

A testscene with the color keyer is attached. You might need to change some paths in the "movie" section of the scene to get it running. Test-Material is at \\192.168.3.35\public\Downloads\Media\Video\HollywoodCameraWork

## The Keyer

This node can be used to cut out the foreground from bluescreen or greenscreen footage.

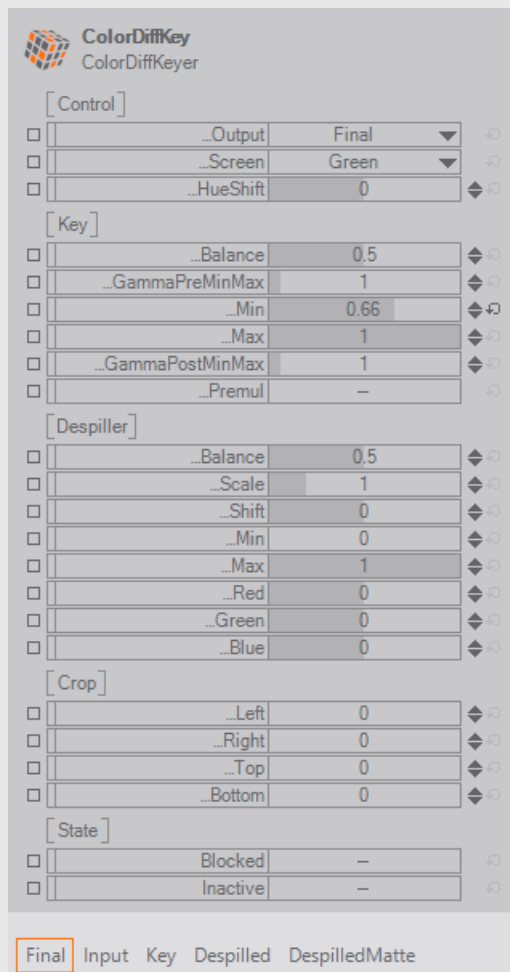
This involves four steps:

- generating the **Key**, which is black for the screen and white for the foreground
- removing color spill from the foreground (**Despiller**)
- **Cropping** areas from the outside. Often the greenscreen / bluescreen is too small to cover the whole camera.
- blend the footage-foreground on some other background.

This text assumes a greenscreen. When using a bluescreen, the colors mentioned will be different.

### Using the Keyer

There are many parameters to this node. This chapter gives an overview on how to tweak the parameters, and in which order to work. The next chapter will explain all parameters in detail.



The output property allows to control what you see: Either the final output, the unmodified input, or the result of intermediate processing steps. It is often easier to look at the Key matte than the final image when trying to get a good key. At the bottom of the property window there are buttons that let you quickly change between the various output options.

Before explaining the properties in detail, let's explain the recommended procedure to tweak the keyer:

### How to Make a good Matte

- Make sure all parameters are set to default values. The counter-clockwise arrows right of the properties do this.
- With the **Screen** property, select if you are using blue or green screen. There is also a setting for a red screen should you need that.
- Set the **Output** property to **Key**. Note that you can use the buttons at the bottom of the property window to change the output property.
- Sometimes the green or blue screen does not have the "right" color. Use the **HueShift** to compensate for that. Change the value until the background is as dark as possible. Do not use the HueShift to shift from green to blue, use the **Screen** property for that.
- Next is **Min** and **Max** in the **Key** group. This is the most important step in the whole process. These values limit the range of the key, cutting pixel darker than Min and brighter than Max. The key will then be rescaled and biased to full range. First increase Min until the background is completely dark. Next decrease Max until opaque areas in the foreground are completely white. The contour of the key will have more contrast if the min and max values are far apart.

- When not satisfied with the result, move **Min** and **Max** a bit apart from each other, so that keying-errors are more visible. Then try tweaking **Balance** and **HueShift** and see if that improves the situation. Having set **Min** and **Max** close to the correct values makes the effect of **Balance** and **HueShift** more visible. Then go back to the step above.

- Set the **Output** to **Final** and set **PreMul** to true.

This should produce a key in the alpha channel that is fit for premultiplied alpha blending (the default mode for Ventuz).

