



QR Code Detection



← → 🔍 DSPML.com

# Real-Time QR Code Detection, Decoding, and Processing with URL Extraction

**Rostata - Malveda**



QR Code  
Detection

Group Members



← → 🔍 DSPML.com

# Developers



**Carl Louise Rostata**

CPE32



**Jermaine Pau Malveda**

CPE32



QR Code  
Detection

Group  
Members

Introduction

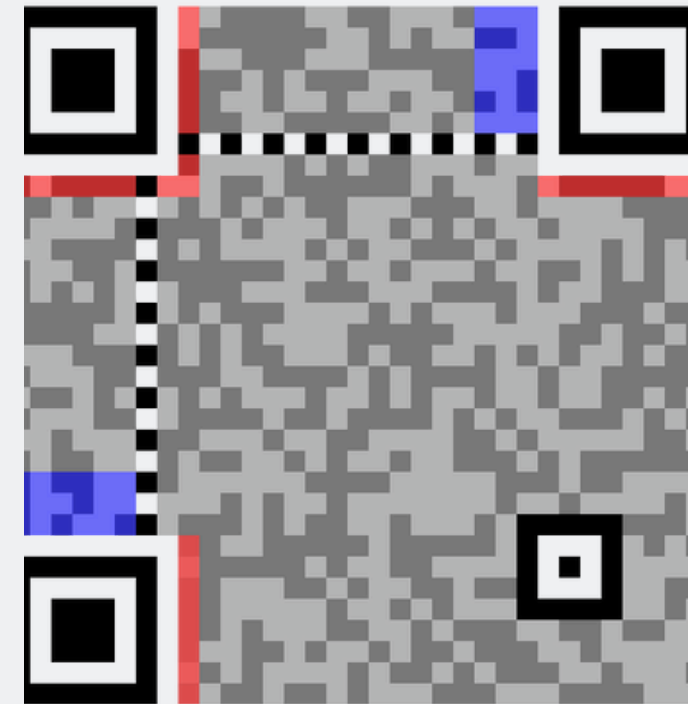


← → 🔍 DSPML.com

# Introduction

## QR Codes:

- Definition: Image and information encryption/decryption.
- Advantages: Fast readability, ample storage.
- Features: Position detection patterns, error correction (Mehta, 2021).



- 1. Version Information
- 2. Format Information
- 3. Data and Error Correction Keys
- 4. Required Patterns:
  - 4.1. Position
  - 4.2. Alignment
  - 4.3. Timing



QR Code  
Detection

Group  
Members

Introduction



← → 🔍 DSPML.com

# Introduction

## Alpha Blending:

- Importance: Key in image processing.
- Objective: Integrate light/transmittance matrices while preserving image edges.
- Method: Foreground overlays background seamlessly (GeeksforGeeks, 2021; Lian, 2023).

## Decoding QR Codes:

- **Process:**
  - Analyze edge directions for distortion.
  - Algorithms: Pattern recognition, error correction, data extraction (Karrach, 2020).



QR Code  
Detection

Group  
Members

Introduction

Objectives



← → 🔍 DSPML.com

# Objectives

The specific objectives of the project are:

- To develop a program that can detect and decode QR codes from real-time video capture source
- To process decoded QR code data to extract information (e.g. Text, URL)
- To develop a functionality to extract URLs from decoded QR codes and redirect users to the web.
- To implement Alpha blending and Pyzbar decoding in real-time video processing.



QR Code  
Detection

Group  
Members

Introduction

Objectives

Practical Application ✕ +

← → 🔍 DSPML.com



# Practical Application

Versatile tools optimizing information dissemination and transactional efficiency across diverse areas.

