Real-time 2-D Calibration and Augmented Reality

Yuan Zhao

CS5330 40208: Pattern Recognition and Computer Vision, Project 4

March 19th, 2024

# Project Overview

The project, created for the CS5330 Computer Vision course, centers on Calibration and Augmented Reality, pertains to the development and implementation of camera calibration techniques, as well as the subsequent deployment of these techniques for the integration of virtual objects into a live video feed. The essence of the project revolves around crafting an application capable of identifying a predefined target within the video stream and accurately overlaying a virtual object in relation to this target. This process must maintain precision irrespective of any movements or alterations in the orientation of either the camera or the target.

# Functionality Demonstrations

The project is structured into three critical components:

1. **Calibration**: Calibration involves the creation of an application designed to recognize a chessboard within the video stream. This application should be capable of extracting and highlighting the corners of the chessboard. Additionally, it allows for the selection of specific frames to be utilized in the calibration process of the camera. The culmination of this phase is the generation and storage of the camera matrix and distortion coefficients into a persistent storage medium.
2. **Virtual object**: Showing virtual object process advances the project by requiring the development of a software solution that not only detects the chessboard within the video feed but also computes the pose of the board based on the corner locations identified in the previous phase, in conjunction with the calibration data derived from the camera. The key deliverable for this phase is the integration of a virtual object into the video stream, positioned in relation to the chessboard. The virtual object must maintain its correct position and orientation, even as the camera's perspective shifts.
3. **Robust Features detections**: This component shifts the focus towards enhancing the program's capability to identify robust features within the video stream, such as Harris corners, SIFT (Scale-Invariant Feature Transform), or SURF (Speeded Up Robust Features). The program developed in this phase should be able to visually represent the location of these features in real-time, as the video is being streamed.

# Experiment and data Analysis.

## Detect and Extract Target Corners

A black and white checkered board

Description automatically generated

A black and white checkered board

Description automatically generated

The augmented reality framework commences its operation through the identification of chessboard corners within the visual feed. Leveraging the capabilities of OpenCV's *findChessboardCorners* and *cornerSubPix* functions, the system is engineered to pinpoint and enhance the precision of corner positions to a sub-pixel level. This process facilitates the visual depiction of the detected corners within the imagery. Furthermore, for purposes of debugging and verification, the coordinates corresponding to the initial corner, which is generally situated at the top-left corner of the chessboard, are systematically extracted and presented.

## Calibration Images.

A screenshot of a computer

Description automatically generated

A black and white checkered board

Description automatically generated

A black and white checkered pattern on a white surface

Description automatically generated

A black and white checkered board

Description automatically generated

A black and white checkered board

Description automatically generated

A black and white checkered pattern

Description automatically generated

A black and white checkered board

Description automatically generated

This project involves calibrating a camera using six chessboard images taken from different angles. Users can select an image for calibration with the 's' key, capturing 2D image coordinates and corresponding 3D world points for alignment within a uniform coordinate system. The calibration process hinges on these point pairs to compute the camera's intrinsic and extrinsic parameters, essential for accurate 3D visualizations in augmented reality. This foundational work aims to precisely blend virtual elements into real-world environments, enriching the augmented reality experience.

### Camera Calibration and Error Estimation

A screen shot of a computer

Description automatically generated

Upon successful identification of the board pattern, the project employs the *cv::calibrateCamera* function to execute the calibration of the camera. This calibration phase involves a methodical refinement of both the camera matrix and the distortion coefficients. The objective of this refinement is to diminish the re-projection error to a minimal range, typically observed between 0.2 and 0.3. This range is subject to variation based on several external conditions, including the ambient lighting and the degree of flatness exhibited by the board surface. This project also increases the number of calibration images to assess whether a higher quantity of images could lead to reduced errors. The findings showed that using 5 images versus 9 images resulted in almost identical errors. Following the completion of the calibration process, the user is presented with the option to store the calibration parameters. By initiating the 'c' command, the calibration data is securely saved to the designated directory at /bin/calibration.csv (later changed into /data/calibration.csv), thereby ensuring the preservation of the camera's calibration parameters for future use.

