A Delaunay Keyer for Colorspace-Local Matte Extraction and Spill Suppression

Isaac Criddle[1] Seth Holladay [2] Parris Egbert [2]

[1]Stanford University, USA
[2]Brigham Young University, USA

Abstract

Keying is a fundamental, common, and expensive problem in visual effects, consisting of two tasks: matte extraction and screen spill removal. Today's keyers often fall short on common edge cases including unevenly lit screens, fine edge detail, and sky replacements. Keying thus continues to be a laborious, complex and expensive problem. Our keying algorithm interpolates over a Delaunay tetrahedralization of user-given sample colors to increase flexibility and expressive power. We compare our method to the visual effects industry's standard keyers, achieving comparable or superior results for matting and spill suppression.

Keywords: Chroma key, Green screen, Blue screen, Matting, Image processing, Color spill, Delaunay triangulation

CCS Concepts

• **Computing methodologies** → **Image processing; Image manipulation; Computer graphics;**

1. Introduction

In media production, compositing entails believably integrating foreground objects from one shot into the environment of another. Chroma keying, in which an actor or object is filmed in front of a highly saturated wall or curtain, is the most popular method to obtain foreground elements. Keying is most straightforward when the screen is evenly lit, foreground elements are free of bounce light, and there is no fine edge detail. The reality of visual effects production is not so simple. Keying continues to be expensive and labor-heavy, requiring significant manual tuning and intricate compositing of the outputs of multiple keying methods [OZ20]. Motion blur, defocus aberration and sensor noise further complicate the process.

To address the persistent challenges of user control, fine detail, and spill suppression, we introduce the **Delaunay Keyer**, present a new keying algorithm. Our approach leverages a Delaunay tetrahedralization of user-defined color samples to create a flexible, colorspace-local model. This unified framework provides simultaneous matte extraction and spill suppression, offering artists fine-grained control over complex elements where other methods falter.

2. Background

Smith and Blinn [SB96] show that even the simplified problem of pulling a matte against a perfectly uniform background, such as extracting a matte from green-screen footage, is inherently ill-defined. To remove ambient bounce light, or screen spill, introduces further ambiguity. Thus, keying inherently requires an artist or model to discriminate between viable outputs.

Figure 1: *This footage of an astronaut walking contains tremendous amounts of screen spill from the floor and walls. Our method achieves the best result. Note the neutral spill suppression on the astronaut's suit, a perfectly transparent $\alpha = 0$ background as seen in the center strip, edge detail retention, and limited holes in the foreground. Footage courtesy ActionVFX.*

Screen removal methods generally depend on contrast between screen colors and foreground colors. This reduces ambiguity when extracting a matte for the foreground element. Several techniques exist for mathematically representing the decision boundary between foreground and background colors, adjusting edge pixels, and removing "spilled" bounce light from the screen, though the spill suppression problem has been studied little in academia.

Keyers continue to be used by visual effects artists over other techniques because they offer a high level of editability. They typi-

DOI: 10.2312/egs.20261022

cally apply a function independently to each pixel of the input image, which enhances the temporal stability of the result. Several of today's most popular keyers are *difference keyers* [Fou25a], which subtract each pixel of the image from an expected background color or chroma [SB96]. This difference is then mapped to a matte, which is used to remove screen spill. Difference keyers benefit from simplicity and ease of use. They tend to do well at extracting edge detail, but can struggle with uneven screen lighting.

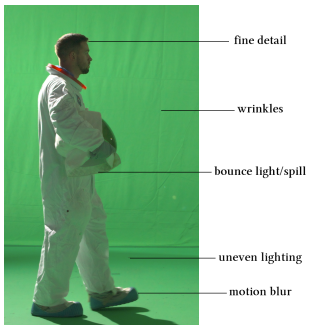

Figure 2: *Green screen footage often contains artifacts that complicate matte extraction and spill suppression. Footage courtesy ActionVFX.*

Our algorithm's logic most closely resembles Primatte [Mis94], a widely used keyer. Primatte generates a small set of nested polyhedra centered around a sampled dominant screen color in color space [Fou25b]. Each polyhedron delineates a foreground, despill, edge, or background color region. Users extend and retract the faces of the polyhedra by sampling the image. Primatte is more robust to uneven screen lighting than many difference keyers, but it sometimes struggles to extract fine matte detail from hair, motion blur, and translucent objects.

Energy-based methods [SJTS04, CZF*19], have proven effective for natural image matting. However, such methods have struggled with temporal consistency, computational cost, fine detail, and user control, and they do not address spill suppression.