- Retail Sector
- Connectivity
- Education
- Financial Transactions



# Methodology

## Frame Preprocessing

### Preprocessing Pipeline for QR Code Detection:

- **Grayscale Conversion:**
  - Utilized `cv2.cvtColor()` function.
  - Transformed RGB color space into single-channel grayscale.
  - Purpose: Reduce computational overhead, enhance QR code visibility by eliminating color variations.
- **Gaussian Blurring:**
  - Applied to grayscale frames using `cv2.GaussianBlur()` function.
  - Objective: Reduce image noise, enhance clarity of QR code structures.

```
def preprocess_frame(frame):  
    # Convert frame to grayscale  
    # We used cv2.COLOR to perform a conversion of the frame from BRG to gray (same with colors' rgb2gray function)  
    # This is to simplify the QR code image/frame to reduce processing task  
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)  
  
    # Apply Gaussian blur for noise reduction  
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)  
  
    return blurred
```



Gray Image



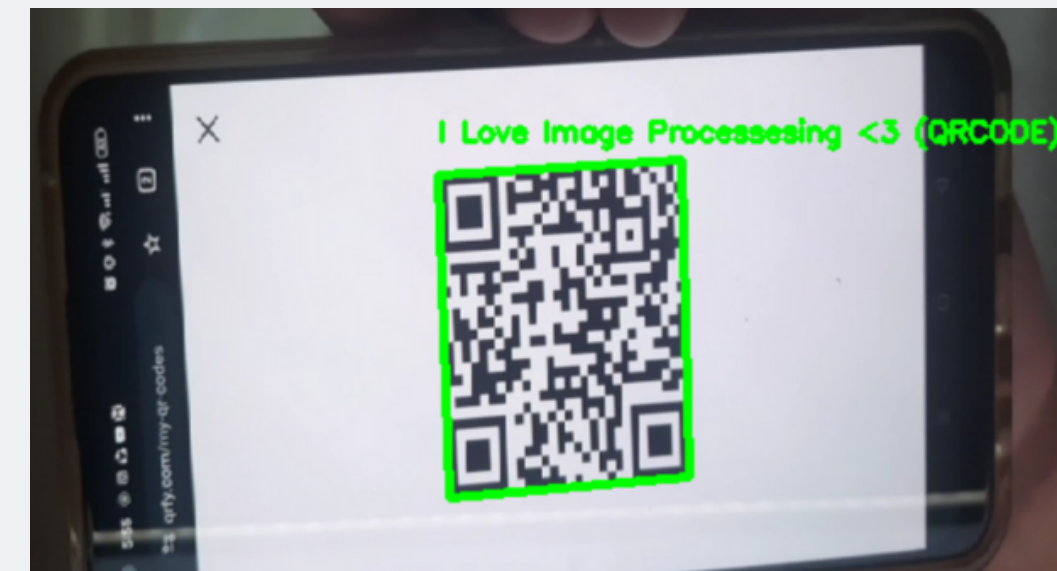
Gray image with Gaussian Blur





# QR Code Detection and Decoding

- **Utilization of pyzbar Library:**
  - Python interface for ZBar library, renowned for barcode and QR code reading.
- **Core Component: decode() Function:**
  - Purpose: Decoding barcode and QR code data from images.
  - Algorithms:
    - **QR Code Detection Algorithm:**
      - Techniques: Image scanning, pattern recognition.
      - Detects distinctive features indicative of QR codes, enabling accurate localization within the image.
    - **Decoding Algorithm:**
      - Extracts encoded data from detected QR code.
      - Process: Interprets arrangement of black and white modules, adheres to QR code standard for accurate data retrieval.
- **Data Representation:**
  - **Returned decoded objects encapsulate:**
    - Decoded data content.
    - Data type (e.g., text, URL).
    - Positional information for subsequent processing and utilization.



Gray image with Gaussian Blur





# Alpha Blending

- Alpha blending used for overlaying watermark onto video frames
- Technique combines pixel values of watermark with underlying video frames
- Incorporates transparency factor (alpha channel) to control blending degree
- Implemented using OpenCV
- Seamless integration of watermark onto video frames while preserving transparency

```
# Watermark using Alpha Blending technique
scan_height, scan_width, _ = scan_image.shape
# we retrieved the image dimensions using the shape attribute and stored them inside separate variables
start_x = int((frame.shape[1] - scan_width) / 2)
start_y = int((frame.shape[0] - scan_height) / 2)
# This lines computes the starting coordinates of the mask to ensure that it is centered within the set webcam frame
mask = scan_image[:, :, 3] / 255.0
# Here we are extracting the alpha channel of the scan_image to determine the transparency of each pixel.
for c in range(3):
    frame[start_y:start_y + scan_height, start_x:start_x + scan_width, c] = \
        scan_image[:, :, c] * mask + frame[start_y:start_y + scan_height, start_x:start_x + scan_width, c] * (
            1 - mask)
# This code applies the mask onto the frame using alpha blending.
# This iterates over each color channel RGB of the frame
# It basically calculates the blended pixel value based on the alpha mask and overlays the
# watermark onto the frame.
```

