CS 5330: Pattern Recognition and Computer Vision (Spring 2024)

Report

Project 3: Real-time 2-D Object Recognition

Submitted by: Haard Shah
Hrigved Suryawanshi

Description:

The project is centred around 2-D Object recognition aiming to empower a computer to identify a specific set of objects placed on a white surface. The key objective is to achieve translation, scale, and rotation invariance in recognition, with objects identifiable from a top-down camera view. The system should be able to recognize individual objects placed in the image and show the position and category of the object in an output image. If provided a video sequence, it should be able to do this in real time. For development of this system, we are providing a "development image dataset" for training and testing the system. The final set-up will involve a workspace with an integrated desktop camera facing against a plain white surface. This system will make sure to differentiate at least 5 objects based on their 2-D shape and a uniform dark colour. Here in this project, we are making a real-time system to compute all the tasks.

Tasks:

1. **Thresholding the input Video:**

In the initial stage of object recognition pipeline, we implemented a dynamic thresholding algorithm to isolate objects from the background in the input video. This involved pre-processing the video frames with a Gaussian blur and saturation adjustment, followed by calculating a dynamic threshold based on the means obtained from k-means clustering (K=2) on a random sample of pixels. This approach effectively separated the darker objects from the white background, laying the groundwork for subsequent object recognition steps. The algorithm was integrated into the existing video framework, and thorough testing on the complete set of objects ensured effective object isolation.

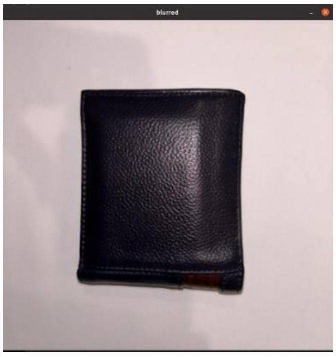


Image 1: Blurred picture

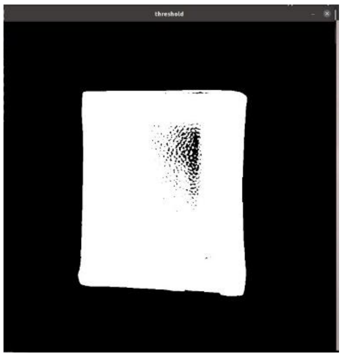


Image 2: Thresholded picture

Here, we displayed the required result i.e. applying gaussian blur and then having a dynamic threshold.

2. **Cleaning up the binary image:**

To refine the binary images obtained from thresholding, we employed targeted morphological filtering based on the observed issues. Erosion was used to reduce noise and eliminate small, isolated regions, while dilation helped fill in gaps and connect broken areas within the objects. These morphological operations were chosen based on the specific characteristics observed in the thresholded images to enhance object clarity and prepare them for further processing. The cleaned-up images demonstrated improved object definition and better separation from the background, contributing to the overall accuracy and reliability of the system.

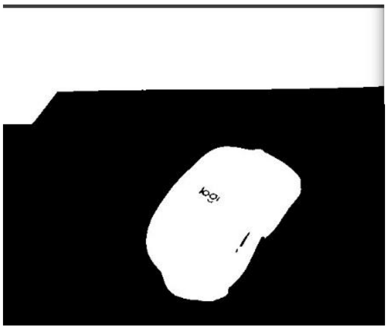


Image 3: Before dilation Image

Image 4: Dilated Image

Here, the required result is that we displayed the image without clean-up and the next image is the dilated image.

3. **Segmenting the image into regions:**

To segment the image into individual objects, we implemented connected component analysis (CCA) on the cleaned binary image. This process identified connected pixel regions, corresponding to distinct objects. We filtered out small regions to focus on potential objects and optionally limited the analysis to the N largest regions. Finally, each remaining region was assigned a unique color for clear visualization, forming a segmented image with distinct object boundaries. By utilizing OpenCV's connected components function and considering region size, centroid, and axis-oriented bounding box, the system effectively identified and displayed regions of interest while ignoring those that were too small. To enhance visual representation, a color palette was used to assign different colors to each region, ensuring clear differentiation. The region maps generated through this segmentation process provided valuable insights into the spatial distribution and characteristics of objects within the images.

