Development of a Mobile Application for Image Processing from the ESP32-CAM-MB Board with Pixel Manipulation Effects and Advanced Filters.

Introduction

In today's world, real-time image and video processing is a fundamental tool in various applications, from security to entertainment. With advancements in technology and the miniaturization of devices, it is possible to develop efficient and accessible solutions. This report presents the development of a mobile application that processes images obtained from the ESP32-CAM-MB board, which includes the OV2640 mini camera. The application connects to a video streaming server deployed on the baseboard, from which it captures and processes the video in real time. Through this app, the video is transformed by applying a filter that employs quantization, resolution reduction, and color manipulation techniques, which simplify the original image before converting it to ASCII. The result will be an image modified according to the effects created for this project; each pixel of the image is represented by its original color, with a unique effect while implementing a retro style, providing a high level of flexibility and customization in image processing.

Problem Description

The growing demand for real-time image processing on mobile devices has driven the need for solutions that are both cost-effective and efficient. The ESP32-CAM-MB board, equipped with the OV2640 mini camera, represents a viable option due to its low cost and compact size. However, integrating this technology with mobile devices for advanced image processing presents several technical challenges that must be addressed:

- **Efficient Video Transmission:** Establish a streaming server on the board to transmit the video to the mobile device.
- **Real-Time Processing:** Apply advanced filters and manipulations without significant delays. **Implementation of Complex Algorithms:** Include operations such as quantization, convolution, histogram equalization, and morphological operations.
- **Integration into a Mobile Application:** Develop a user-friendly interface that allows the user to interact with the system and visualize the results.

Proposed Solution

To address the aforementioned challenges, the proposal was to develop a mobile application in Android Studio that meets the following features:

1. **Streaming Server on the ESP32-CAM-MB:** A video streaming server was deployed on the baseboard using the MJPEG (Motion JPEG) protocol to efficiently transmit video to the mobile device.
2. **Reception and Visualization of the Original Video:** In the mobile application, an MjpegView was implemented, which is a library that allows receiving and displaying the real-time video stream from the OV2640 camera.
3. **Image Processing in C++:** A native library in C++ was used within Android to perform advanced processing of the captured images. This allows for superior performance in computationally intensive operations.
4. **Application of Advanced Filters and Effects:** Specific functions were developed to manipulate the images:
    - Quantization Function: Reduces the grayscale levels in the image to simplify it and prepare it for ASCII conversion. Mathematically, quantization is performed by assigning each pixel a new value according to:

$$I_{\text{cuantizado}}(x, y) = \left\lfloor \frac{I(x, y) \times \text{LEVEL}}{256} \right\rfloor \times \left(\frac{256}{\text{LEVEL}} \right)$$

Where I(x, y) is the original intensity of the pixel, and LEVEL is the desired number of levels (in this case, 10).

    - **ASCII Function**: Converts 8x8 pixel blocks into corresponding ASCII characters, creating a text-based visual representation.
    - **convAscii Function**: Orchestrates the entire conversion process.

a) **Conversion to HLS**: The image is transformed from BGR to HLS to work with luminance.

b) **Downscaling and Upscaling**: The resolution is reduced to remove fine details and then upscaled again.

c) **Desaturation**: Saturation and hue are removed, focusing on luminance.

d) **Quantization**: The previously mentioned quantization function is applied.

e) **Conversion to ASCII**: The ASCII function is used to obtain the final image.

f) **Image Combination**: A bitwise AND operation is used to combine the ASCII image with the original colors. The bitwise AND operation is defined as:

$$\text{Resultado}(x, y) = \text{ImagenASCII}(x, y) \wedge \text{ImagenColor}(x, y)$$

1. **Graphical User Interface**: The application features an intuitive interface:
    - **Upper ImageView**: Displays the video processed with the applied filter.
    - **Camera Activation Button**: Allows starting and stopping the streaming from the ESP32-CAM-MB.
    - **Lower MjpegView**: Displays the original video received from the mini camera.
2. **MjpegView Implementation**: The MjpegView library was integrated to receive the MJPEG streaming. This library handles the decoding and visualization of JPEG image sequences transmitted via HTTP, facilitating the integration of streaming into the mobile application.