Historically, difficult segmentation tasks required incredibly labor-intensive manual rotoscoping and painting. Machine learning-based techniques have replaced some manual rotoscoping; they produce impressive results for matte extraction and are already widely used in production [YWYL24, BDL*21]. However, they tend to be difficult to control and often require frame-by-frame adjustments to achieve production-quality mattes. Machine-learning matte extraction research has historically ignored the spill suppression problem, with a few exceptions [AAPS16].

Despite the pitfalls of individual keying and matte extraction algorithms, skilled compositors achieve plausible results by combining the outputs of several keyers [Wri17]. Thus, an algorithm with unique strengths constitutes a significant contribution to keying.

3. Methodology

The most popular keying algorithms perform well with clean green screen plates, but when presented with unevenly lit screens, motion blur, strong screen spill, and other challenges, they flicker, leave holes in the foreground matte, produce jagged edges, or introduce noise to the background. In contrast to prior methods, we do not assume that:

1. the screen color is highly saturated,
2. the screen is smooth and evenly lit, or that
3. there is only one screen color present.

By giving the user more control over the image and algorithm, we can mitigate undesired artifacts even when faced with difficult footage.

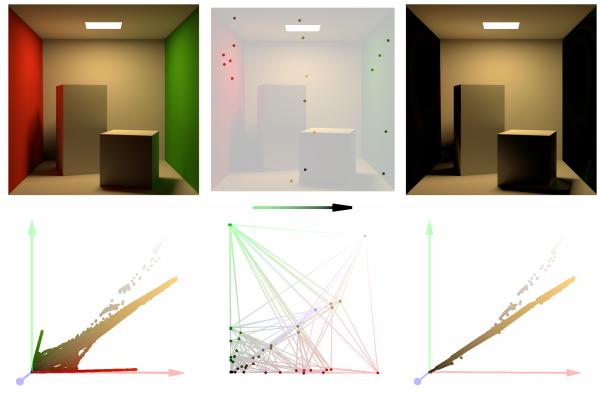

Figure 3: *The Delaunay Keyer is more robust to difficult plates than the commercially available keyers. On the bottom left, the distribution of colors in the original image is shown. On the bottom center, we generate a Delaunay mesh from user-specified sample points. On the right, the mesh is used for matte extraction and spill suppression.*

The algorithm outputs an image matte α and despilled image I'. After keying, we premultiply the image I' by α to allow compositing, but in practice both output layers could be combined with output from another keyer for more polished results.

The Delaunay Keyer requires as user input an input image I with values in a linear RGB color space and a set $\mathcal{A}$ of n sampled color triples $x_1, \ldots, x_n$. Two corresponding weights are required for each color triple: v for the desired matte value, and w to denote the sample's 'cleanliness' from screen spill. For a pixel with visible spill, the spill weight w is set to 0. In practice, most samples will have weights $w = v = 0$ for background pixels and $w = v = 1$ for clean foreground pixels.

We provide a few preset sets of color triples for common base cases, including weights for white, black, the primaries. Certain presets include additional samples for screen color, skin tones, and neutral gray. Additional samples can be taken from the input image or directly inserted into the mesh in a 3D view.

Using our sampled color triples in $\mathcal{A}$, we construct a Delaunay tetrahedralization $\mathcal{T}$ of the points, returning m tetrahedra $t_1, \ldots, t_m$, where each vertex of tetrahedron t_k is in $\mathcal{A}$ [Reb93].

Let C_i denote the color triple for pixel i in image I. Then for each C_i in I, we look up by which tetrahedron t_k the color triple C_i is bounded. We then calculate the normalized barycentric coordinates γ of C_i with respect to t_k., where $\sum\gamma = 1$. To calculate the matte at pixel i, which we will denote as α_i, we take the dot product of the normalized barycentric coordinates and the corresponding weights $w_0, .., w_3$ for the tetrahedron's vertices:

$$\alpha_i = \sum_{j=0}^{3} \gamma_j v_j. \tag{1}$$

To achieve spill suppression, we project the color within its respective tetrahedron in the direction of the spill weights:

$$\gamma'_j = \frac{\gamma_j w_j}{\sum_{j=0}^{3} \gamma_j w_j}. \tag{2}$$

We can then convert the new weighted barycentric coordinates back into Cartesian coordinates to return new pixel values with spill suppressed. For pixels outside of the mesh, we simply look up the properties of the nearest triangular boundary.