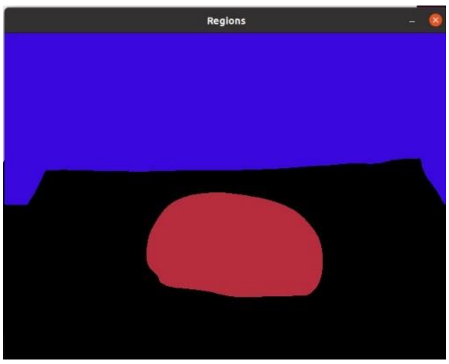


Image 5: Segmented into regions.

4. **<u>Computing features for each major region:</u>**

We implemented a function to extract features from each identified region, focusing on region-based analysis. This included the axis of least central moment to characterize object mass distribution and the oriented bounding box for size and orientation. Additionally, the aspect ratio of the bounding box was calculated. These features, displayed on the objects and incorporated into a feature vector alongside the axis of least central moment, provided essential information for subsequent object classification and recognition, while maintaining invariance to translation, scale, and rotation. The report will include the computed feature vectors for each object in the example images. Additionally, features such as percent filled and bounding box height/width ratio were implemented to analyze translation, scale, and rotation invariance. OpenCV's function for computing moments was utilized to extract relevant moment values, which were then used as features in the analysis. The system was designed to display at least one feature in real time on the video output, allowing for dynamic testing of feature invariance under different object orientations.

