COMPOSER

Creating a multi-camera setup using Chroma Key, multiple brandings, and RTMP targets.

Version 1.1, 2023-02-10

1. Introduction

One of the most powerful features in Composer is the integrated Chroma Keyer and multi-camera support, combined with triggers and support for multiple targets (outputs). This guide is a complement to the demo project **ComposerMultibrandingDemo.zip** which is available for free download at our support area.

The concept of using **multiple cameras** and **multiple outputs** (for different end-customers, or "brands"), can be scaled to more than two cameras and two brands. The only limitation is the available CPU and GPU resources.

While traditional setups include individual video switchers, keyers, media storages, and encoders, Composer combines all these features into one unit. As a result, Composer does not require the same maintenance as traditional setups and is easily scaled to a multi-table environment, yielding a higher ROI.

In this demo, we are creating a setup consisting of the following:

- **Two camera** inputs. For the demo, we use video clips as camera sources, but live sources such as SDI/HDMI and NDI inputs would work in a similar way.
- Dual **Chroma Keyers** (one per camera) with Garbage matte.
- **Images and videos** for backgrounds and overlays.
- Two output signals (RTMP)
    - One for **Customer A** (dark background)
    - One for **Customer B** (bright background)
- Two separate **color curve filters** to improve the white balance between the foreground and the background.
- **Sharpening** filters
- A vignette image for darkening the edges of the image.
- An API for **switching between the two cameras.**

In the setup, the **RGB Straight blend mode** is used to improve the edges of the Chroma Keyers.

Components that will be used are movie and image sources, HSV Chroma Keyer, Color Curves Filter, Sharpening Filter, Alpha Channel Coring Filter, Connectors, and RTMP targets.

2. Inputs

In this demo, we are using two cameras as inputs. However, these are not captured via SDI/HDMI/SDI. Instead, we are using two **video clips** found in the Media folder of the zip file. The two clips (**Front Camera, Top Camera**) were recorded in a studio using a greenscreen background.

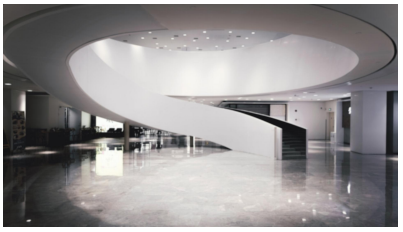
Notice the cabling and scaffolding in the upper left corner of the second clip (top camera). We will use a **garbage matte** to remove those objects and to avoid spill suppression on Number 0 (green).

Other inputs include the **background image** used together with the front camera for Customer A. The image was created using a third-party 3D rendering suite (Maya).

On top of the background, there is a layer of **animated curtains** (video clip). By separating the two layers (background image and curtains), you can create a parallax effect simulating a zoom-in, a camera pan, or a camera dolly. However, such a feature is not part of this demo.

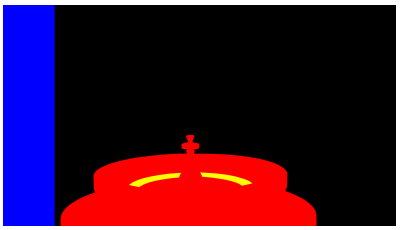
The animated curtains video clip contains an alpha channel and uses the SPEEDHQ video codec.

For the front camera of customer B, an image is used as the background. This image is slightly **blurred** in order to reinforce the feeling of depth of field. The image is downloaded from Unsplash (Steven Wei, Free to use under the Unsplash License) and was pre-processed using Adobe Photoshop.



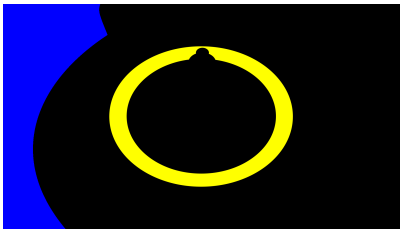
To remove unwanted visual objects on the front camera and to remove spill suppression on the Number 0 pocket of the top camera, we use two **garbage matte** images. The garbage matte images were created in Photoshop.

The garbage matte image is used for defining areas where you want to add or remove space for chroma keying or spill suppression.



The garbage matte image is added to the HSV keyer Operator.

A garbage matte is an RGB image containing three types of complementary information: Additive alpha, subtractive alpha, and spill suppression reduction.



Red channel

Areas with added alpha. In these areas, the Alpha channel values are determined by the values in the garbage matte image.

