Results so far

* Start by generating image and its shifted component with weightings summing to one.
* Begin by computing the shift
  + The shift is computed by performing an autocorrelation on the shifted image itself. This correlation is smoothed and the smoothed values are subtracted from the original to find the best peaks. These peaks are representative of when the autocorrelation finds a matching part of the image and its shift value
  + A green line with a white background

    Description automatically generated
  + Each channel is representative of a red green or blue channel
  + Some have better than others but normally all show a peak unless one red green or blue channel is suignificantly suppressed due to lack of that colour.
  + The positive peak in the center is representative of when the image is shifted over itself which is at the 0 peak it should match the best
* So now that the shift value has been obtained the w1 and w2 values need to be found that also make up the image.
* Knowing the shift values, a point spread function can be made with the weights at each end which can be used in a weiner deconvolution or Richardson (Richardson did not work well with the developed loss function.)
* Estimation
  + Differential evolution optimizer is used. This is done with scipy’s optiomsier. Uses all cpu cores for faster evalution
  + 3 sets of losses are added together for overall loss func using the autocorrelation of the deconvolved image with the estimated psf
    - The first one is ensuring there are no peaks in the autocorrelation valid output. That is the maximum values of the autocorrelation response when the image matrices match
    - Next is minimizing the peaks in the filtered autocorrelation response at the shift values. There always are peaks at the value where the image is shifted by. This makes sense and is the same logic as from identifying the initial shift values. The central peaks are ignored
    - Finally the flatness is also added to the loss. The flatness of the filtered autocorrelation response is simply determined by the standard deviation of all the values. Whatever that may be is added to the loss function.
  + There were a lot of issues arising with w1 and w2 being mixed up for some reason. Likely because the autocorrelation doesn’t take into account pixel intensities and appears to be symmetrical for weightings close to each other. This could change if the weightings were spaced further apart to make image shifts more distinct on one side rather than the other. This has been fixed by taking mean sobel value of the image for estimated w1 and image for w1 as w2. The higher the image the less “clear” it is. So these sobel averages are compared against one another and if one is less then the other that weighting is used. This is a temporary fix until I can look into measuring intensity in the correlation.

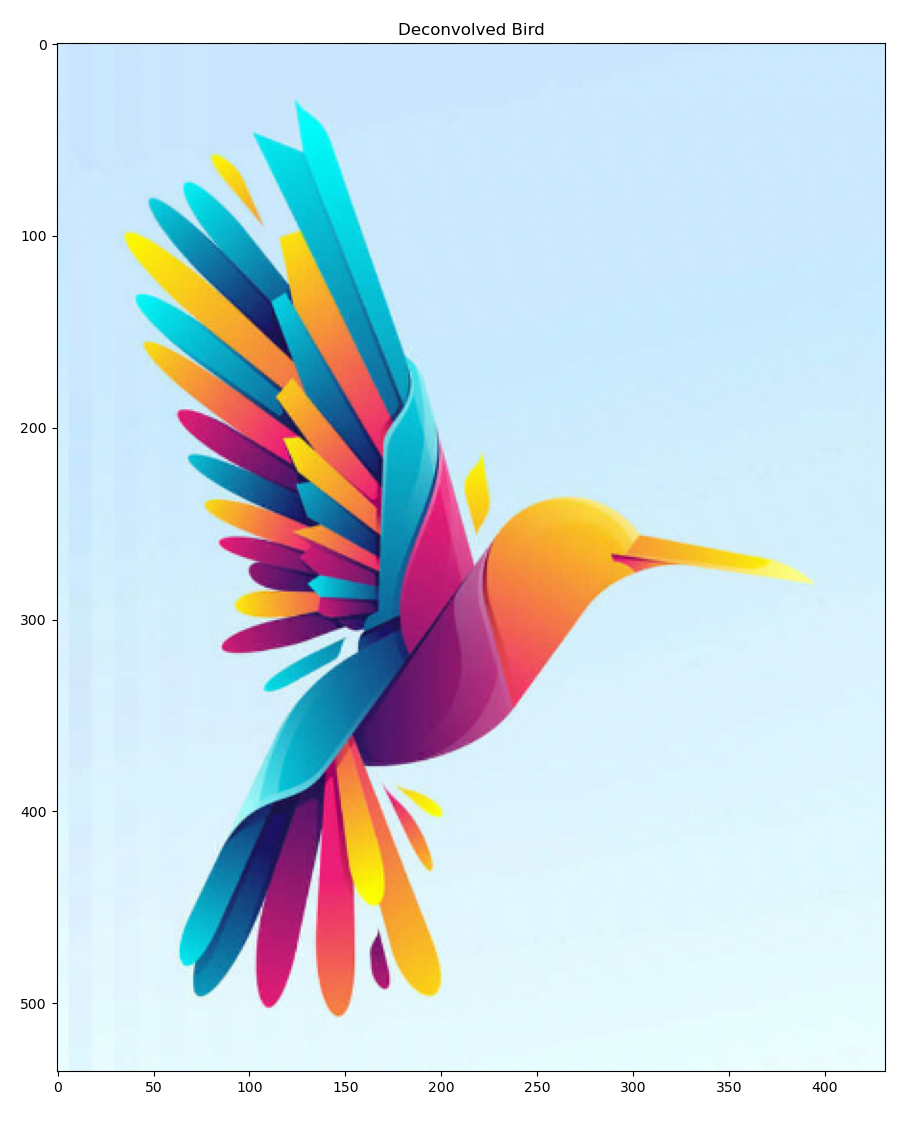


Figure weiner

A diagram of a bird

Description automatically generated

Figure weiner

A drawing of a hummingbird

Description automatically generated

Figure correct sobel

A bird with wings spread out

Description automatically generated

Figure incorrect sobel