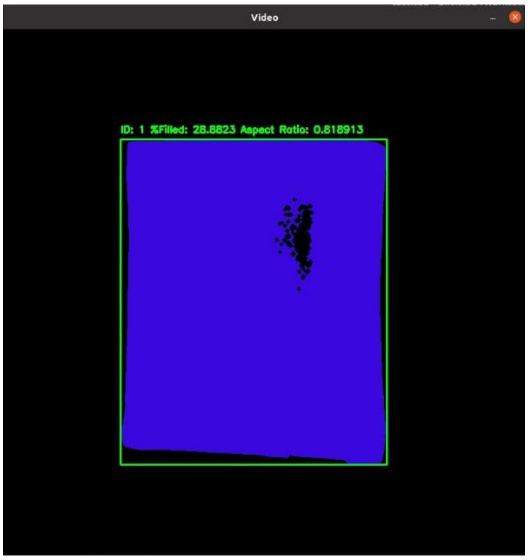


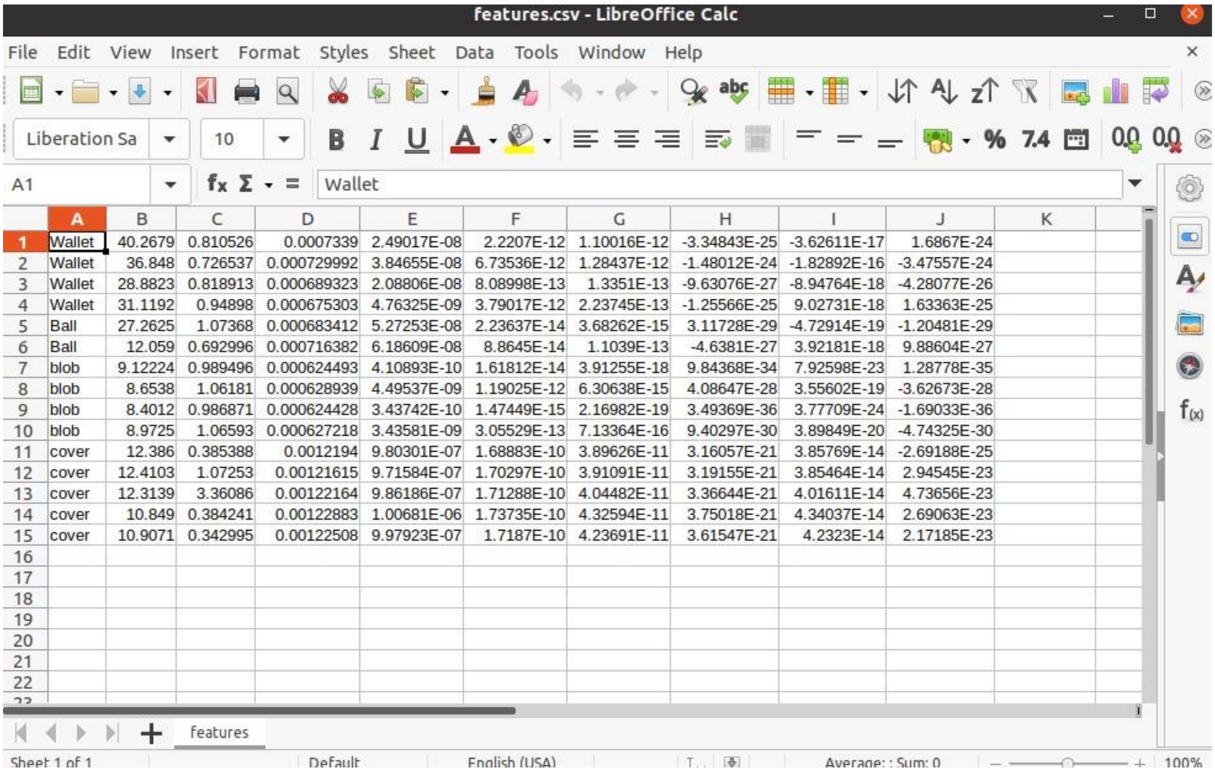
Image 6: Filtering small regions and displaying region ID

```
hrigved@ubuntu-broo:~/PRCV/P3/build$ ./test
Region ID: 1
Percent Filled: 27.2625%
Bounding Box Aspect Ratio: 1.07368
Hu Moments: 0.000683412 5.27253e-08 2.23637e-14 3.68262e-15 3.11728e-29 -4.72914
e-19 -1.20481e-29
```

Initially, our system was successful in classifying regions and extracting features for each frame in the video stream. However, we encountered a significant issue with data overload in our CSV file. This was since the system was recording features for every single frame, rather than just the frame corresponding to a key press event. This not only led to an unmanageable amount of data but also made it challenging to isolate the specific features we were interested in. To address this, we implemented a conditional feature recording approach, where features are only extracted and recorded upon the occurrence of a key press event. This modification significantly reduced the data load and provided a more manageable dataset for our analysis and classification tasks.