Green channel

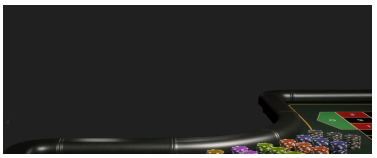
Areas with no color spill suppression. In these areas, the amount of final spill suppression is multiplied by the values in the garbage matte image and the amount of spill coming from the keyer.

Blue channel

Areas with subtractive alpha. In these areas, the Alpha channel values are subtracted from the values in the garbage matte image.

Yellow pixels are a combination of the red and the green channels, and for those pixels, the alpha channel is set to solid, and no spill suppression will be applied.

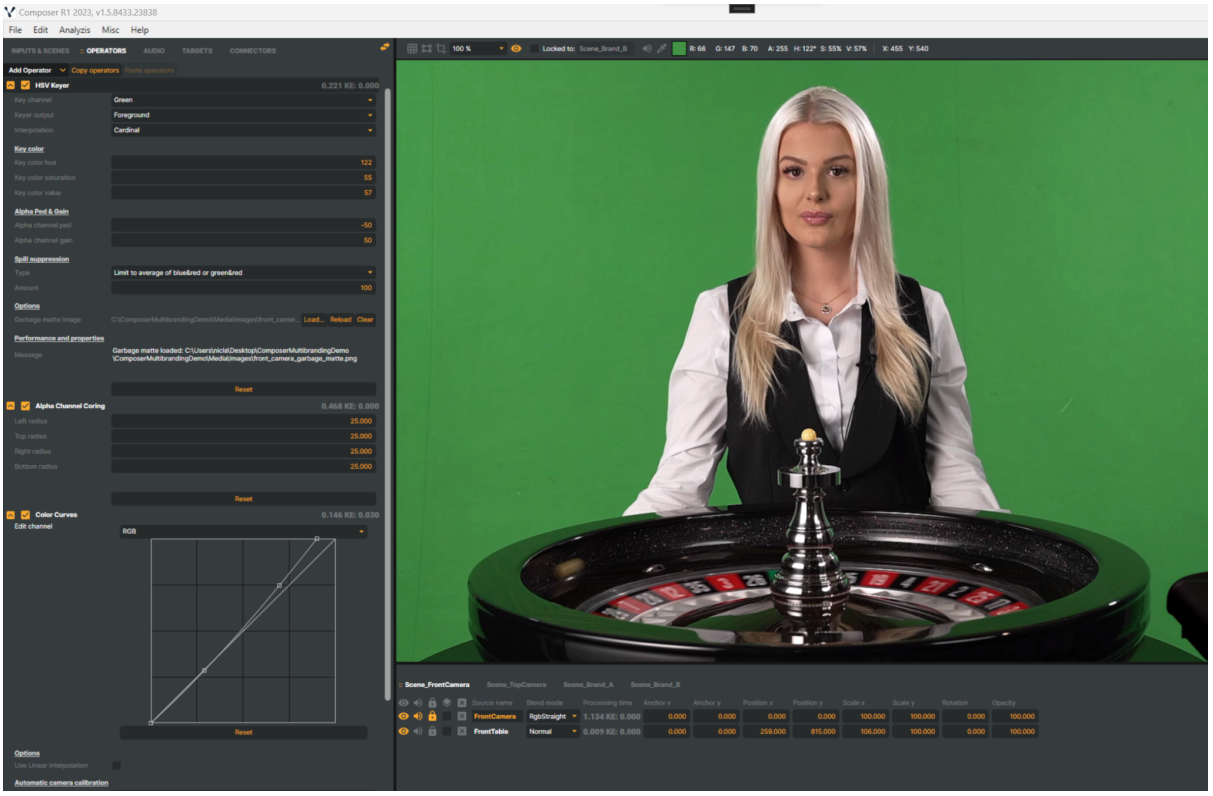
A **virtual table** section (in front of the roulette wheel) - is also used together with the front camera. This image helps give the impression of a genuine roulette table.



Finally, two images are used as the **floor** in the top camera.

3. Chroma Keyer

The **Color Picker tool** can be used to ensure the correct hue, saturation, and value of the background. **Select** the Chroma Keyer (title) and use the Color Picker tool to automatically pick the HSV values and transfer them into the settings of the Chroma Keyer.

