**Object Segmentation and Comparison**

Team member: Xin Feng(82509956), Yitong Liu(50842989)

**Introduction:**

The field of digital image processing continues to evolve rapidly, offering significant contributions to a plethora of industries including security, healthcare, autonomous vehicles, and social media, among others. Central to many of these applications is the task of object detection and identification in a set of images. It involves not only the isolation of individual objects within images, but also the comparison of these objects to a set of reference objects for accurate identification. The primary objective of this project is to develop a systematic approach that performs object segmentation on a collection of images, calculates similarity with reference images, and effectively identifies whether a specific object is present within these images.

To achieve this goal, we intend to initially focus on the development of a robust image segmentation function. Image segmentation forms the basis of our object detection methodology, partitioning an image into multiple segments, each representing an individual object within the image. This crucial step aids in the extraction of critical features from each object, enabling further processing and comparison.

Following the image segmentation process, our focus will shift towards the comparison of segmented objects with a set of pre-defined reference images. The comparison will hinge on the calculation of image similarity metrics, which quantify the likeness between the segmented objects and the reference images based on various features such as color, texture, shape, and size.

Finally, we aim to pinpoint the presence of specific objects within the image collection. By cross-referencing the similarity metrics calculated against each reference image, we intend to successfully identify and highlight the presence of the target objects within the image collection.

Through this project, we aspire to push the boundaries of object detection methodologies, creating a system capable of efficient and accurate identification of objects across a broad collection of images. The insights and technologies arising from this project have the potential to further the scope and effectiveness of various applications that rely heavily on advanced image processing techniques.

**Technical approach**

1. Object Segmentation

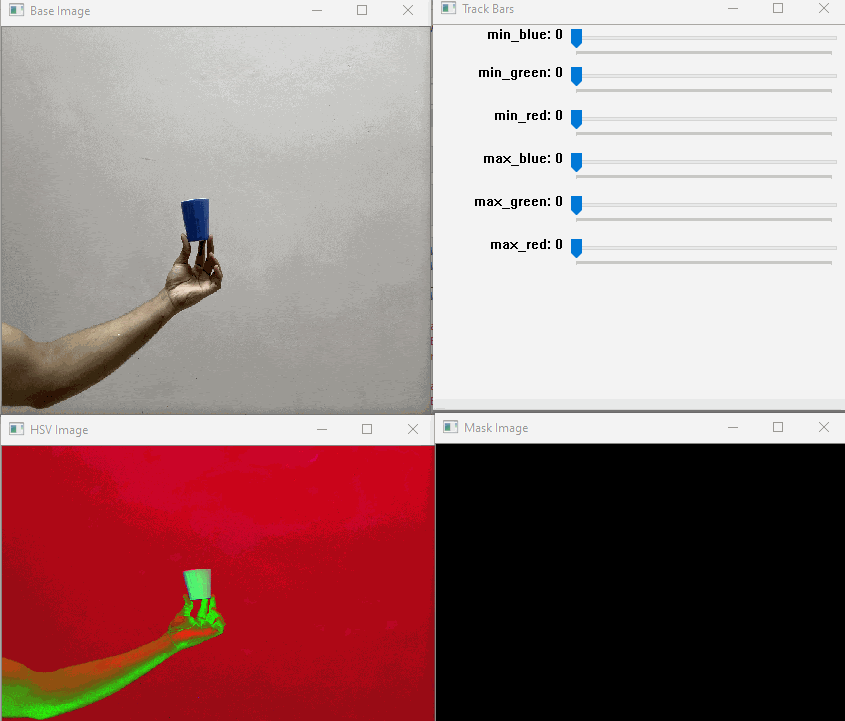
In addressing the challenge of object detection in images, our technical approach is predominantly centered around the use of Hue, Saturation, and Value (HSV) Color Segmentation. This technique forms the core of our image segmentation process, and it has been selected for its robustness in differentiating objects based on color, irrespective of lighting conditions. This choice is pivotal, as traditional methods such as Edge-Based Segmentation often falter when confronted with complex elements such as shadows and table edges, resulting in inaccurate object identification.

HSV Color Segmentation operates by converting the color representation of an image from the standard Red, Green, Blue (RGB) model to the HSV model. This alternative color model has three components: Hue, which represents the color type; Saturation, which indicates the vibrancy of the color; and Value, which signifies the brightness of the color. One of the significant advantages of this color model is its ability to separate color information from intensity or lighting, thereby making it less susceptible to variations in illumination which can greatly impact the RGB model.