5. **<u>Collecting training data:</u>**

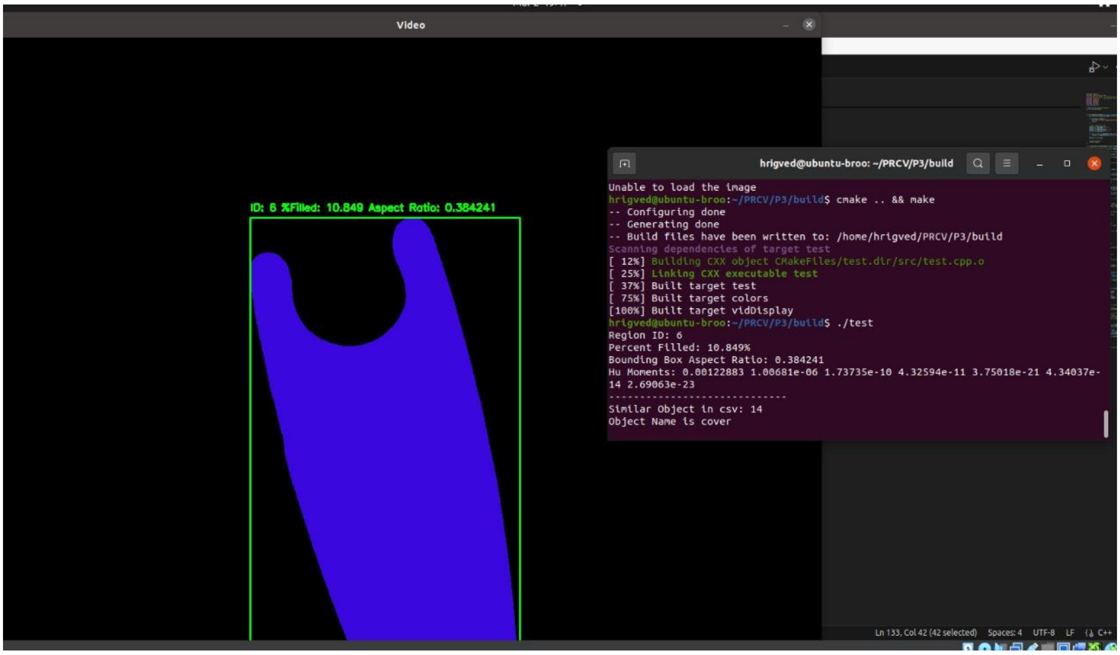
Our system offers a dedicated training mode for constructing a labelled dataset. When the user types (pressing "O"), the system prompts for an object label from user, extracts feature from the current image, and stores them alongside the label in a designated file format (CSV). This mechanism allows us to accumulate labeled feature vectors for various objects, crucial for training the object classification model in the recognition phase.