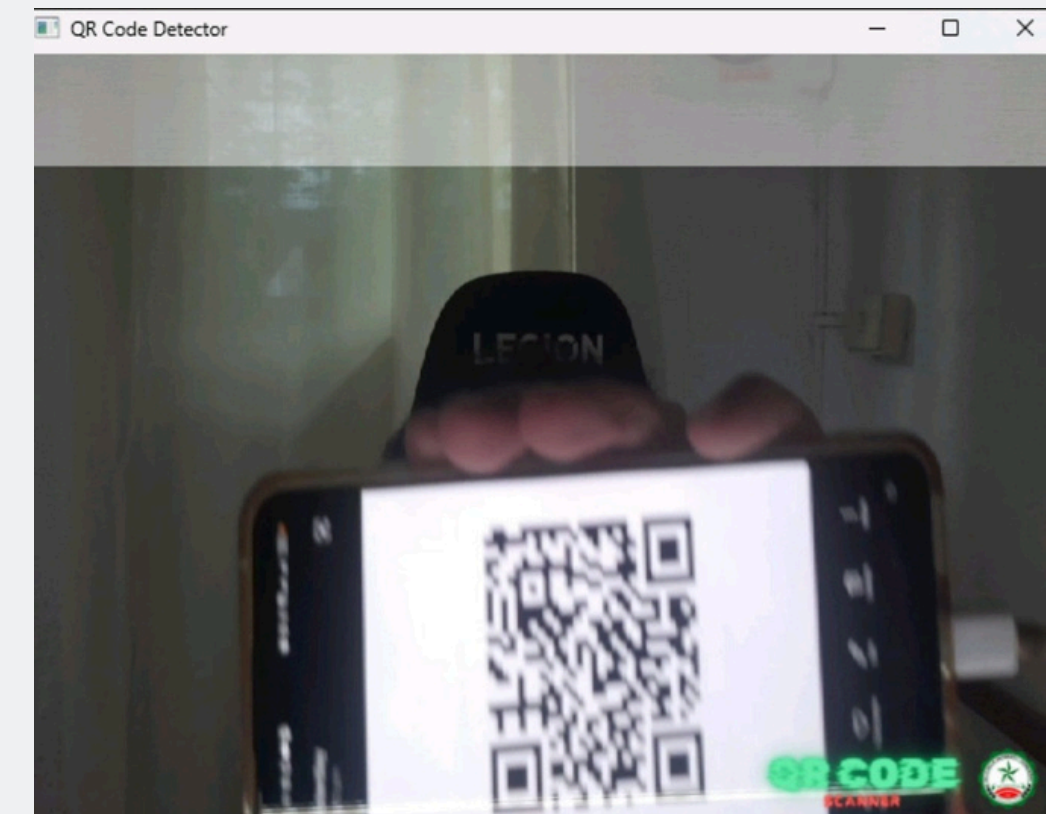


Alpha Blending Applied in Video Frame



# URL Extraction and Redirection

- QR code content extracted to identify embedded URLs
- Decoding process inspects data for "http://" or "https://" prefixes
- Users prompted to decide whether to access extracted links
- Utilization of webbrowser.open() function for URL redirection
- Redirection occurs upon user consent
- Facilitates direct access to online resources embedded in QR codes, enhancing user engagement



Video Frame when URL was scanned

```
#Extraction of URL and redirecting
# Check if decoded data is a URL, if it is, then a prompt will show up in the output log
if Qrcode_data.startswith("http://") or Qrcode_data.startswith("https://"):
    response = input("Do you want to open this link? (y/n): ")
    # if the response is 'y', then the webbrowser.open() function will redirect the QR Code link
    # to a tab in the browser. If 'n' then nothing happens, and the code proceeds to loop.
    if response.lower() == 'y':
        webbrowser.open(Qrcode_data)
```

```
[INFO] Found QRCODE QR Code: https://dlsud.edu20.org/
Do you want to open this link? (y/n):
```

