INT3404E 20 - Image Processing: Final Report

Group 11

1 Introduction

Sino-Nom is a complex historical writing system to represent the Vietnamese language based on Chinese characters with various modifications, used predominantly from the 13th to the early 20th century.

Sino-Nom characters present unique challenges for image processing due to their intricate designs and the deterioration of historical documents over time. The task of retrieving these characters from a 3D database using 2D image queries adds another layer of complexity, requiring sophisticated techniques to effectively map and recognize the characters.

In the task presented, we are provided with a list of 2D image queries and a database containing 3D models in STL format. The goal is to find the 5 most relevant database entries to each query and log the results in a csv file. The result will be ranked based on Mean Reciprocal Rank (MRR) on a private dataset.

This report contains the introduction to the problem, the characteristics of the sample dataset, the approaches used, results on each dataset, an analysis of the result and a conclusion.

2 Data

2.1 Input

- A list of 2D images queries. Characteristics:
  - File type: PNG image
  - Size and color model: 512x512 RGB images
  - Image quality: gray background; characters have fuzzy edges; strokes of out-of-frame characters occasionally appear at the edges; some characters are missing strokes or have too many strokes, creating a lack of clarity.

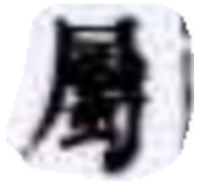

Figure 1: Example of a query

- A database containing 3D models in STL format. Characteristics:
  - Extracted from 3D scan
  - Containing a lot of noises (bumps, grooves, clefts), creating difficulty in identifying the shape of characters , thus requiring pre-processing to improve clarity

The dataset provided for the task before evaluation contains three folders: **pairs**, **database** and **queries**.

Within the **pairs** folder, there are two subfolders containing the sample queries and their corresponding STL 3D models, marked by the names of the file. For example, 0.stl is the correct 3D label for the query 0.png. The number of pairs here is 227.

The **database** has 251 3D models of Sino-Nom characters, and the **queries** has 251 2D PNG images; these two folders are linked through a *label.csv*, showing the five most relevant 3D models to each query.

2.2 Output

The output is formatted as a csv file containing the name of 5 best matches from the 3D database for each 2D queries, similar to the *label.csv* file.
The format of the output csv file is as follows:

query_name_file_1,“output_file_1,output_file_2,output_file_3,output_file_4,output_file_5”

These matches will be used for calculating the final MRR score for assessment.

3 Approach

3.1 Preprocessing

- 2D queries
  - Convert the color space of queries to grayscale, and then apply a threshold to generate binary images. The threshold was applied through the function **cv2.threshold(gray, 150, 255, 0)** with gray is the grayscale image.
  - These images are then cropped based on their contours, determined via **cv2.findContours**. We resize the cropped images to 256x256.
- 3D Database
  - Each STL model is loaded, then cut at a plane perpendicular to the z-axis at a fixed 6/7 of model's height. In other words, all vertices above the cutting plane are used for the processed mesh.
  - The processed meshes are positioned so that the z+ axis face up and plotted, or flatten, to generate a 2D version of the database, which are also preprocessed with the 2D procedure above.

3.2 Feature Extraction and Matching approaches

We tried several different feature extraction and matching algorithms after processing the data. These algorithms are as follow:

- Histogram of Oriented Gradients (HOG) and Cosine Similarity

  HOG is used to detect objects in an image by focusing on the structure or the shape of the object. The steps are:
  - **Gradient Computation**: Compute the gradient of the image intensity to highlight regions with strong intensity changes.
  - **Orientation Binning**: Divide the image into small connected regions called cells, and for each cell, compute a histogram of gradient directions.
  - **Normalization**: Normalize these histograms within larger regions called blocks to improve invariance to illumination changes.
  - **Feature Vector**: Concatenate the normalized histograms to form a feature vector representing the image.

  - **Comparison**: Compare HOG descriptors (feature vectors) of the two images using a distance metric like Euclidean distance. The selected descriptor here is **cosine similarity**

  HOG is predefined in a function called **hog** in the **skimage.feature** library, and cosine similarity is imported from **sklearn.metrics.pairwise**.

- Template matching

  Template Matching is a technique used to find parts of an image that match a template image. The basic steps are:

  - **Create a Template**: Select the template image you want to find in the larger image.
  - **Sliding Window**: Slide the template image across the larger image and compare the template to the sub-regions of the larger image.
  - **Similarity Metrics**: Use similarity metrics (e.g., normalized cross-correlation) to measure how well the template matches the sub-region of the image.
  - **Detection**: The locations with the highest similarity scores are considered the matches.

  We used **cv2.matchTemplate** to compare the 2D processed versions of queries against the 2D database

- Structural similarity index measure (SSIM)

  SSIM measures the similarity between two images based on structural information. The steps are:

  - **Luminance Comparison**: Compare the brightness of the images.
  - **Contrast Comparison**: Compare the contrast of the images.
  - **Structure Comparison**: Compare the structure of the images, focusing on how well the patterns of pixel intensities align.
  - **Overall SSIM Index**: Combine these comparisons into a single SSIM index value ranging from -1 to 1, where 1 indicates perfect similarity.

  SSIM comparison is provided in the **skimage.metric** library in the form of **compare_ssim**

- Oriented FAST and Rotated BRIEF (ORB)

  ORB is a fast and efficient alternative to SIFT and SURF, combining the FAST keypoint detector and the BRIEF descriptor. The steps are:

  - **Keypoint Detection**: Use the FAST (Features from Accelerated Segment Test) detector to find keypoints quickly.
  - **Orientation Assignment**: Assign an orientation to each keypoint based on the intensity centroid.
  - **Descriptor Computation**: Compute the BRIEF (Binary Robust Independent Elementary Features) descriptor for each keypoint, creating a binary string.
  - **Matching**: Match keypoints between images using the Hamming distance for the binary descriptors.