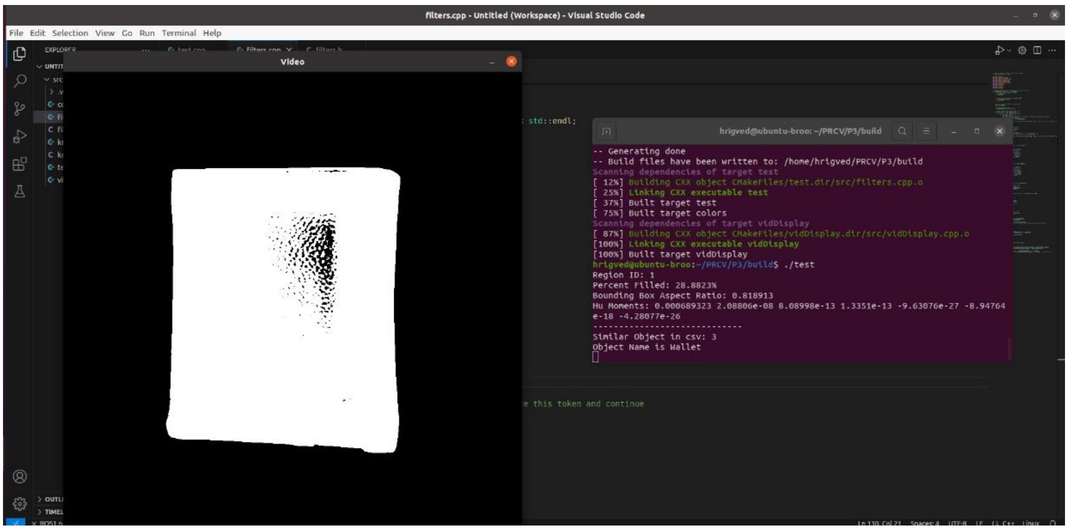
	A	B	C	D	E	F	G	H	I	J
1	Wallet	40.2679	0.810526	0.0007339	2.49017E-08	2.2207E-12	1.10016E-12	-3.34843E-25	-3.62611E-17	1.6867E-24
2	Wallet	36.848	0.726537	0.000729992	3.84655E-08	6.73536E-12	1.28437E-12	-1.48012E-24	-1.82892E-16	-3.47557E-24
3	Wallet	28.8823	0.818913	0.000689323	2.08806E-08	8.08998E-13	1.3351E-13	-9.63076E-27	-8.94764E-18	-4.28077E-26
4	Wallet	31.1192	0.94898	0.000675303	4.76325E-09	3.79017E-12	2.23745E-13	-1.25566E-25	9.02731E-18	1.63363E-25
5	Ball	27.2625	1.07368	0.000683412	5.27253E-08	2.23637E-14	3.68262E-15	3.11728E-29	-4.72914E-19	-1.20481E-29
6	Ball	12.059	0.692996	0.000716382	6.18609E-08	8.8645E-14	1.1039E-13	-4.6381E-27	3.92181E-18	9.88604E-27
7	blob	9.12224	0.989496	0.000624493	4.10893E-10	1.61812E-14	3.91255E-18	9.84368E-34	7.92598E-23	1.28778E-35
8	blob	8.6538	1.06181	0.000628939	4.49537E-09	1.19025E-12	6.30638E-15	4.08647E-28	3.55602E-19	-3.62673E-28
9	blob	8.4012	0.986871	0.000624428	3.43742E-10	1.47449E-15	2.16982E-19	3.49369E-36	3.77709E-24	-1.69033E-36
10	blob	8.9725	1.06593	0.000627218	3.43581E-09	3.05529E-13	7.13364E-16	9.40297E-30	3.89849E-20	-4.74325E-30
11	cover	12.386	0.385388	0.0012194	9.80301E-07	1.68883E-10	3.89626E-11	3.16057E-21	3.85769E-14	-2.69188E-25
12	cover	12.4103	1.07253	0.00121615	9.71584E-07	1.70297E-10	3.91091E-11	3.19155E-21	3.85464E-14	2.94545E-23
13	cover	12.3139	3.36086	0.00122164	9.86186E-07	1.71288E-10	4.04482E-11	3.36644E-21	4.01611E-14	4.73656E-23
14	cover	10.849	0.384241	0.00122883	1.00681E-06	1.73735E-10	4.32594E-11	3.75018E-21	4.34037E-14	2.69063E-23
15	cover	10.9071	0.342995	0.00122508	9.97923E-07	1.7187E-10	4.23691E-11	3.61547E-21	4.2323E-14	2.17185E-23

6. **<u>Classifying new images:</u>**

For classification of new objects, the system utilizes a nearest-neighbour approach with a scaled Euclidean distance metric to compare feature vector of the unknown object with stored in object database to classify new images objects. The nearest-neighbour recognition approach is employed to label the unknown object based on the closest matching feature vector in the database. The system displays the assigned label on the output video stream, allowing for real-time object recognition. To further enhance the system's capabilities, we have also implemented an extension to detect when an unknown object, not present in the object database, is present in the video stream or provided as a single image. The results of our classification are demonstrated through the inclusion of a result image for each category of object in our report, with the assigned label clearly shown.



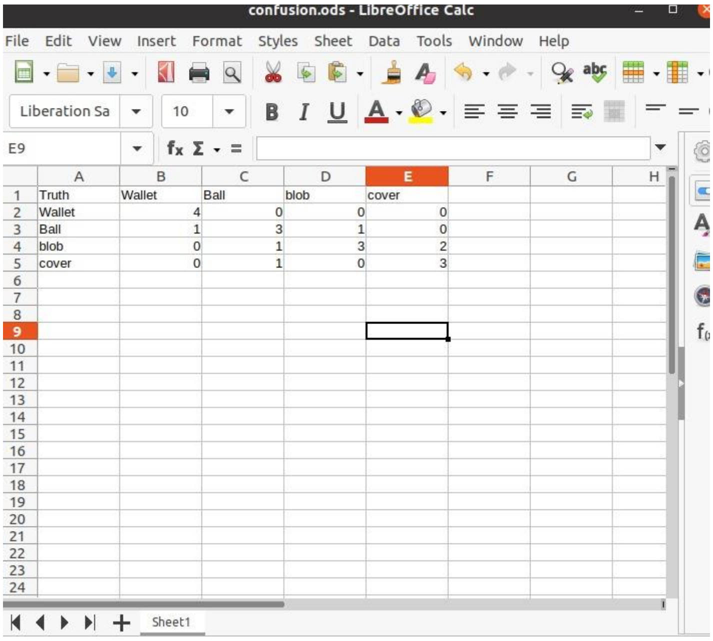
Classified Image 1.