Program Prompt



QR Code  
Detection

Group  
Members

Introduction

Objectives

Practical  
Applications

Analysis and Result



← → 🔍 DSPML.com



# ANALYSIS AND RESULT

- OpenCV efficiently detects single QR codes
- Pyzbar accurately decodes information from single QR codes
- Limitation: inability to simultaneously scan multiple QR codes
- OpenCV's capability extends beyond built-in camera to alternative sources like OBS
- Pyzbar decodes QR codes swiftly and accurately
- Integration of webbrowser module enhances user experience by redirecting users to decoded URLs, bridging QR code scanning and web interaction





QR Code  
Detection

Group  
Members

Introduction

Objectives

Practical  
Application

Analysis and Result



← → Q DSPML.com

# Documentation and Report

- OpenCV efficiently detects single QR codes
- Pyzbar accurately decodes information from single QR codes
- Limitation: inability to simultaneously scan multiple QR codes
- OpenCV's capability extends beyond built-in camera to alternative sources like OBS
- Pyzbar decodes QR codes swiftly and accurately
- Integration of webbrowser module enhances user experience by redirecting users to decoded URLs, bridging QR code scanning and web interaction





# Limitations and Future Work

- Limitations
  - Inability to perform simultaneous tasks
  - Frame freezes when scanning QR codes containing URLs because the program waits for a response before proceeding with the loop
  - Reliance solely on laptop camera output may not always be practical
- Suggestions for Future Research
  - Integration of functionality into a microcontroller unit (MCU) for versatility
  - Enhancement of functionality to include scanning and displaying images on-screen
  - Implementation of diverse redirection techniques for enhanced security





QR Code  
Detection

Group  
Members

Introduction

Objectives

Practical  
Application

Analysis  
and Result

Limitations  
and Future  
Work

Conclusion



← → 🔍 DSPML.com

# Conclusion



- OpenCV and Pyzbar are the backbone of the developed QR code scanning program.
- Limitations include the inability to scan two QR codes simultaneously.
- QR code detection and decoding effectiveness maintained with open-source libraries.
- Image processing techniques like alpha blending in OpenCV and preprocessing methods enhance understanding and application.
- Comprehensive toolsets greatly improve QR code scanner efficacy.



QR Code  
Detection

Group  
Members

Introduction

Objectives

Practical  
Application

Analysis  
and Result

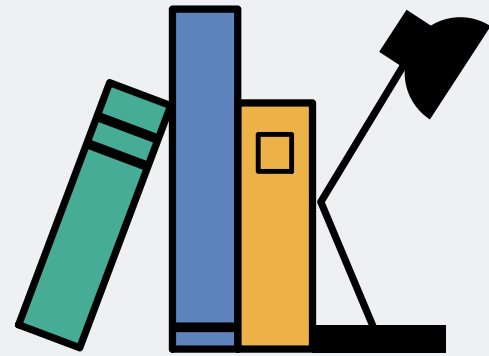
Limitations  
and Future  
Work

Conclusion

References



← → 🔍 DSPML.com



# References

- 01 Detect and read barcodes and QR codes with ZBar in Python | note.nkmk.me. (2022, October 15).  
<https://note.nkmk.me/en/python-pyqzbar-barcode-qr-code/>
- 02 Karrach, L., Pivarčiová, E., & Božek, P. (2020). Identification of QR Code Perspective Distortion Based on Edge Directions and Edge Projections Analysis. Journal of Imaging, 6(7), Article 7.  
<https://doi.org/10.3390/jimaging6070067>
- 03 Liang, T. (2023). Research on smoke removal of industrial scene image based on an improved dark channel prior. Third International Conference on Computer Vision and Pattern Analysis (ICCPA 2023), 12754, 234–243. <https://doi.org/10.1117/12.2684339>
- 04 Mehta, A., & Solanki, D. K. (2021). Design and Development of QR Code Recognition from Digital Image. International Journal of Engineering Research, 9(5).
- 05 [OpenCV - Alpha blending and masking of images. (2021, March 16). GeeksforGeeks.  
<https://www.geeksforgeeks.org/opencv-alpha-blending-and-masking-of-images/>