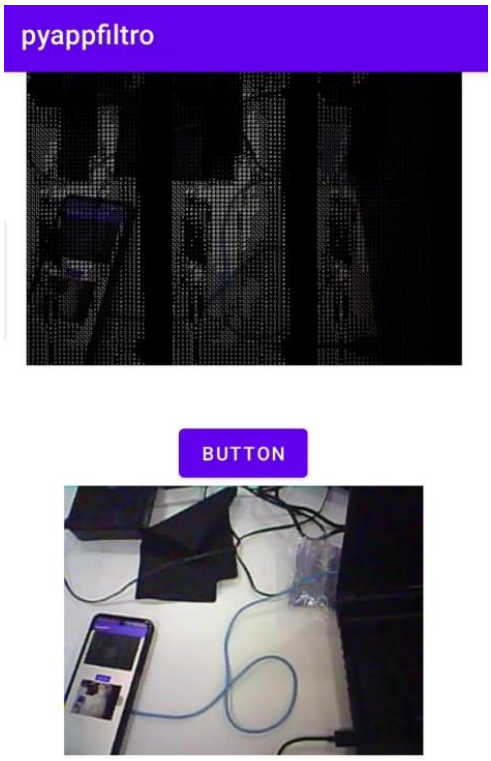

Figure 1

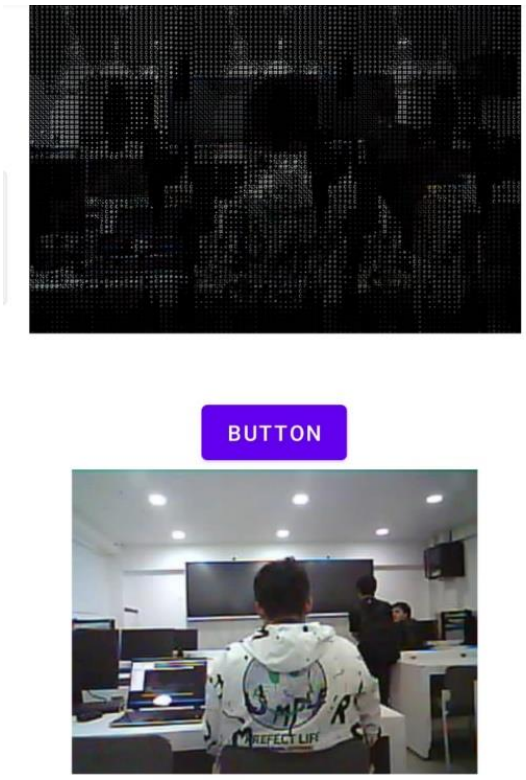

Figure 2

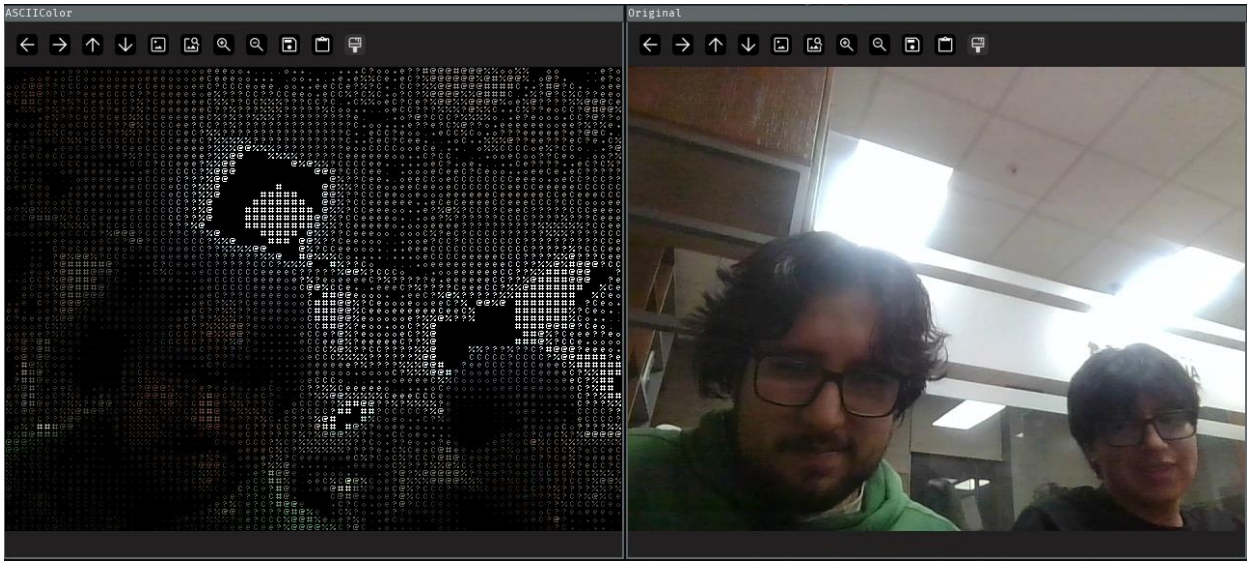

Figure 3

Conclusions

The project successfully concluded with the development of a mobile application capable of real-time processing of images captured by the OV2640 mini camera connected to the ESP32-CAM-MB. By implementing a streaming server on the board and using efficient libraries in C++, it was possible to apply advanced pixel manipulation effects and filters, meeting the established requirements. The use of techniques such as quantization, ASCII conversion, and bitwise operations allowed for the exploration of new methods of visual representation, combining image processing with artistic creativity.

Furthermore, the effective integration between hardware and software demonstrated the feasibility of using low-cost devices for advanced image processing applications in mobile environments. This achievement not only meets the proposed objectives but also lays the foundation for future improvements and expansions, such as the incorporation of artificial intelligence algorithms for pattern recognition or further performance optimization. The project highlights the importance of innovation and adaptability in technological development, opening doors to new opportunities in areas such as education, entertainment, and security.

References

- González, R. C., & Woods, R. E. (2018). *Digital Image Processing*. Pearson.
- OpenCV Library. (2023). *Open Source Computer Vision*. Recuperado de https://opencv.org/
- Espressif Systems. (2021). *ESP32-CAM Module*. Recuperado de https://www.espressif.com/en/products/socs/esp32
- Android Developers. (2023). *NDK Guides*. Recuperado de https://developer.android.com/ndk/guides
- Smith, S. W. (1997). *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing.
- Fanning, D. W. (2011). *A Practical Introduction to Computer Vision with OpenCV*. Wiley.

- Patel, K. (2020). *Mastering OpenCV 4 with Python*. Packt Publishing.
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.
- Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer.