STAG: Smart Tools & Apps for Graphics (2014)
Andrea Giachetti (Editor)

CHROMAGRAM: A real-time Chroma Key application for mobile devices

Fabrizio Corda, Fabio Sorrentino, Riccardo Scateni

Department of Mathematics and Computer Science, University of Cagliari - Italy

Abstract

Chroma Key is a special-effects technique widely used by television and motion picture industries for image composition. This technique allows users to replace sections identified by a chosen colour in a multimedia stream (like a video or a photo) with another image or video stream.
In this paper, we describe an easy-to-implement technique for the creation of an Android based application for mobile devices (like smartphones and tablets) that applies Chroma Key-based effects to the video stream coming from the device camera. We discuss the algorithm used to achieve the Chroma Key effect focusing on the computational performance and on the quality of its final result. Using a picture selected from the device gallery, this application makes possible the replacement of video stream background areas characterized by a chroma value with the chosen picture.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Object recognition

1. Introduction

Nowadays the Keying effect plays a very important role in the film industry field, and it is also widely used by amateur videomakers and photographers during the post-processing image composition to generate visual effects (see Fig. 1 for a simple example). This technique allows to replace identified sections in a multimedia stream with another media. It is possible thus to replace a video with an image as well as an image with a video depending on our needs.

The main drawback of this technique is that it has a high computational cost, however this negative aspect is not really significant thanks to the increasing of the computer, as well as mobile devices, processing capabilities during the last years.

In this work we focused on mobile devices capabilities due to the progressive diffusion of smartphones and tablets and their usage as photo and video recorders. This paper describes an efficient, fast and cheap (avoiding the need of buying high cost devices) easy-to-implement method for the real time chroma key matting effects computation, using mobile devices like smartphones and tablets, minimizing as much as possible the computational cost to achieve good visual results.

Figure 1: *Use of the blue of the sky as key colour.*

The main idea is to study how much the Android platform is suitable for the real-time image processing, giving to users a powerful tool able to stimulate their imagination allowing, for example, the generation of fast low-resolution preview of complex chroma keying effects, or simply generating cool and fun images and videos.

DOI: 10.2312/stag.20141240

2. Related work

During the application development, we studied several keying (and matting) techniques and approaches to understand which of these could be the best to use in a mobile setting.

Two of the most widely used techniques are the **Bayesian** [CCSS01] and the **Poisson Matting** [SJTS04] that provide great results with random backgrounds and intricate boundaries.

The **Luma Keying** technique [Bri08] allows to create the matte using brightness information (luminance) of the HLS representation of images. By setting a threshold value for the luminance, that is independent from the color information of the image, we can create the binary matte for the image. This approach is useful if applied to bright foreground objects on dark background.

The **Difference Keying** approach [Wri13] provides the matte creating it from the subtraction of two photos, the first one is made up using the background and foreground objects, while the second one contains only the background. This technique does not require a dedicated photo set, but needs only two photos (or videos) with the same background.

Depth Keying-based [KOY*95, CZGL09, KIA*04, GKOY03] techniques provides the matte through special depth camera equipment that defines the depth of the scene objects, making transparent everything that it is located beyond a chosen depth threshold.

The **3D Keying** [Sch06, Pho] represents the image in a three dimensional space where every axis represents one of the primary colors: red, green, and blue. Then we are able to separate the background from the foreground using two shapes.

In this work, we chose the 3D Keying technique thanks to its great quality of the results, the good computational efficiency but mostly thanks to the easy implementation that allows us to test the performance of the Android platform with real-time image processing.

3. Application design

The proposed application has been developed for the Android mobile platform. We chose this platform for several reasons like the high distribution of mobile devices that make use of it, its simplicity and the low cost of development. Moreover, at the time we started to develop it, no similar applications were found in the Android app store.

3.1. User Interface

We designed a very simple interface, allowing the use of all implemented features in an easy way without additional texts or tips (see Fig. 2).

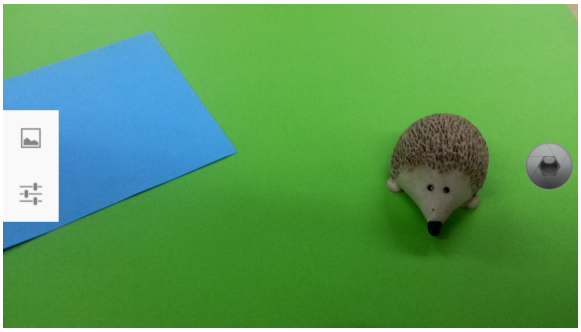


Figure 2: *UI screenshot.*

The Background Substitution effect can be done simply touching the camera preview, through this operation the entire image section characterized by the key colour picked is selected.