1. Current Position of the Camera

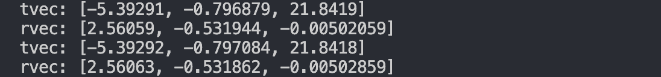
In this segment of the project, the camera moving and create the different sets of rotation and translation parameters. Here are the screen shot of the chessboard with different angels and the results of the p

A black and white checkered board

Description automatically generated

A black and white checkered board

Description automatically generated



A number on a black background

Description automatically generated

two types of recognition tests were conducted. Initially, the program was tasked with recognizing 8 objects, using only one training dataset for each. This test was successful in identifying all 8 objects. However, challenges arose when the objects bore close resemblance to one another in terms of size and shape, leading to recognition errors.

Subsequently, the project focused on three objects: a straw, glasses, and a clipper. For each, 8 different photographs were taken and analyzed in a live video setting, with the training data saved to a .csv file. By incorporating 7 additional training datasets featuring various orientations, the recognition accuracy approached nearly 100%. This high success rate was consistent regardless of whether the objects were isolated or positioned among other items, effectively eliminating recognition errors.

First set of recognition.

The image below is displayed with random region IDs and unlabeled objects. Users can utilize the 'N' key and command line to change the labels and assign different names to these objects for future recognition.A close-up of different colored objects

Description automatically generated

After promptly inputting all object labels, the project begins to classify objects based on the objectDB.csv, achieving perfect recognition initially.

A close-up of different objects

Description automatically generated

However, altering the orientation of the objects introduces some recognition errors. For instance, a spoon may be misidentified as a ballpen, while clippers and a bracelet could be mistakenly recognized as a coin.

A computer screen shot of various objects

Description automatically generated

A green circle with a black background

Description automatically generatedA yellow logo on a black background

Description automatically generatedA green and black background

Description automatically generated with medium confidenceA close-up of a pair of sunglasses

Description automatically generatedA blue and green line on a black background

Description automatically generatedA blue and black spoon

Description automatically generated

Second set of recognition.

A close-up of a pair of sunglasses

Description automatically generated

A close-up of a black background

Description automatically generated

A blue and white object on a black background

Description automatically generated

A pink cat on a black background

Description automatically generated

A green rectangle with black background

Description automatically generated A screen shot of a computer

Description automatically generated

A blue mouse in the sky

Description automatically generated with medium confidence

A pink rectangle with a green arrow

Description automatically generated

The images above pertain to a screenshot of sunglasses undergoing the feature extraction process, which is based on 8 different orientations of the objects. This same process has been applied to both the straw and the clipper.

The results are more accurate.

## Evaluate the performance of your system

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | >real object | glasses | clipper | straw | pen | spoon | braclet |
| resuls from compute |  |  |  |  |  |  |  |
| glasses |  | 10 |  |  |  |  |  |
| clipper |  |  | 10 |  |  |  | 4 |
| straw |  |  |  | 10 |  |  |  |
| pen |  |  |  |  | 5 |  |  |
| spoon |  |  |  |  | 5 | 7 |  |
| braclet |  |  |  |  |  | 3 | 6 |

of feature extraction and matching design. For the images, the upper section predominantly features the blue sky, while the lower section is characterized by darker elements such as trees, grass, or shadows. It conforms to the top-bottom distribution concept inherent in the images.

1. Video demo

Link: <https://drive.google.com/file/d/1jM_PgVTPZHYQJVXBxExOj3yt8gHMWUUf/view?usp=sharing>

## Second classification method

This component of the project was not implemented due to the extreme workload demands from a full-time job in real life.

# Conclusion

Throughout this project, we gained significant insights and acquired new skills, including the art of thresholding and choosing the correct thresholds, applying morphological operations, and familiarizing ourselves with OpenCV functions such as Moments and HuMoments. We delved into segmentation and feature computation, learned how to generate a confusion matrix, and explored new distance metrics like the scaled Euclidean distance.