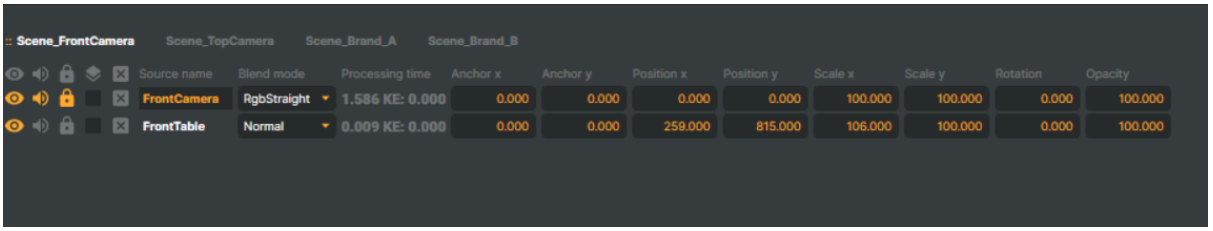


We set the **Interpolation** mode to Cardinal (instead of Linear), and adjust the **Ped** and **Gain** to improve the alpha channel. We also add the **Garbage Matte** image (created in Adobe Photoshop)

In this setup, we only use one Chroma Keyer per camera but two different backgrounds. Limiting the number of Chroma Keyer instances will reduce the overall load of the server. In order to reuse the keyer outputs, put the keyers into separate Scenes (Scene_FrontCamera and Scene_TopCamera), and create two other scenes for the compositing (Scene_Brand_A and Scene_Brand_B).

More information on the HSV Keyer operator is found here:
https://composer.docs.vindral.com/docs/hsv-keyer?highlight=hsv%20keyer

4. Blend mode RgbStraight



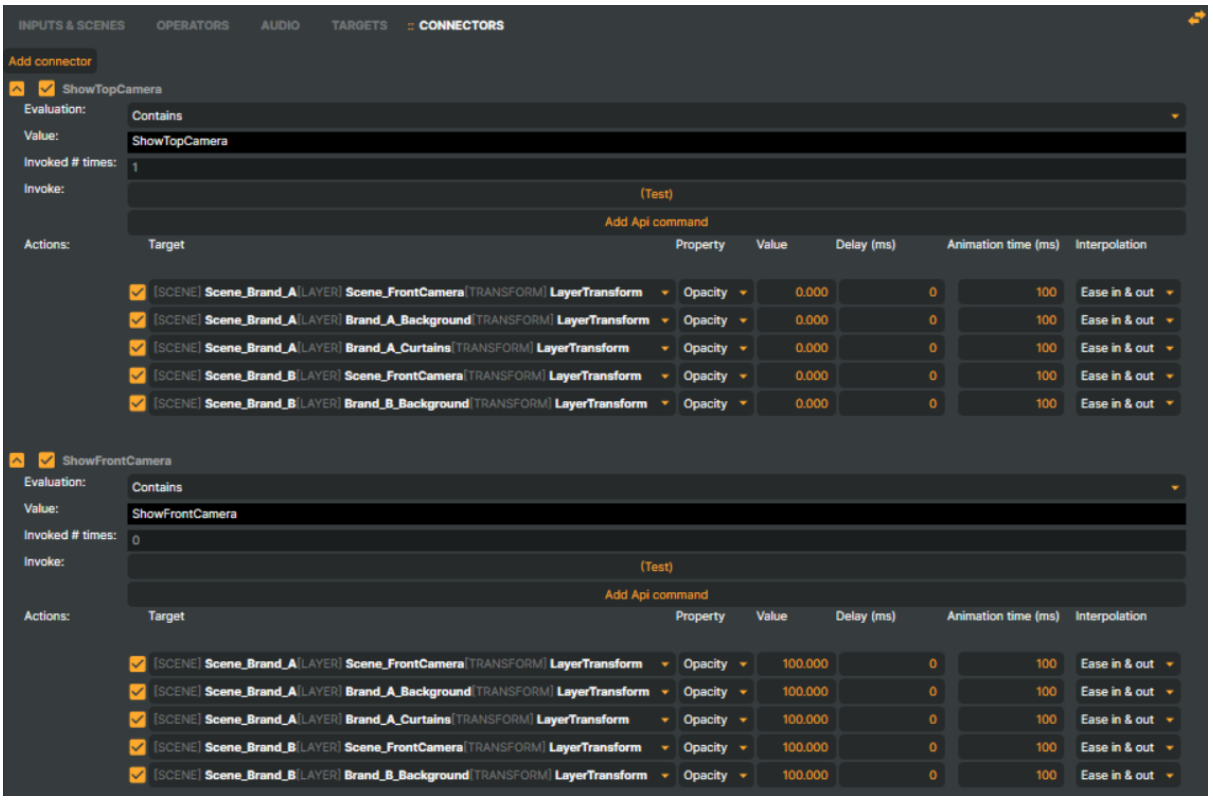
To keep as much information (RGBA) as possible from the keyer, we use the **Blend Mode RgbStraight** (on the same scene layer we added our HSV keyer). The RgbStraight Blend Mode will improve the edges of the final images created in Scene_Brand_A and Scene_Brand_B.

5. Switching cameras using Connectors

In order to externally (via API:s) control which camera to use, we create two **Connectors** – **ShowTopCamera** and **ShowFrontCamera**. Both of these endpoints can be reached using the built-in HTTP server, allowing camera switching to be controlled via an external switcher, timer, or trigger.

The two connectors use several API Commands where layer opacities are changed to show/hide different layers. This feature has the same effect as changing the camera. We also added a bit of **animation time** (100 ms) to create a **mix effect** instead of a cut effect. We set the interpolation mode to Ease in & out, rendering a smooth effect.

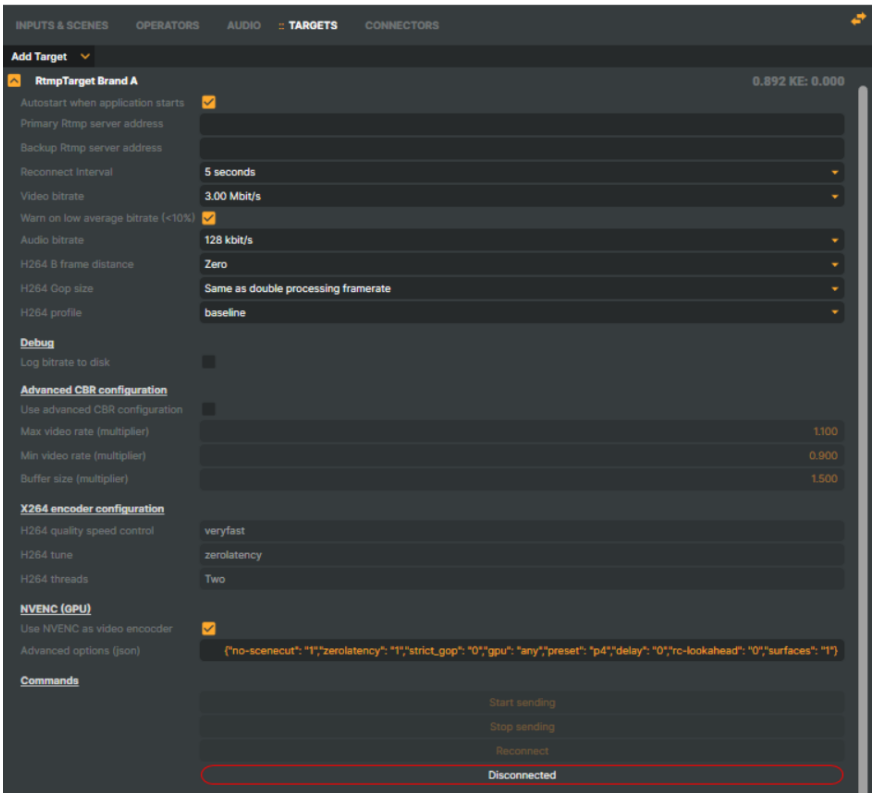
You can test the connectors manually by pressing the **(Test) button**.



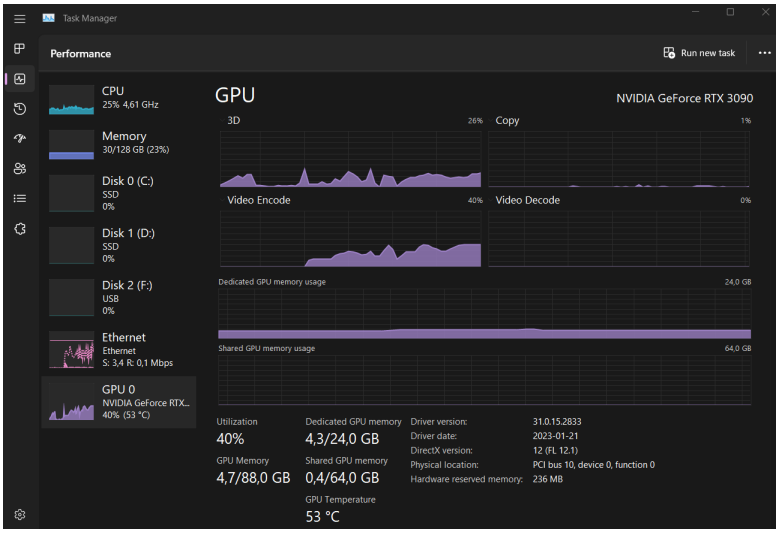
More information on connectors is found here:
https://composer.docs.vindral.com/docs/en/connectors?highlight=connector

6. Targets

As we create two scenes (one per brand), we can have **separate output configurations** per brand. In this example, we use two separate RTMP targets – one per brand. Note that this is an illustrative example, in which the targets have not been fully configured yet.



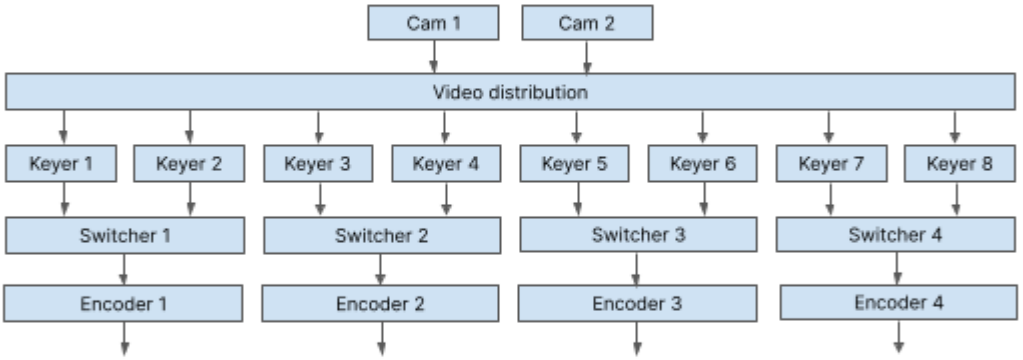
We activate the **NVENC** (GPU) option, which will use the built-in encoder of the NVidia GPU. This option will reduce the load on the CPU. However, please note that the encoder capacity of the GPU is limited. Use the **Performance Monitor** to check the **Video Encode load** on the GPU:



7. Adding more outputs (brands)

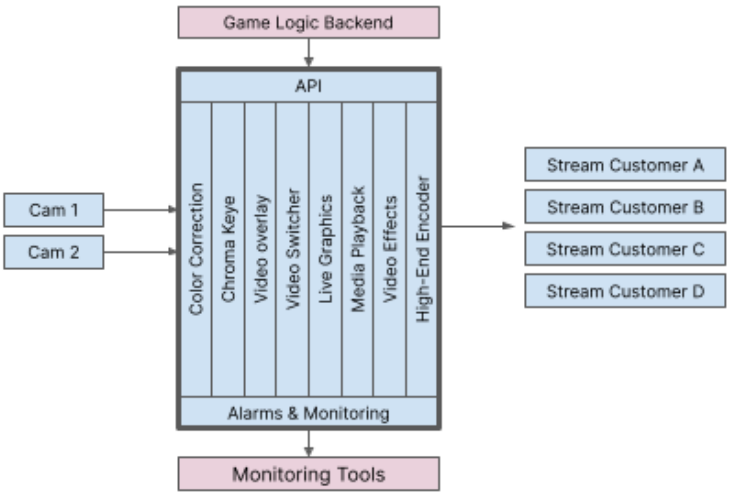
The demo is easily modified to support more than two rtmp outputs. By adding more scenes, like Scene_Brand_A, and adding graphics for the two camera views, the setup can support up to 8 brands (depending on the CPU and GPU performance, additional brands may be supported).

A traditional multi-camera setup using chroma keyers with multiple outputs (different brands), quickly becomes complicated. The below example illustrates a typical setup using two cameras and four brands.



This setup normally requires 17 devices and 26 cables. In reality, some setups are even more complex.

A similar setup based on Composer only requires one server (one RU) and less cabling. It is also easier to control via the built-in API:s (connectors).



Note: If you are using the LE version of Composer, you are limited to a maximum of 2 RTMP targets.

More Information

For more information, visit

https://www.realsprint.com

https://www.vindral.com

https://docs.vindral.com