George Bohorquez

Report

The Bayer Filter program takes a captured image and deconstructs it into a grid like structure using RGB color values. This structure is appropriately named the Bayer Filter. The structure of the Bayer Filter can be seen in the example below:

G	R	G	R	G
B	G	B	G	B
G	R	G	R	G
B	G	B	G	B
G	R	G	R	G

Throughout the pattern, the most prevalent color is green due to the fact that the human eye can detect more shades of that color than any other. From the Bayer Filter, a reconstruction of the original image can be reproduced. Let's use the table below as an example:

R1	G1	R2
G4	B = CP	G2
R3	G3	R4

The blue pixel in the center will accept the average color values that surround it. That is to say the following:

$$CPg = (G1 + G2 + G3 + G4)/4$$
$$CPr = (R1 + R2 + R2 + R4)/4$$
$$CPb = CPb$$

CP = Center Pixel
g = Green Color Value
r = Red Color Value
b = Blue Color Value

This will continue to be done for every blue pixel throughout the whole of the image. Similarly every red pixel will undergo a similar process. The green pixels however work a little differently:

| G1	R1	G2
B1	G = CP	B2
G3	R2	G4

$$CPg = (G1 + G2 + G3 + G4 + Gcp)/5$$
$$CPr = (R1 + R2)/2$$
$$CPb = (B1 + B2)/2$$

