PROJECT REPORT
STATISTICAL METHODS I

CHROMA-KEYING

Presented To ARNAB CHAKRABORTY

Presented By

AMAN SINGH (BS2308)
SATYAPRAKASH SAHDEV JAISWAR (BS2342)
RONAK GUPTA (BS2339)
SAURABH BHIKAN NANKAR (BS2344)
SUBHADIP ROY (BS2353)
PALLAV SAHAY (BS2335)

ABSTRACT

- This project is about creating a new and better way to use chroma keying, a crucial technique in filmmaking, photography, and video production. We started by trying different methods to smoothly remove and replace specific-colored backgrounds.

- First, we explored the Min-Max Bound approach, but it had limitations. So, we looked into the Pixel Predominantly Green approach, focusing on the common use of green screens. However, this approach also had its issues.

- Undeterred, we combined the strengths of both approaches and added a new element called Grey Bound. This integrated approach became our final solution, overcoming the challenges faced by the previous methods.

- Through careful testing and adjusting, we developed a more robust and adaptable chroma keying solution. This journey represents our exploration, experimentation, and determination to create a comprehensive approach that shows promise for the future of visual media production.

INTRODUCTION

- In our project, we aim to create a highly efficient algorithm specifically designed for removing backgrounds in photos under certain conditions.

- Firstly, we conceptualize and design an algorithm that meets specific criteria for background removal. Then, we move on to the coding phase, where we carefully create a program that serves as the backbone of our project. This code allows us to seamlessly remove backgrounds from a variety of photos.

- As we run our code on different images, we actively identify and document any shortcomings in our algorithm. This iterative process is crucial, helping us systematically address and fix limitations, progressing toward an outcome that meets our high standards.

- Our project follows a continuous cycle of implementation, observation, and refinement, showcasing its dynamic nature. Through these iterative steps, we not only fine-tune the algorithm but also gain valuable insights into the complexities of background removal in various photographic scenarios.

APPROACH 1 :
Minimum and Maximum bound

- **A. Idea:**

Aim is to remove green screen backgrounds from images using a loop that goes through each pixel. If a pixel's RGB components fall within a specified “green” range, it gets replaced with a new background.

- **B. Algorithm:**

Image Retrieval: Use a function like 'readJPEG' to read the image.

Colour Channel Range Definition: Set minimum and maximum values for Red, Green, and Blue channels.

Original Image Plotting: Display the original image for reference.

Result Array Initialization: Create an array ('result') with zeros, matching the image dimensions.

Pixel Iteration: Use nested loops to go through each pixel.

Colour Channel Check: For each pixel, check if all three color channels are within the specified range.

Result Assignment: If the condition is met, set the corresponding pixel in the result array to white.

Preservation of Original Values: If not, keep the original pixel values in the result array.

ERRORS

- 1. **Tolerance Levels**: - Define adjustable thresholds for green, red, and blue components to accommodate diverse images and lighting conditions, enhancing the algorithm's versatility.
- 2. **Colour Variation Issues**: - RGB values for green can vary significantly due to lighting, green shades, and camera settings. A fixed threshold may not consider these variations, potentially leading to inaccurate removal or retention of green shades.
- 3. **Complex Green Shades**: - Green screens may have diverse shades due to lighting variations or imperfections. A fixed threshold may struggle to distinguish between different green shades, causing challenges in the removal process.
- 4. **Spill and Reflections**: - The approach may not effectively handle color spill or reflections from the green screen onto the subject. This could result in remnants of the green color on the subject, compromising the realism of the final composition. And if we try to remove those, we may get uneven edges around the foreground object in the resultant image.

INPUT

OUTPUT

INPUT

OUTPUT

APPROACH 2 :
Pixel Predominantly Green

- **A. Idea:**

Aim is to remove green screen backgrounds from images using a method that compares the green component of each pixel with the red and blue components. If the green component is higher than both red and blue, the pixel is considered part of the green screen and may be removed or replaced.

