure: Fragment used to compared the results in different models.

# Feature Matching Overview

The core idea of this method is to identify locations within the Mandelbrot set by comparing visual features between an input image and a database of reference images.

- Uses SIFT (Scale-Invariant Feature Transform)
- Matches distinctive points between images
- Database of pre-processed reference images

# How It Works

1. **Feature Detection**
   - Extract SIFT keypoints and descriptors
   - Each point has location, scale, orientation
2. **Feature Matching**
   - FLANN-based matching algorithm
   - Fast search through reference database
3. **Location Prediction**
   - Best match determines position and zoom
   - Coordinates from reference image

# SIFT

SIFT operates in two main steps:

1. **Keypoint Detection**:
   - Identifies distinctive points in the image
   - Each keypoint has:
     - Location (x, y)
     - Scale (size of the region)
     - Orientation (dominant gradient direction)
2. **Descriptor Computation**:
   - For each keypoint, computes a 128-dimensional descriptor
   - Descriptor captures gradient information around the keypoint
   - These descriptors are what get matched between images

# FLANN

**FLANN** (Fast Library for Approximate Nearest Neighbors).
**Algorithm**: FLANN with kd-tree index

1. For each input feature, find k=2 nearest neighbors
2. Apply Lowe's ratio test:

$$ratio = \frac{distance_{best\_match}}{distance_{second\_best}} < 0.7$$

3. If ratio $<$ threshold - it's a good match

# FLANN

It is essentially an efficient search algorithm for nearest neighbor search problems, it:

1. Creates multiple kd-trees to organize points for our SIFT descriptors
2. For each query point:
   - Searches these trees in parallel
   - Checks a specified number of leaf nodes
   - Returns the closest matches found

# Implementation Details

- **Database**
  - 1000 reference images
  - 80-20 train-test split
- **Matching Process**
  - SIFT for feature detection
  - FLANN with kd-trees for efficient search
  - Ratio test for match filtering