Then, using the side menu buttons, we can choose the substitutive background photo, edit the threshold and smoothness of the algorithm and shoot a photo of the resulting effect.

3.2. Implementation

The basic idea to obtain the background substitution effect is to find and replace all pixels in the video stream that are characterized by a colour within a range using a certain tolerance applied to a key colour chosen by the user (in Fig. 3 a single-colored piece of fabric is placed in the scene for these purposes).

Figure 3: *Using a green screen to achieve the final effect.*

To implement the processing system we used the OpenGL ES 2.0 graphic library. In detail, after the drawing of a rectangular plane that covers the entire screen surface, every frame obtained through the device camera will be computed and processed through the OpenGL fragment shaders obtaining the background substitution matting effect. Then will be possible to renderer this frame on the screen using it as a plane texture. Through the OpenGL fragment shaders, for

every image texel of the frame obtained through the camera, we can apply the matting effect as described in the next paragraph.

An initial development phase of this application involved the OpenCV graphic library usage instead of OpenGL. We abandoned this development due to the better performances obtained with OpenGL that makes a better and optimized use of the GPU.

3.3. Algorithm

The algorithm used in this application make use of the 3D keying technique as described in [Sch06] where they represent all the pixels in a R^3 Euclidean space where the three dimensions represent each of the primary colours: red, green and blue.

Given a point k representing the chosen key colour that should be replaced with an alternative background, using two spheres S_1 and S_2 centered in k (with different radius), it is possible to define the transparency of every pixel (or texel) of the image. Every pixel p placed inside the inner sphere S_1 will be fully transparent while the pixels placed between S_1 and the outer sphere S_2 will have a proportional transparency to the distance from S_2: closer to the inner sphere S_1, higher transparency value; closer to the outer sphere S_2, lower transparency value (se an example in Fig. 4).

In details, for every pixel, to calculate the distance of this pixel p from the key colour k, we use the Euclidean distance:

$$distance = \sqrt{(p_x - k_x)^2 + (p_y - k_y)^2 + (p_z - k_z)^2}$$

After this calculation, if *distance* is lower than the radius r_1 of the inner sphere S_1, then the α opacity value of the pixel will be equal to 0.0 (fully transparent), else if the *distance* is higher than r_1 but lower than the radius r_2 of the outer sphere S_2, the α value will be calculated with the following formula:

$$\alpha = (distance - r_1)/(r_2 - r_1)$$

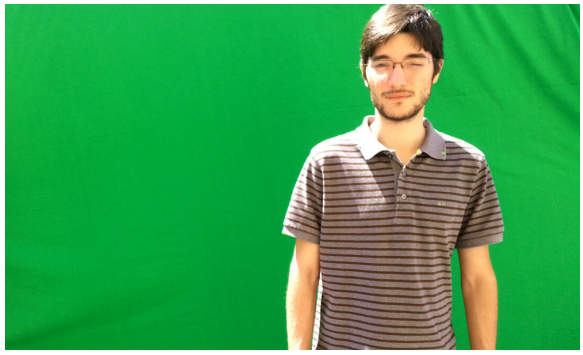
The final pixel p_f value will be calculated as linear combination of original pixel p_o and substitutive image pixel p_s with α as the scalar value, as represented in the following expression:

$$p_f = \alpha p_o + (1 - \alpha) p_s$$

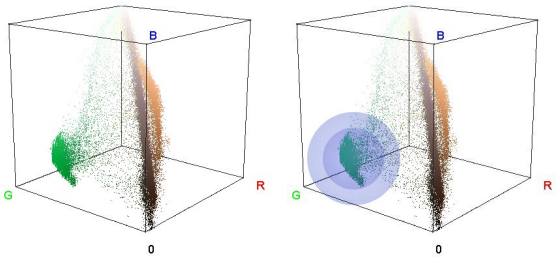
So, the r_1 radius represents the threshold of the algorithm and the r_2 radius show the smoothness.

3.4. Performance

We tested both the OpenCV and the OpenGL-based applications on several Android devices obtaining very different performance results. In Table 1 we reported the performances of both applications on devices with different resolution and processing power.