- **B. Algorithm:**

1. **Input**: Obtain pixel values (R, G, B) for each pixel.
2. **Comparison**: - Compare G against R and B for each pixel.
3. - If G > R and G > B, consider green dominance.
4. **Decision**: - Remove or replace pixel if green dominates.
    1. - Keep pixel unchanged if not.
5. **Output**: Modified image reflecting decisions made.

ERRORS

- 1. **Green Balance Variation**: - Green background balance may vary due to factors like lighting and color calibration. Allow flexibility in the algorithm to accommodate variations in the green background's precise green balance.
- 2. **Foreground Variations**: - Foreground subject colors can vary, and elements in the subject may have a higher green component than the background. Be cautious not to unintentionally remove parts of the subject, especially if it contains green-colored objects or green spill from the background.
- 3. **Limited Sensitivity to Variations:** - Solely comparing green, red, and blue components may not consider variations in lighting conditions, shadows, or color intensity. The approach might mistakenly remove pixels when lighting causes variations in color intensity on the green screen.
- 4. **Foreground Colour Similarities**: - If the foreground subject has green or similar colors, especially in clothing or certain objects, the methodology may erroneously remove these elements. This could lead to incomplete or distorted results.
- 5. **Edge Artifacts**: - The approach may introduce artifacts along the edges of the subject due to abrupt pixel removal based on color thresholds. This can result in a jagged or unnatural appearance at the boundary between the subject and the background.

INPUT

OUTPUT

INPUT

OUTPUT

APPROACH 3 :
The Final Approach

- **A. Idea:**

Aim is to focus on simulating RGB color transformation, specifically exploring pixels with a greenish hue. The approach involves examining each pixel individually, excluding those with a higher green component than red and blue. A user-configurable threshold enhances flexibility in filtering, allowing users to adjust elimination criteria. The project also incorporates a green upper bound to preserve light green tones. Special consideration is given to retaining grey pixels with a maximum green component and red/blue values close to green, ensuring precision in the pixel filtering process.

- **B. Algorithm:**

1. **Input:** Accept parameters: 'image' (RGB input image), 'threshold' and 'threshold2' (user-defined green component thresholds), and 'threshold3' (user-defined threshold for absolute differences).
2. **Initialization:** Create a 'result' array with the same dimensions as the input image to store the processed result.
3. **Pixel Iteration:** Iterate through each pixel in the image using nested loops.
4. **Extract RGB Values:** Extract the red ('r'), green ('g'), and blue ('b') components of the current pixel.

APPROACH 3 :
The Final Approach

5. **Condition Checking:** Check conditions for each pixel::
    - Green component is the maximum among RGB values.
    - Green component is within user-defined thresholds ('threshold' and 'threshold2').
    - Absolute differences between red and blue, red and green, and blue and green are greater than or equal to 'threshold3'
6. e. **Pixel Replacement:** If conditions are met, set the pixel in the 'result' array to white (full intensity); otherwise, retain the original pixel values
7. f. **Result Array:** The processed image is stored in the 'result' array.
8. **Output:** The 'process_image' function returns the processed image in the form of the 'result' array.

Shiny implementation:

In the project, a recurring challenge similar to Approach 1 has emerged— the necessity to modify the threshold. To overcome this, a Shiny app has been created, leveraging Shiny, a web application framework for R. This app facilitates dynamic adjustment of all three thresholds, enhancing the interactive aspects of the project.

ERRORS

- 1. **Edge Artifacts**: - Pixel-by-pixel iteration may lead to noticeable edge artifacts, especially in intricate or detailed subject outlines. This can result in jagged edges or halos around the composited subject, diminishing overall visual quality.
- 2. **Inconsistent Thresholding**: - User-configurable thresholds may lead to inconsistent results depending on the image content. Users might struggle to determine optimal thresholds for different scenes, potentially causing over-removal or insufficient removal, leading to color spill artifacts.
- 3. **Selective Exclusion Challenge**: - Selective exclusion of pixels based on the green component alone may not fully address variations in lighting conditions or shades of green. This could result in incomplete removal of the green screen, leaving residual artifacts and impacting the realism of the composited image.
- 4. **Handling of Grey Pixels:** - The consideration for retaining grey pixels with a maximum green component, accompanied by red and blue values close to green, may introduce complexities. Depending on specific shades involved, this approach may inadvertently result in the removal of pixels crucial for the subject's fidelity.