Once the images are represented in the HSV model, we can construct a histogram or apply thresholding rules using only the saturation and hue components. This step essentially forms a color 'fingerprint' of the objects within the image. By reducing the influence of the value component, we effectively minimize the impact of shadows or other lighting conditions on the segmentation process, allowing the true colors of the objects to guide the segmentation.

However, while HSV Color Segmentation is generally reliable, it can still face challenges with images under different scenes or varying lighting conditions. To manage this, we will incorporate Threshold-Based Segmentation for images that fall under similar scenes. This approach sets specific intensity thresholds to separate the objects from the background, further refining the object detection process.

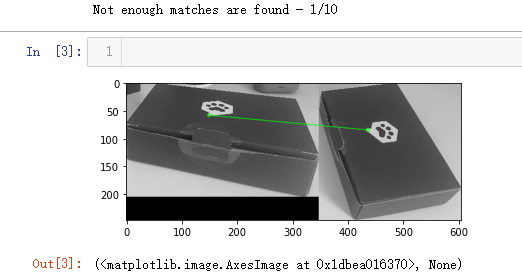
Through the combined use of HSV Color Segmentation and Threshold-Based Segmentation, we aim to create an effective image segmentation process that can robustly and accurately identify objects within a collection of images. This innovative approach is designed to overcome the limitations of traditional segmentation techniques and offer a new level of precision in object detection.



1. Object Comparison

In the next phase of our project, we ventured into the creation of a dedicated library aimed at storing potential objects extracted from the images. In the next phase of our project, we established a library to store images of potential objects. This library serves as a reservoir of reference images that we will compare against the segmented objects from our target images.The underlying concept was to accumulate a comprehensive database of object images segmented from the original sources. These images serve as a pool of target pictures that we can use for comparison with our test images.

Initially, we attempted to perform this comparison using feature matching, a common technique in computer vision that we practiced in this class that involves comparing key features between images. While theoretically sound, this method proved less than ideal in practice. Feature matching often struggles with variations in size, perspective, and other transformations that frequently occur across different images. Given the nature of our image collection, these variances were common, thereby limiting the effectiveness of feature matching.



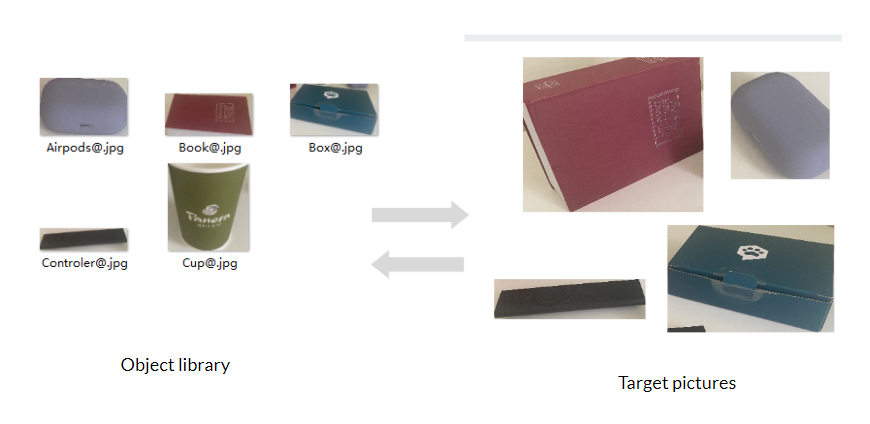
To tackle this issue, we decided to employ dense vector representations, a technique widely used in machine learning and computer vision tasks.

Dense vector representations, unlike sparse representations, use fewer dimensions to capture the information in an object while still preserving the vital characteristics required for comparison. Each dimension in a dense vector encapsulates some aspect of the object's information, hence making it a suitable technique to handle variations in image attributes like size and transformation.

For the implementation of dense vector representations for our images, we turned to the 'sentence-transformers' library, which, despite its name, can be effectively utilized for image processing tasks as well. This library provided a convenient and efficient method for calculating dense vector representations for our images.

Moreover, we integrated the OpenAI Contrastive Language–Image Pre-training (CLIP) Model into our process. The CLIP Model is a powerful neural network that has been pre-trained on a diverse set of image-text pairs, enabling it to understand and relate visual and textual information effectively.