  ORB is included in **cv2** library

- Scale-Invariant Feature Transform (SIFT)

  SIFT is a robust algorithm to detect and describe local features in images. The steps include:

- **Keypoint Detection**: Identify keypoints in the image using a difference of Gaussians (DoG) that are invariant to scale and rotation.
- **Descriptor Computation**: For each keypoint, compute a descriptor based on the local gradient information around the keypoint, creating a 128-dimensional vector.
- **Matching**: Match keypoints between the two images by comparing their descriptors using a distance metric (e.g., Euclidean distance). Commonly, nearest neighbor matching is used.

SIFT is included in **cv2** library.

4 Results

We used the **pairs**, **database** and **queries** folder to compute the MRR for the results using each of the feature extraction and matching approaches. The results are shown in the table and figure below:

Approaches	Database	Pairs
HOG + Cosine	0.89345	0.7507
Template matching	0.89	0.7787
SSIM	0.8731	0.75504
ORB	0.7603	0.7617
SIFT	0.7743	0.7145

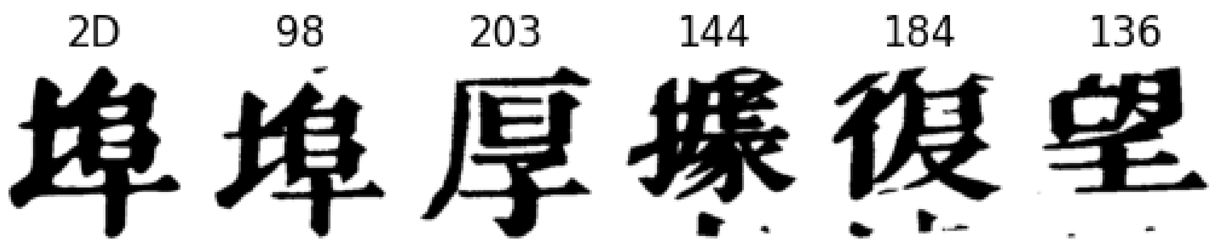

Figure 2: A plot of the five most relevant 2D version of database entries for a 2D query (Template matching)

During the assessment on the hidden dataset, we chose the Template Matching as the feature extraction and matching approach. The result received for the hidden dataset was approximately **0.57**.

5 Analysis

5.1 Weakness

The result for the hidden dataset was much lower than that from the given dataset. We suspect a number of reason for such an underwhelming MRR score:

- Because the threshold value was fixed at 150, it is possible that some images, specifically queries, return entirely black or entirely white results after thresholding, due to the noises from the original queries and the grayscale converter of cv2. This caused the cropping function to fail to find a bounding box, leading to some unhandled exceptions during the assessment process.

- We used the bounding box function provided to find the minimum and maximum coordinates on the x and y axis of the greyscale image. However, as mentioned in the Data section, most queries contain noise from surrounding characters at their border, resulting in rather inaccurate bounding box for the target character.
- We used the feature extraction and matching methods provided by the library **cv2** to reduce the runtime of the pipeline. It is possible that the raw implementation of these methods result in higher accuracy at the cost of run time.

5.2 Potential improvements

- To fix the potential errors with the fixed threshold mentioned, we can use **adaptive thresholding**to reduce the occurence of such polarizing results.
- For the bounding box problem, we suggest 2 approaches:
  - The pure image processing approach: we can process the noise at the edge of images, so that all the black smudges from neighboring characters are cleaned up, leaving only the central character intact.
  - The machine learning approach: we can use a model from task 1 - Sino-Nom character Localization to get the bounding boxes prediction with high confidence.

  .
- A more thorough examination of the implementation of the feature extraction and matching approaches within the **cv2** library is needed to fully understand and optimize the results for the task.

6 Conclusion

In this report, we addressed the challenging task of retrieving Sino-Nom characters from a 3D database using 2D image queries. This task required advanced image processing techniques due to the intricate designs of the characters and the deteriorated state of historical documents.

We explored various feature extraction and matching algorithms, including Histogram of Oriented Gradients (HOG), Template Matching, Structural Similarity Index Measure (SSIM), Oriented FAST and Rotated BRIEF (ORB), and Scale-Invariant Feature Transform (SIFT). Each method was evaluated based on its Mean Reciprocal Rank (MRR) performance on a provided dataset and a hidden dataset.

Our results indicated that Template Matching yielded the best performance on the provided dataset, though it did not perform as well on the hidden dataset. Several factors, such as fixed thresholding values and noise in the queries, contributed to this discrepancy. To address these issues, we proposed potential improvements, including adaptive thresholding and noise reduction strategies for bounding box accuracy.

Future work should focus on implementing these improvements and further optimizing the feature extraction and matching processes. By refining these techniques, we aim to enhance the accuracy and robustness of Sino-Nom character retrieval, contributing to the preservation and digital accessibility of this historical writing system.

7 Task Assignment

Member	Tasks Assigned	Percentage
Vũ Thành Long (21020647) (Team-leader)	Research 2D image preprocessing methods Preprocess 2D queries Use **SSIM** for matching Write reports	33%
Dương Quang Minh (21020219)	Research feature extraction and matching methods Preprocess 3D database Use **HOG + Cosine** and **SIFT** for matching	33%
Dương Bảo Long (21021514)	Research 3D preprocessing methods and libraries Preprocess 3D database Use **Template matching** and **ORB** for matching	34%