## Getting rid of color spill

Often the foreground has color spill. That means that color from the background (green or blue) is reflected by the foreground into the camera. The **Despiller** is very effective to remove this using the default settings. This can be seen by comparing the **Final** output with the **Input** (use the **Output** property to compare).

With the **Output** property, you can see the **Despilled Matte**. Dark areas are not effected by despilling, white areas will be fully despilled. Set it to **Despilled** shows the despilled image. When using greenscreen, the despilled image subtracts the green out of the screen making it look brownish.

It can happen that the default setting make the contours and hair in the foreground look colorful. It is not the green color you will see, but the despilled brownish color. The **Green**, **Blue** and **Red** properties can be used to change the color to something grayish, which would be less noticable. The default values are 0.

If areas are effected by despilling that should not, or vice versa, the despill mask needs tweaking. The **Min**, **Max** and **Balance** in the **Despill** group work the same as the key-properties.

## The Parameters of the Keyer

The explanations assume a greenscreen. When using bluescreen the colors are switched.

### Control

**Output:** Select what to see:

- (0) **Final:** The final image with key and despill applied
- (1) **Input:** The unmodified input texture
- (2) **Despilled:** Despill applied, but no key.
- (3) **Key:** The key as grayscale image
- (4) **Despill Matte:** a grayscale image showing where and how strongly to apply despilling.

**Screen:** Select green, blue or red-screen.

**HueShift:** Compensate for screen that do not perfectly match the green, blue or red of the camera. The HueShift is not applied to the foreground, just used for key generation.

### Key Generation

This node uses the **color difference** method to generate a key. Unlike a chroma keyer that converts the signal into HSV color space and then picks a range, a color difference keyer looks at the difference between the green color channel and the other color channels. This quite simple technique produces astonishingly good results with little tweaking and is the basis of many of the most famous keyers.

**Balance:** Balance between the non-key components of the input. Assuming green-screen, the red and blue channels are compared to the green channel with equal effect when Balance is set to 0.5. Changing Balance toward 0.0 will reduce the influence of the blue channel, changing it towards 1.0 will reduce the influence of the red channel.

**GammaPreMinMax:** Use a gamma function on the key before applying the min/max limiting. The function is `pow(input,gamma);` . This is good to make dark (background) even darker, but should be used sparingly, otherwise the contrast in the key is lost.

**Min** and **Max:** Limit the range of the key to values between min and max, and then scale and shift the key to fill the whole range (from black to white or 0 to 1).

**GammaPostMinMax:** Another gamma function, this time after the min/max limiting. This is good to control how soft the contours / hair are keyed.

**!Premul:** The key is used together with alpha blending. By default, Ventuz uses premultiplied alpha blending and this property should be set to true (checked). If a custom blending is used, it might be required to uncheck this.

### Despiller

The despiller works also on looking at the difference between key-color (green) and the other channels (red & blue), but in a different way. this produces a despill matte that marks the areas that need despill.

A simple despiller would just use this matte and subtract it from the green channel. This version allows to control the amount and additionally change the other channels too.

**Balance:** Balance between the non-key components of the input. For a green-screen, balance how much to look at the red and blue.

**Scale:** Multiplies (changes brightness) the despill matte.

**Shift:** Adds (contrasts) to the despill matte.

**Min** and **Max:** Limit the range of the despill matte to values between min and max, and then scale and shift the key to fill the whole range (from black to white or 0 to 1).

**Red**, **Green** and **Blue:** Control the color change of the despill process. First the despill matte is subtracted from the green channel of the result image,

dragging the green out of the picture. Then the despill matte is multiplied by the color supplied and added to the result image. Usually this will result in a brownish background (using greenscreen). By changing these values the background can be shifted to any other color, preferably gray. By the way this process works, controlling the colors can be unintuitive.

## Crop

Cropping allows to limit the effective area of the keyer to cut out parts of the screen where the greenscreen is too small. The outside of the cropped area will be transparent black.

Cropping is only effective when **Output** is set to **Final**.

**CropLeft, CropTop, CropRight, CropBottom**: A value of 0% means no cropping. A value of 100% means the crop-line moves to the opposite border, total cropping.