# Results

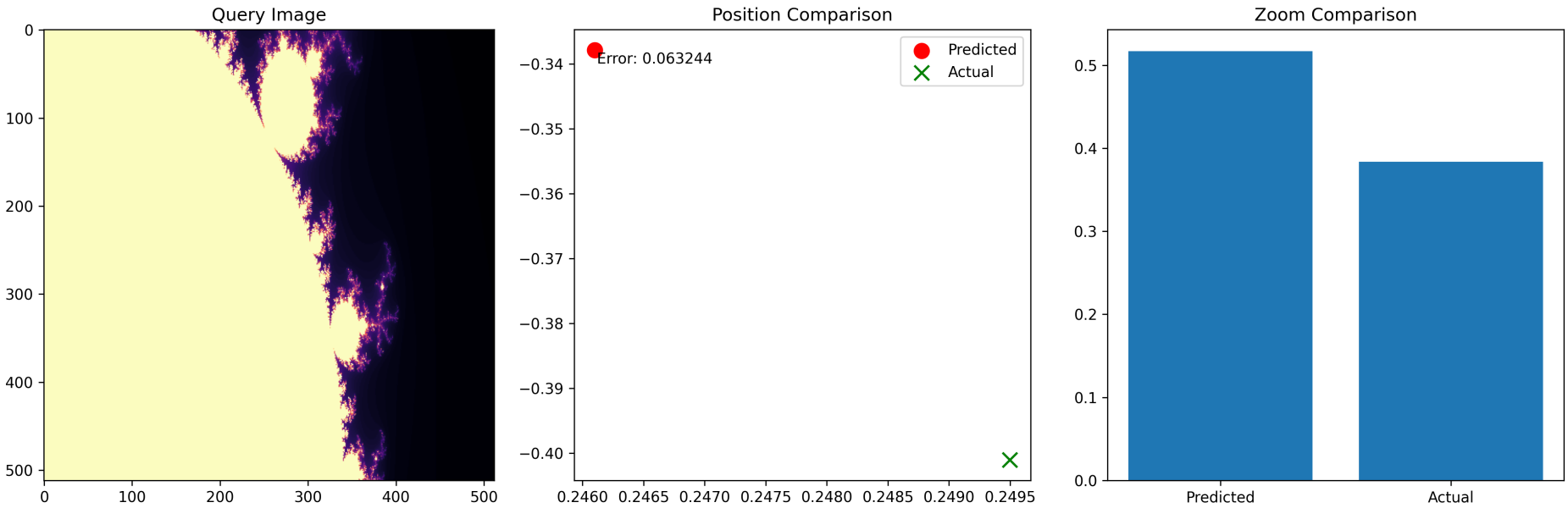


Figure: Example matching result showing position and zoom prediction.

# Results

**Test Set Results:**

- Total test images: 200
- Successful matches: 186
- Failed matches: 14
- Success rate: 93.0%
- Average position error: 0.494662
- Average zoom error: 0.226827
- Average matches found: 87.7

# Results

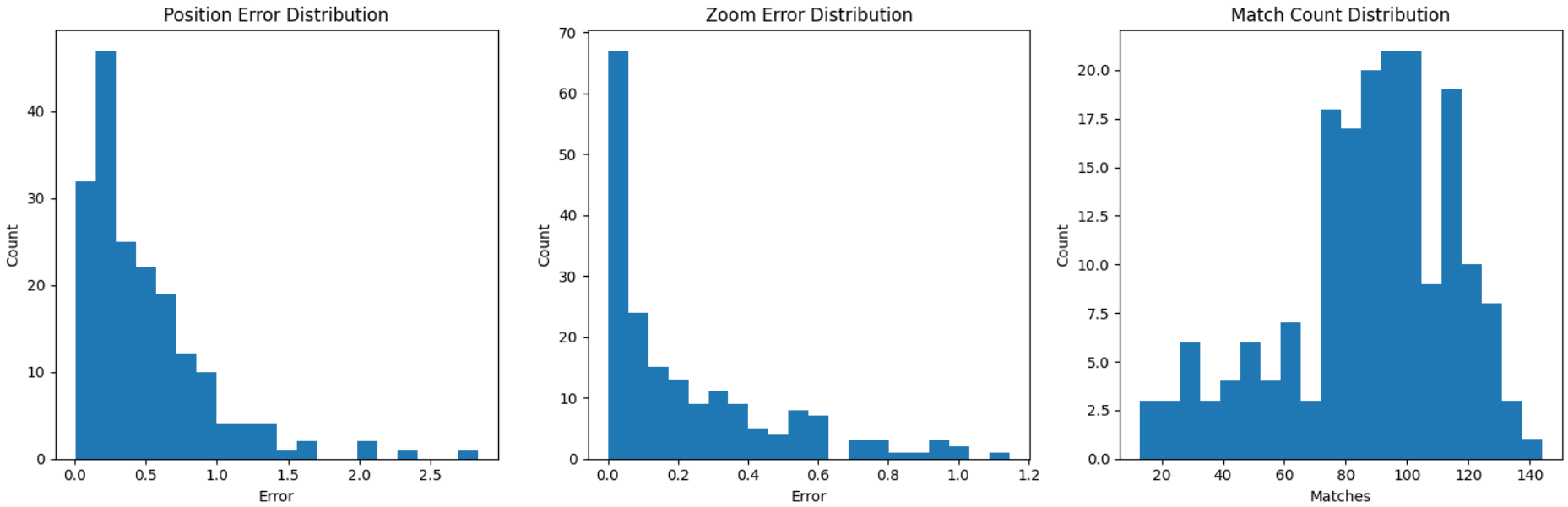


Figure: Distribution analysis of feature matching performance. On the left we have position error distribution showing histogram of different errors in coordinate prediction, in the center - zoom error distribution showing the accuracy of zoom level predictions, and on the right match count distribution displaying the number of successful SIFT feature matches found per image.

# Interactive Demo

Figure: Feature matching-based zoom animation

# The End