By combining the use of the 'sentence-transformers' library for generating dense vector representations and the CLIP Model for leveraging its pre-training on multimodal data, we were able to overcome the challenges posed by image variations. This synergy enabled us to create a robust comparison framework capable of accurately identifying whether different images depict the same object, thus significantly enhancing the effectiveness of our object detection approach.

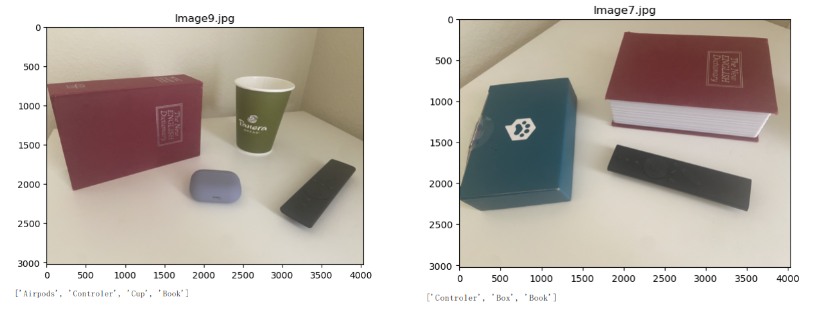


**Results**

The effectiveness of our technical approach was put to the test using a set of 11 images that we captured ourselves. These images served as the perfect testing ground to assess the robustness and accuracy of our object detection system. The images encompassed a diverse range of objects, lighting conditions, and positions, providing a comprehensive assessment of the system's capabilities.

We're delighted to report that our system successfully detected all objects in each of the 11 test images. This result unequivocally demonstrates the efficacy of our approach combining HSV Color Segmentation, Threshold-Based Segmentation, dense vector representations, and the use of the sentence-transformers library and OpenAI's CLIP Model.

These results underscore the potential of our innovative approach to offer a reliable, efficient, and highly accurate system for object detection within images. Further, it proves the robustness of the system to handle variations in lighting, size, perspective, and other common challenges encountered in the realm of image processing. We anticipate that these findings will provide a strong foundation for future research and applications in the field of image processing and object detection.



**Discussion and Future Directions:**

The project we undertook involved the implementation of object detection on a collection of images, using a combination of color segmentation and feature matching. Our system successfully identified whether a specific object present in a reference image also appeared in other images in our set. Despite the success of the project, we encountered some challenges. For instance, we initially attempted to use feature matching for image comparison, but due to the variety in size and transformations of objects across different images, this approach proved ineffective. Consequently, we opted for the utilization of dense vector representations, an approach that yielded promising results.

Throughout the implementation process, we noted several areas that would benefit from further improvement and refinement. One key area is the expansion of our object library. Currently, our library has a limited number of objects, which narrows the scope of object detection our model can perform. Increasing the size of the object library would allow the model to detect a broader range of objects, improving its utility and versatility.

In future work, we could focus on methods to automate the expansion of the object library. This could involve leveraging machine learning techniques to learn and store new objects from a continuously updated dataset. Additionally, future improvements could include refining the process of determining object matches to accommodate more complex transformations and variations between images.

Another future direction could be exploring how to enhance the model's efficiency and performance when handling large-scale image datasets. We could look into optimization techniques such as parallel processing or batch processing of images to speed up the object detection process.

Lastly, we could consider developing a user interface for the model, which would allow users without coding knowledge to easily use the object detection system. This would make the tool more accessible and could open up potential applications in fields like image editing, content

**References**

1. Code reference for Object Segmentation

<https://stackoverflow.com/questions/69295962/drawing-bbox-for-realtime-object-detection-without-ml/69307887#69307887>

<https://medium.com/globant/maneuvering-color-mask-into-object-detection-fce61bf891d1>

This part was mostly done by my partner, we added the while loop that continually decreases the value of blue within the segmentation process, which helps prevent the scope of the detected object from becoming overly broad. This modification was instrumental in refining our object detection mechanism, ensuring it could accurately isolate the intended objects in various images.

2. Code reference for Object comparison

<https://stackoverflow.com/questions/11541154/checking-images-for-similarity-with-opencv>

This source code loads the OpenAI CLIP Model, encoding the images, processes the encoded images to find pairs of images with the highest cosine similarity score. And then identify the Near-Duplicate images.

We remodify it:

Addition of a function to calculate histograms and compare color similarity.

The implementation of a function to get the most similar image.

Classification of the images based on a given threshold and storing them in a dictionary.

Visualizing the images and printing the results.