INPUT

OUTPUT

INPUT

OUTPUT

INPUT

OUTPUT 1

OUTPUT 2

GUIDE (using the software)

Step 1: Upload Your Image

- - Click on the "Select an Image" button to upload the image with a green screen background.

Shooting Images for Effective Green Screen Chroma Keying:

- - Ensure even and consistent lighting across the green screen to minimize shadows and color variations.
- - Invest in a high-quality green screen material with uniform color.
- - Position the subject at an adequate distance from the green screen to avoid color spill and maintain separation.
- - Stretch the green screen tightly to eliminate wrinkles or creases for a smoother keying process.
- - Choose attire that contrasts with the green screen color to prevent unintentional removal during keying.
- - Keep camera settings, lighting conditions, and green screen setup consistent for multiple shots within the same project.
- - Prevent light from the green screen spilling onto the subject to maintain clarity.
- - Capture images in high resolution to preserve details for accurate keying.

Step 2: Adjust Green Screen Thresholds

- - *Green Lower Bound (threshold):* Use the slider to set the minimum intensity of the green color for removal. Adjust until the green screen is removed without affecting the subject.

GUIDE (using the software)

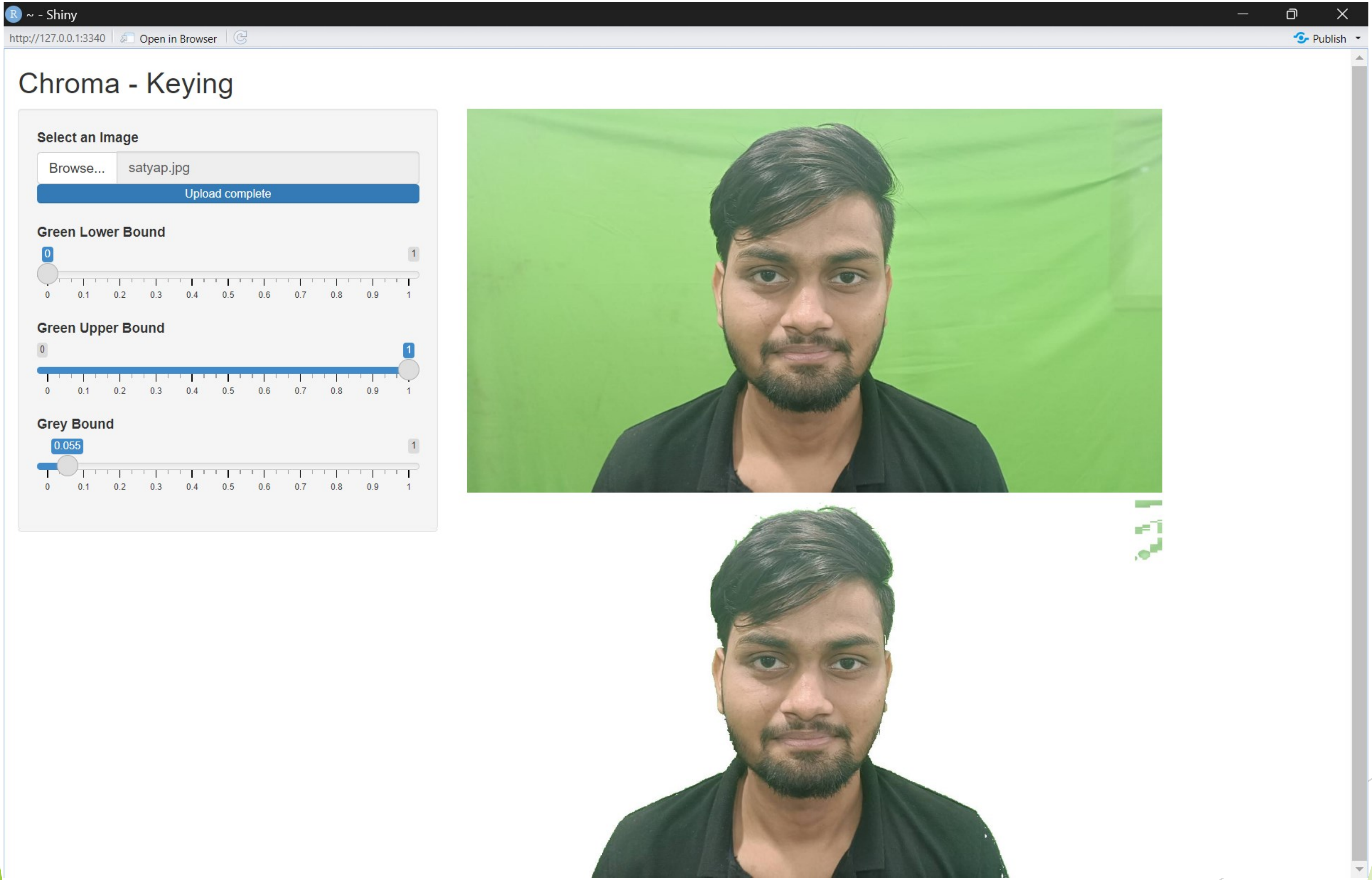
- - *Green Upper Bound (threshold2):* Utilize the slider to establish the upper limit of green intensity, preserving light green tones for a nuanced removal.
- - *Grey Bound (threshold3):* Adjust the slider to set the threshold for retaining greyish pixels with a maximum green component, crucial for preserving certain shades.