Algorithm 1 Delaunay Keyer

Input: $I, A, \{v_1, ..., v_n\}, \{w_1, ..., w_n\}$
Output: I', α
  $\mathcal{T} \leftarrow$ DelaunayTetrahedralization(A)
  for all $C_i \in I$ **do**
    Find tetrahedron t_k in $\mathcal{T}$ such that C_i is bounded by t_k
    $\gamma \leftarrow$ ToBarycentricCoordinates(C_i, t_k)
    $\alpha_i = \sum_{j=0}^{3} \gamma_j w_j$
    $\gamma'_j \leftarrow \frac{\gamma_j v_j}{\sum_{j=0}^{3} \gamma_j v_j}$
    $I'_i \leftarrow$ ToCartesianCoordinates(γ', t_k)
  end for

Because the algorithm only considers one local region of the color-space to produce a pixel's output, we can key a wide range of chromas or luminances out of the image simultaneously. The tetrahedral mesh implicitly forms internal surfaces representing core foreground and background color regions, and interpolates between these regions. In Figure 4(a), we visualize an example spill suppression for a green screen key.

In three dimensions, Delaunay tetrahedralizations can suffer from elements which take the form of small slivers when vertices are clustered, despite optimality guarantees [EG02], especially. While slivering is a significant problem for Delaunay tetrahedralization in mesh generation problems, in this case, it is usually trivial to adjust input samples to achieve a more even mesh. Our prototype allows the user both to sample colors from the image in a 2D view and to insert or adjust vertices in a 3D mesh view.

Finally, we also allow the user to blend between the original image and the keyed image independently for chroma and luminance to temper the effects of spill suppression as desired.

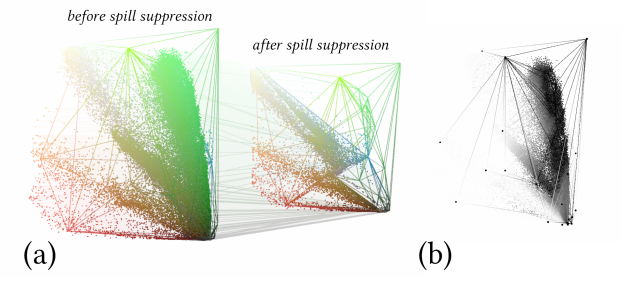

Figure 4: *An example of a sample mesh from a green screen image. (a) Colors are projected locally. (b) Alpha values are interpolated over the tetrahedra. In tetrahedra where all four vertices are background samples, sampled colors are assigned* $\alpha = 0$.

4. Results

Given that unevenly lit green screens and heavy screen spill are an unfortunate reality of visual effects productions, our keyer has the potential to reduce keying difficulty in real-world production scenarios. In Figure 1, which contains extreme amounts of screen spill, fine edge detail, and uneven lighting, our keyer obtains comparable or superior edge detail, spill suppression, and core mattes to the other keyers. The keyer is capable of plausible spill suppression, assigning neutral grays to the screen without losing saturation on the astronaut's cyan boots.

We compared to the keying algorithms most used in visual effects productions, including the keyers Primatte [Mis94], Keylight [Fou25a], Ultimatte, and the Foundry's Image-Based Keyer.

The Delaunay Keyer is more robust to variance in screen luminance than any of the keyers to which we compared. When a screen is unevenly lit, its dark and light areas appear less saturated to a standard keyer. In our tests, the standard keyers either fail to extract fine detail or fail to achieve a solid core foreground and a

Figure 5: *Our keyer obtains plausible and art-directable spill suppression. In this case, we have projected away from all green screen color. Compare Figure 2.*

completely black ($\alpha = 0$) background. With some careful adjustment, our keyer is both able to regress a plausible edge matte and achieve solid foreground and background core mattes as well or better than the other keyers. We find that 20 to 40 samples is typically sufficient for reasonable results.

The colorspace-local nature of the interpolation method allows the keyer to remove multiple chromas at the same time, as can be seen in the unrealistic yet demonstrative case of the Cornell Box as seen in Figure 3. Note that spill from the highly saturated walls has largely been removed from the ceiling and the outward-facing sides of the foreground boxes. No other keyer was able to remove and despill both sides. As a more practical example, see Figure 6, where our keyer pulls a key for the sky. Other keyers were not able to remove both blues and whites, and all other keyers struggled with temporal consistency.

Our Houdini-based prototype processes 1080p video with 40 samples at ~1fps on an Intel i9 CPU. The most expensive step is searching the tetrahedral mesh for the bounding tetrahedron for each pixel. This problem is well-suited for modern GPU architectures, so realtime implementation should be possible.