Classified Image 2.

7. Evaluating the performance of the system:

To evaluate the performance of our system, we evaluated its ability to correctly classify various objects across different positions and orientations. This involved testing on multiple unique images of each object and constructing a **5x5 confusion matrix** summarizing the results. The matrix compares true object labels with the system's classified labels, providing insights into classification accuracy and potential misclassifications. The confusion matrix will be included in the report, along with an explanation of its interpretation and the overall performance evaluation of the system.



Truth	Wallet	Ball	blob	cover
Wallet	4	0	0	0
Ball	1	3	1	0
blob	0	1	3	2
cover	0	1	0	3

TRUTH	WALLET	BALL	BLOB	COVER
WALLET	4	0	0	0
BALL	1	3	1	0
BLOB	0	1	3	2
COVER	0	1	0	3

Percentage accuracy for each object is as below:

WALLET	100%
BALL	60%
BLOB	50%
COVER	75%

8. Capturing a demo of the system working:

To demonstrate the functionality of our system, we captured a video showcasing its real-time object recognition capabilities. This video, is included in the report via a link, provides a visual representation of the system processing live video frames, extracting features, and classifying the identified objects.

Link: 2024-03-02 22-40-33.mkv

Please log-in from your northeastern ID to view this video on OneDrive.

9. **<u>Implementing a second classification method:</u>**

For the second classification method in the Object Recognition (OR) system, a pre-trained deep network is utilized to generate embedding vectors for the thresholded objects, followed by nearest-neighbor matching using cosine distance as the distance metric. This approach harnesses the capabilities of deep learning to extract complex object features and create compact representations for efficient comparison. By measuring similarity through cosine distance, the system can effectively match objects based on their learned features. In comparison to traditional classifiers like nearest neighbor, this deep learning-based method is expected to enhance recognition accuracy by capturing intricate object characteristics. The report will detail the implementation process, compare performance metrics with the baseline system, and highlight the advantages of incorporating deep learning techniques for object classification in the OR system.

Learning Outcomes:

This project provided valuable experience in the fundamentals of 2D object recognition. We successfully implemented a system that can segment and classify objects in images and videos, achieving translation, scale, and rotation invariance.

<u>Image Processing Techniques:</u> We gained hands-on experience with essential image processing techniques, including thresholding, morphological filtering, connected components analysis (CCA), and feature extraction.

<u>Machine Learning Fundamentals:</u> The project allowed us to explore basic machine learning concepts through implementing a nearest-neighbour classifier and evaluating its performance.

The project emphasized the importance of considering real-world constraints, using high-quality training data, and evaluating performance objectively. This practical application of theoretical concepts provided valuable insights into the complexities and potential of 2D object recognition.

Acknowledgement:

We extend our heartfelt gratitude to the following platforms for their invaluable support and resources throughout the Real-time 2-D Object Recognition project:

- Springer
- ResearchGate
- GitHub
- Stack Overflow
- ChatGPT
- Gemini

The insights and references obtained from these platforms have significantly enriched our understanding and greatly contributed to the successful development and implementation of various aspects of the project. Their assistance has been indispensable in ensuring a comprehensive and well-informed approach to the tasks at hand. We are truly thankful for the wealth of information and guidance provided by these platforms, which have been essential in our journey towards excellence in Real-time 2-D Object Recognition.