(a) sim



(b) histo1 (c) histo2

Figure 4: *An example of how the 3D Keying effect works: given an image (a), this technique represents all the pixels in the RGB space (b),separating the background from the foreground using two spheres (c).*

Device	Resolution	OpenCV	OpenGL
Sensation XE	800x480	15 fps	18 fps
Galaxy S3	1280x720	15 fps	30 fps
Galaxy S4	1920x1080	8 fps	30 fps

Table 1: *Performances measured on Samsung Galaxy S3, Samsung Galaxy S4 and HTC Sensation XE.*

3.5. Limitations

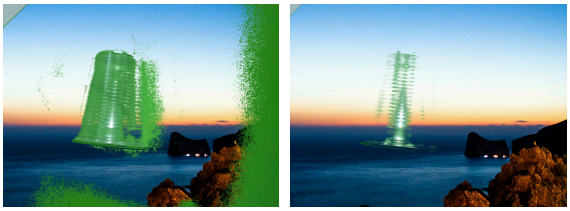
The Chroma Key matting effects have several issues related to the scene illumination, the video compression and obviously due to the camera quality. Other issues are related to the limitation of the algorithm used to obtain the matting effects.

In particular, we might see some reflections on foreground objects that have the same value of the key colour (this effect is called color spill) caused by one of these issues or their combination: low smoothness value of the algorithm, bad scene illumination, excessive color reflection of the Chroma

Figure 5: *Color spill around the girl's hair.*

Key screen, and lastly, the materials that characterize the foreground objects (see Fig. 5).

(a)



(b) (c)

Figure 6: *An example of semitransparent object (a). With a small threshold, we preserve the glass details (b) but we still see some of the back key colour. Increasing the threshold (c), we loose glass details.*

Implementing a spill removal algorithm is a good way to solve this problem, but this addition decreases the application performance and it requires a good setup of the studio scene. Another issue that we might encounter is the illumination difference between foreground objects and substitutive background, that produces unrealistic results of the final image composition. A last noticeable issue is that we can lose small details of foreground objects, like hair for framed persons, or semitransparent objects made with material like glass (see Fig. 6).

To solve most of this issues and obtain good quality results, at this moment we have to set correctly the shooting scene, using independent lights for foreground and background objects. Again, the background illumination of the shooting scene must be as uniform as possible and its color must have a good contrast with the foreground objects. The most widely used background color for this kind of effects is light green or blue.

4. Conclusion and future work

In this paper we described an Android application for computing Chroma Key-based effects with the video stream coming from the built in camera. We are satisfied by the obtained results since the used techniques revealed high efficiency when implemented on a mobile device.

In the next future we plan to add new features, allowing the recording of video and improving the Chroma Key algorithm using a Bayesian or a Poisson Matting filter that provide better visual results, solving some of the issues described previously. We are also working to decrease the complexity of the algorithm, preserving the computational performance results.

References

[Bri08] BRINKMANN R.: *The art and science of digital compositing: techniques for visual effects, animation and motion graphics*. Morgan Kaufmann, 2008. 2

[CCSS01] CHUANG Y.-Y., CURLESS B., SALESIN D. H., SZELISKI R.: A bayesian approach to digital matting. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 2, IEEE, pp. II–264. 2

[CZGL09] CHO J.-H., ZIEGLER R., GROSS M., LEE K. H.: Improving alpha matte with depth information. *IEICE Electronics Express 6*, 22 (2009), 1602–1607. 2

[GKOY03] GVILI R., KAPLAN A., OFEK E., YAHAV G.: Depth keying. In *Electronic Imaging 2003* (2003), International Society for Optics and Photonics, pp. 564–574. 2

[KIA*04] KAWAKITA M., IIZUKA K., AIDA T., KURITA T., KIKUCHI H.: Real-time three-dimensional video image composition by depth information. *IEICE Electronics Express 1*, 9 (2004), 237–242. 2

[KOY*95] KANADE T., ODA K., YOSHIDA A., TANAKA M., KANO H.: *Video-Rate Z Keying: A New Method for Merging Images.* Tech. rep., DTIC Document, 1995. 2

[Pho] PHOTRON: `www.primatte.com`. 2

[Sch06] SCHULTZ C.: Digital keying methods. *University of Bremen Center for Computing Technologies, Tzi 4* (2006). 2, 3

[SJTS04] SUN J., JIA J., TANG C.-K., SHUM H.-Y.: Poisson matting. In *ACM Transactions on Graphics (ToG)* (2004), vol. 23, ACM, pp. 315–321. 2

[Wri13] WRIGHT S.: *Digital compositing for film and video*. Taylor & Francis, 2013. 2