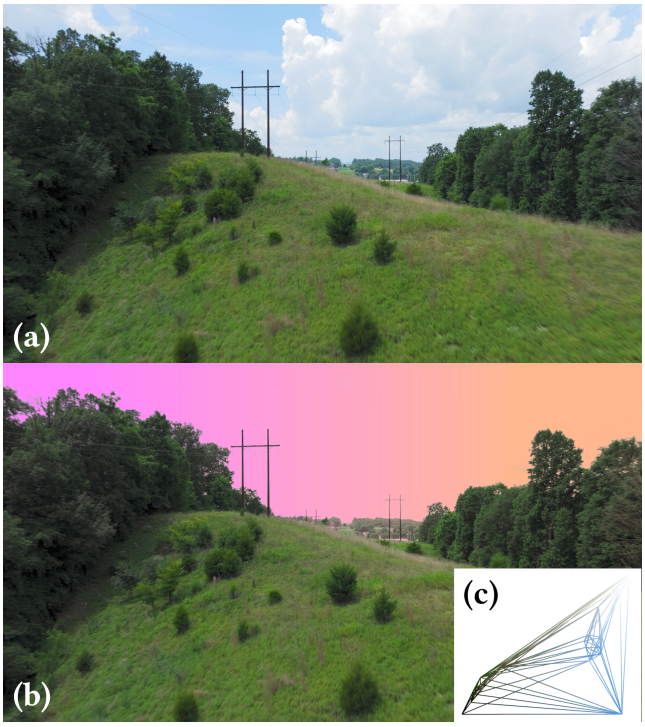

Figure 6: *(a) Keyers are sometimes used for sky replacements. No other keyer obtains good results for this image. (b) Our keyer is able to remove both blues and whites, also suppressing the blue "spill" or atmosphere on the hills in the background. It also protects correct colors on the edges of the leaves in the midground. Sample mesh is shown in (c) Footage courtesy ActionVFX.*

5. Conclusion

We present a novel keying technique that relaxes implicit and unnecessary assumptions in industry-standard algorithms. By constructing a Delaunay mesh from user samples, we suppress spill and obtain a matte by interpolating over the generated tetrahedra locally in color space. This approach generates plausible results for a wide variety of green screen problems, especially edge cases neglected by existing methods. Our method enables fine art direction of spill suppression, handles multiple background colors simultaneously, and keys successfully across a wider variety of background luminances and chromas.

References

[AAPS16] AKSOY Y., AYDIN T. O., POLLEFEYS M., SMOLIĆ A.: Interactive high-quality green-screen keying via color unmixing. *ACM Trans. Graph. 35*, 5 (Aug. 2016). URL: https://doi.org/10.1145/2907940, doi:10.1145/2907940. 2

[BDL*21] BERMUDEZ L., DABBY N., LIN Y. A., HILMARSDOTTIR S., SUNDARARAJAN N., KAR S.: A learning-based approach to parametric rotoscoping of multi-shape systems. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2021), pp. 776–785. doi:10.1109/WACV48630.2021.00082. 2

[CZF*19] CAI S., ZHANG X., FAN H., HUANG H., LIU J., LIU J., LIU J., WANG J., SUN J.: Disentangled image matting, 2019. URL: https://arxiv.org/abs/1909.04686, arXiv:1909.04686. 2

[EG02] EDELSBRUNNER H., GUOY D.: An experimental study of sliver exudation. *Engineering with Computers 18* (2002), 229–240. URL: https://api.semanticscholar.org/CorpusID:2880270. 3

[Fou25a] FOUNDRY:, 2025. URL: https://learn.foundry.com/nuke/content/reference_guide/keyer_nodes/keylight.html. 2, 3

[Fou25b] FOUNDRY: Explanation of how primatte works. URL: https://learn.foundry.com/nuke/content/comp_environment/keying_with_primatte/how_primatte_works.html. 2

[Mis94] MISHIMA Y.: Soft edge chroma-key generation based upon hexoctahedral color space, Oct 1994. Primatte. 2, 3

[OZ20] OKUN J., ZWERMAN S.: *The VES Handbook of Visual Effects: Industry Standard VFX Practices and Procedures*, third ed. Focal Press, London; New York, 2020. 1

[Reb93] REBAY S.: Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-watson algorithm. *Journal of Computational Physics 106*, 1 (1993), 125–138. doi:https://doi.org/10.1006/jcph.1993.1097. 2

[SB96] SMITH A. R., BLINN J. F.: Blue screen matting. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, Association for Computing Machinery, p. 259–268. URL: https://doi.org/10.1145/237170.237263, doi:10.1145/237170.237263. 1, 2

[SJTS04] SUN J., JIA J., TANG C.-K., SHUM H.-Y.: Poisson matting. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 315–321. URL: https://doi.org/10.1145/1015706.1015721, doi:10.1145/1015706.1015721. 2

[Wri17] WRIGHT S.: *Digital Compositing for Film and Video*. Taylor & Francis, Nov 2017. 2

[YWYL24] YAO J., WANG X., YE L., LIU W.: Matte anything: Interactive natural image matting with segment anything models, 2024. URL: https://arxiv.org/abs/2306.04121, arXiv:2306.04121. 2