When putting the concept into code, the biggest problem came with interpreting the mathematics behind the for-loops. The green pixels were problematic in that for some there would be blue pixels directly above and below the green while others had red pixels directly above and below the green. This relegated the horizontal positions to have red or blue pixels respectively. Using (i%2), it was possible to detect when it would be necessary to capture the red or blue horizontal or vertical pixels. This was done using two if-statements that would act accordingly if the check for (i%2) = 1 came true or (i%2) = 0 came true.
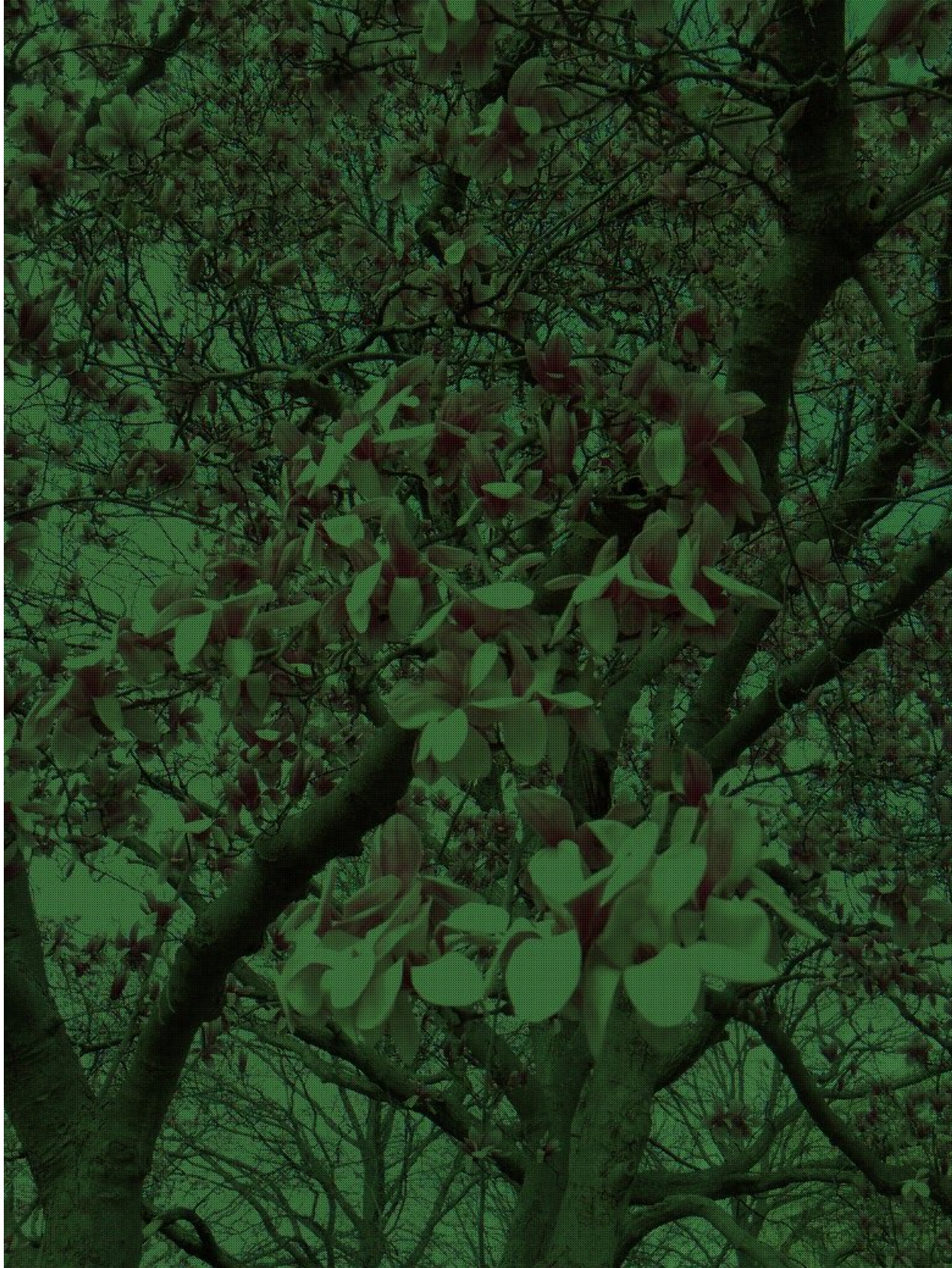
Original Image, RGB Filters, Bayer Filter
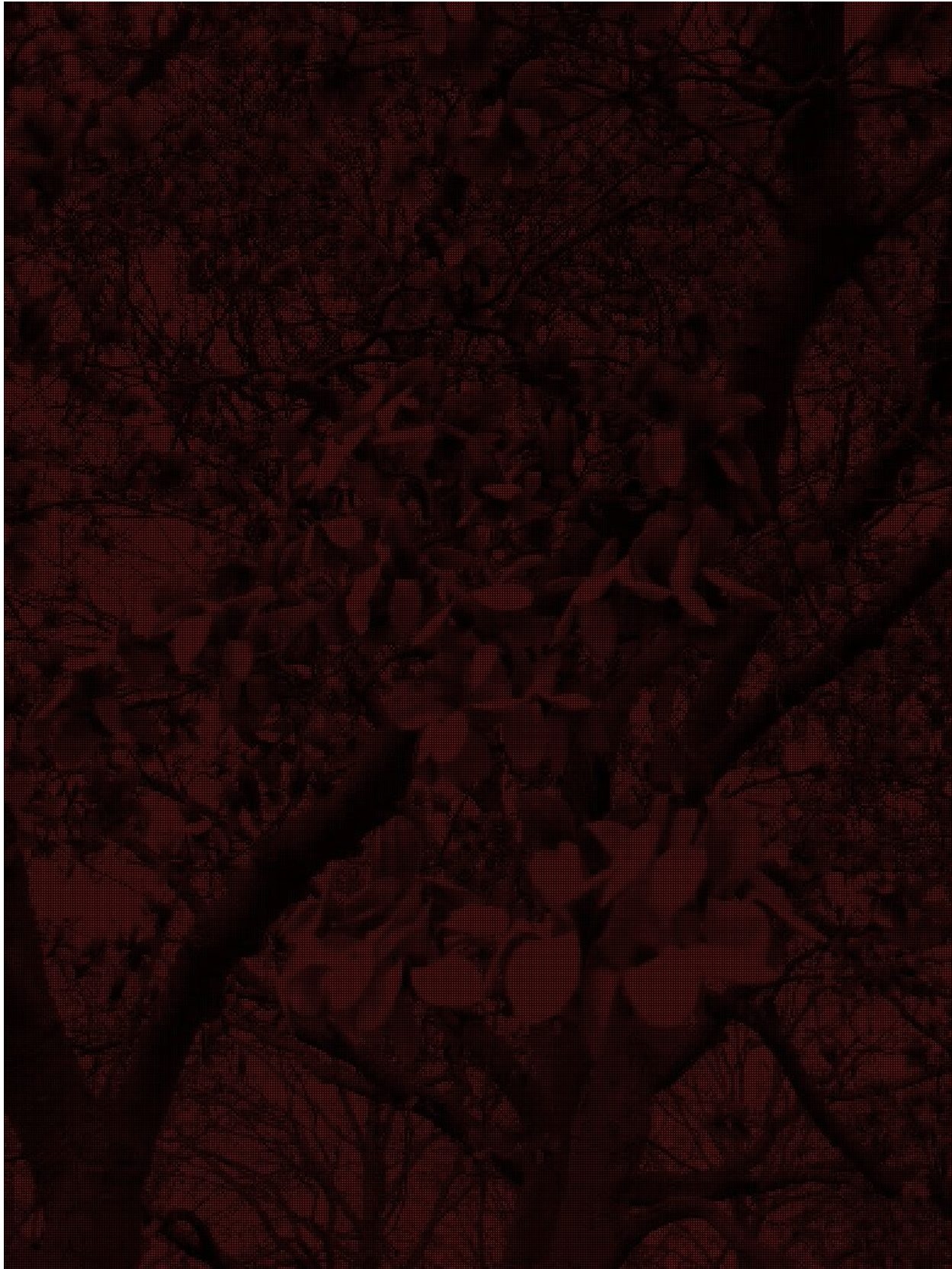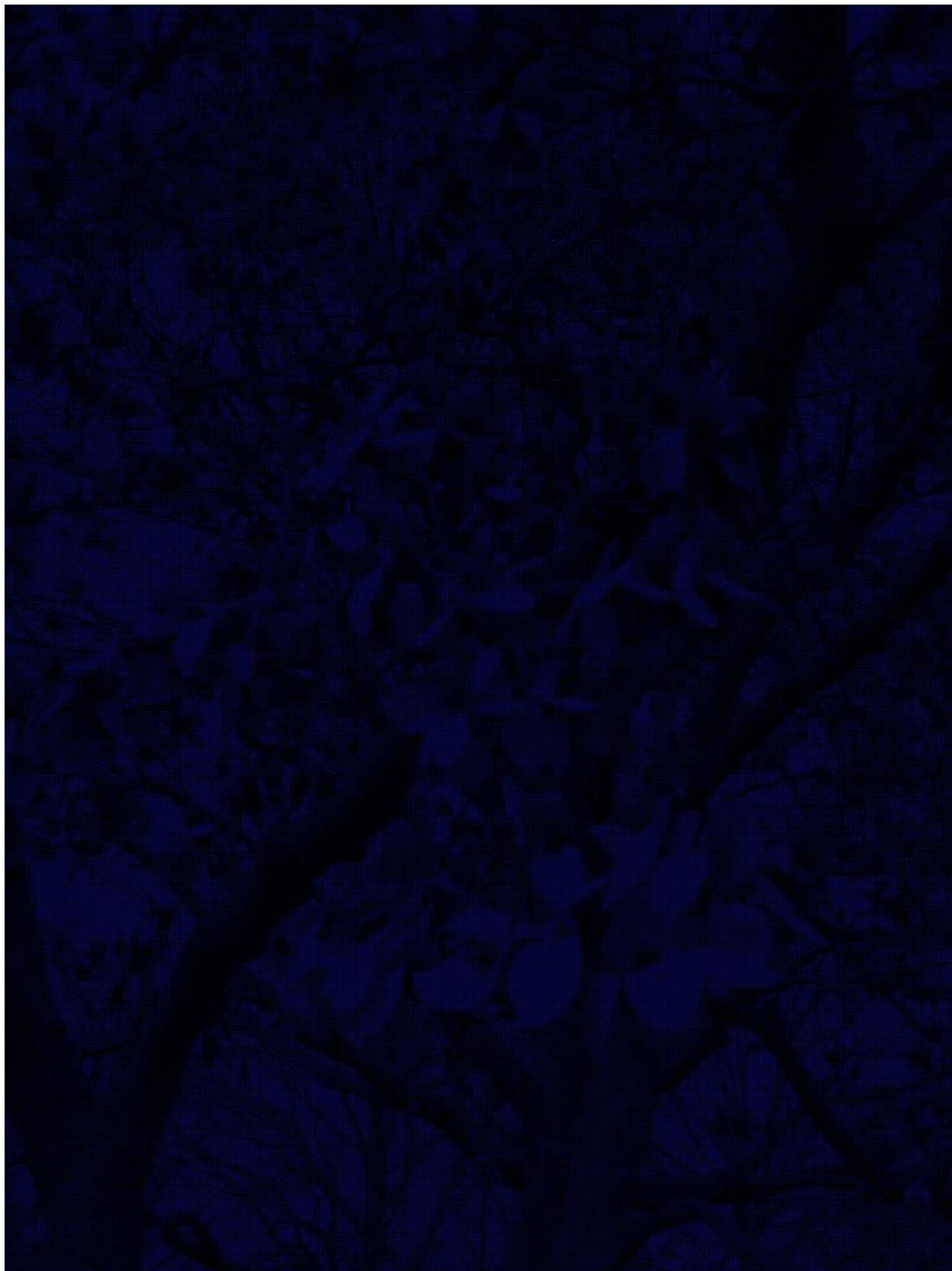
Other Images Tested - Original, Bayer Filter, Reconstructed

© Neil deGrasse Tyson 2001