**Approach**

1. Import all necessary libraries
2. Load the images or videos
3. Resize the images and the videos to the same size
4. Load the upper and lower BGR values of the green color
5. Apply the mask and then use bitwise\_and Subtract bitwise\_and from the original green screen image
6. Check for matrix value 0 after subtraction and replace it by the second image
7. You get the desired results.

**pseudo code for the algorithm**

chromakey algorithm  
input:  
fg(i,j) = i,j element of forground  
bg(i,j) = i,j element of background  
tola, tolb  
returns:  
out(i,j)  
  
get Cb and Cr of key color; call them Cb\_key and Cr\_key  
for each i,j:  
   get Cb and Cr for pixel value; call them Cb\_p and Cr\_p  
   let mask = colorclose(Cb\_p, Cr\_p, Cb\_key, Cr\_key, tola, tolb)  
   let mask = 1.0 - mask  
   out(i,j) = fg(i,j) - mask\*key\_color + bg(i,j)\*mask  
  
def colorclose(Cb\_p, Cr\_p, Cb\_key, Cr\_key, tola, tolb)  
   temp = math.sqrt((Cb\_key-Cb\_p)\*\*2+(Cr\_key-Cr\_p)\*\*2)  
   if temp < tola:  
      return 0.0  
   else if temp < tolb:  
      return (temp-tola)/(tolb-tola)  
   else:  
      return 1.0

**Description of the algorithm**

The basic idea is that we convert our images into YCbCr and look at the CbCr plane (we ignore luminesence). The key color defines a point on the plane. We divide the plane into three regions a region close to the key color, a region far from the key color, and a middle region. The close region is all colors that are less than tola from the key color. the far region is all colors farther than tolb from the key color. The middle region is the colors inbetween tola and tolb. We generate a mask as follow, the near region is all background (0), the far region is all foreground (1). The middle region is assigned a value based on its distance from the key color. we use the function (temp-tola)/(tolb-tola) where temp is the distance to the key color. (this function is 1 at distance = tola and 0 at distance = tolb.)

The neastest part of this algorthim, and one I borrowed from from Ashihkmin 1997 (available at http://bmrc.berkeley.edu/courseware/cs294-3/fall97/projects/ashikhmin/) is to remove the key color from the foreground so that you get less green halo around the forground objects. I have had remarkable results on really poor imput data. I got usuable results by standing in front of my pale green wall that was lit from the bottom. I have had really good results using a special very green cloth in the evening (even light on the screen).

**Sources**

<http://gc-films.com/chromakey.html>

<https://www.geeksforgeeks.org/replace-green-screen-using-opencv-python/>