Step 3: Visualize Original and Processed Images

- - Observe the original image in the left panel under "Original Image.“ Witness the processed image in the right panel under "Processed Image," showcasing green screen removal based on configured thresholds.

Tips for Varying Green Screens:

- - Solid Green Screens: Start with a conservative "Green Lower Bound" for uniformly colored green screens. Gradually increase until the desired removal is achieved.
- - Complex Green Screens: For screens with variations or intricate details, adjust "Green Lower Bound" and "Grey Bound" sliders to fine-tune the removal process.
- - Feel free to experiment with sliders and fine-tune thresholds to suit the nuances of your specific green screen.


~ - Shiny
http://127.0.0.1:3340
Open in Browser
Publish
Chroma - Keying
Select an Image
Browse...
satyap.jpg
Upload complete
Green Lower Bound
0
1
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Green Upper Bound
0
1
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Grey Bound
0.055
1
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

OBSERVATIONS

- Pixels that appears green visually have green component as maximum.
- When RGB values are close to each other they make a different variation of grey colour ranging from complete black to complete white.
- A large variation in green colour of the green screen background give poor results.
- Edges of the result images are not smooth, due to the colour spill present at the corners of the foreground object.
- More the contrast present in the foreground and background, the easier it gets to remove the green background. While similarities between them causes to remove all those pixels that appears even a little greenish.

CONCLUSION

- After exploring three approaches, the third approach emerged as the final solution for chroma-keying.
- The journey involved learning from various methodologies, encountering challenges, and gaining valuable lessons.
- Chroma-keying efficacy is closely tied to factors like lighting, color consistency, and equipment quality.
- Green screens are popular due to minimal color interference with skin tones, clothing, and common scene elements.
- Approach 1 focused on delineating RGB values to eliminate unwanted green background but faced precision issues.
- Approach 2 compared green components with red and blue, faced challenges with variations in lighting and foreground color.
- Approach 3, Dynamic Thresholding, removed pixels where the green component exceeded others, with user-adjustable thresholds.
- The dynamic thresholding system ensures adaptability and effectiveness, including a new threshold for predominantly green pixels that visually appear greyish.