Depth keying

Ronen Gvili, Amir Kaplan, Eyal Ofek and Giora Yahav

3DV Systems Ltd.

ABSTRACT

We present a new solution to the known problem of video keying in a natural environment. We segment foreground objects from background objects using their relative distance from the camera, which makes it possible to do away with the use of color for keying.

To do so, we developed and built a novel depth video camera, capable of producing RGB and D signals, where D stands for the distance to each pixel. The new RGBD camera enables the creation of a whole new gallery of effects and applications such as multi-layer background substitutions. This new modality makes the production of real time mixed_reality video possible, as well as post- production manipulation of recorded video.

We address the problem of color spill - in which the color of the foreground object is mixed, along its boundary, with the background color. This problem prevents an accurate separation of the foreground object from its background, and it is most visible when compositing the foreground objects to a new background.

Most existing techniques are limited to the use of a constant background color. We offer a novel general approach to the problem with enabling the use of the natural background, based upon the D channel generated by the camera.

Key Words: Depth key, compositing, chroma key, matte creation, alpha channel, blue screen, color spill

1. DEFINITIONS

Throughout this paper we shall refer to the objects of interest as the *foreground*, and to the objects that we want to exclude as *background*, regardless of their actual position in the scene.

An *alpha channel* (known also as *matte* or a *key*) is a grayscale mask channel corresponding to a color channel. The alpha channel has full value pixels representing foreground pixels in the color channel, zero valued pixel representing background. Typically, the alpha channel is implemented using 8 bit values. In this paper we will refer to the alpha values as the fraction between 1 (full opaque) and 0 (transparent). Fractional alpha represent degrees of semi-transparency or a mixed pixel, partially covered by the foreground object.

A *trimap* is a segmentation of the frame into three regions: “Definitely foreground object” (full alpha), “definitely background” (zero alpha), and “unknown” zone. The common use of the trimap is to locally define the colors of the foreground object and background. The boundary of the “Definitely foreground” area, in the frame, locally defines the foreground color, while the contour of the “definitely background” defines the background color locally.

2. INTRODUCTION

The art of compositing, by mixing several video (or film) sources, some photographed, others created by graphic artists, in order to form a single video composite is an important part of video production.

The most common technique used for live action matting is known as *chroma key*. The method is extensively documented in the literature[4,6,7]. Foreground objects are placed in front of a screen of constant selected reference color;

in most cases it is blue or green. A camera captures the image of the foreground object and the colored background. A new background then replaces the colored background. The chroma-key technique has the advantages of being simple to comprehend and yield a very good quality images, which makes it popular. The method has several disadvantages:

1. **The need for special background**. The matte is generated by color-based segmentation, which isolates the foreground objects from theirs background using the color as a data. Smith and Blinn[7] formalized this problem and proved it has infinite solutions. Several approaches minimize the problem dimension to solvable cases. The main approach to the matting problem is using a constant background color, which does not appear in any of the foreground objects. There is much effort placed in using special paints and special lighting to create a background as homogenous as possible. In particular there is no possibility to use natural background. The need to use a special studio colored by the chroma key color, limits the usability of the technique.
2. **No partial replacement.** The background of a chroma-key frame has to be completely replaced by a new video (unless the intention of the production is to show a areas of constant color). There is no way to perform just partial replacement, such as insertion of a moving graphic object behind the foreground object, while maintaining the original background in all the pixels, that are not covered by the graphics. As so, there is no way to composite semi-transparent graphics, showing the original background through it.
3. **Foreground color problem.** Foreground objects must not contain colors similar to the color used for keying. Even small contamination of the pixel color by background color can be interpreted as semi-transparency.
4. **Segmentation into 2 layers only.** Using chroma key with a single background color, the segmentation of the frame is limited to two layers only: a foreground layer and background layer (chroma key colored) that will be replaced completely by a new image. No farther separation between foreground objects or actors is possible.

Most of these problems rise from the use of the color signal for the generation of the key. This limits the use to special videos including the keying signal. Moreover, the keying signal cannot be simply filtered out without affecting the image. Ben-Ezra[1] uses light polarization as an invisible keying signal. Although he shows great improvement in solving the color spill problem, there is a need for special polarization preserving background in the scene for generation of the key. The need for a special background retains many of the disadvantages of the chroma key.

The commercial matting application *Knockout*[2] by Procreate can generate mattes of still frames shot in a natural environment. The tool uses a trimap that is manually defined by the user. The foreground and background colors, defined locally using the trimap contours, are used for alpha generation using a chroma key. Knockout will fail if either the foreground or the background color changes inside the “unknown” zone. To achieve best results, the foreground and background contours should be set as close as possible to the edge of the object, minimizing the “unknown” area. The extensive user interactive limits the use of knockout for still frames only.

Chuang et al[3] describes a Bayesian matting method for solving the matting problem, given an initial trimap. They model the background and foreground color distributions with mixtures of Gaussians, assume fractional blending of foreground and background colors, and estimate their optimal values using a maximum-likelihood criterion. The user draws an initial trimaps at specific key frames, which are interpolated across the video using forward and backward optical flow. Ruzon and Tomasi[8] suggest quite a similar approach. Although this framework seems to make the processing of large video sequences manageable, it is still an offline process, which depends on manual input, thus far from real time implementation.

The color spill problem

Removing the background color spill, is a problem that needs special treatment when handling unconstraint backgrounds. Color spill is the contamination of the foreground objects with background colors. This problem is most visible at the edges of foreground objects, where a pixel can accumulate light coming from both a foreground object and the background behind it, or when foreground objects are semi-transparent, letting light returning from the background pass through it and reach the camera.

There is a need to clean the original background color, before compositing the foreground objects on a new background. For example, suppose the pixel $\boldsymbol{I}(i,j)$ is semi-transparent with a key α ($0<\alpha<1$). This means that the color of pixel $\boldsymbol{I}(i,j)$ in the original image is a mix of the true foreground color at that pixel, $\boldsymbol{F}(i,j)$, and background color $\boldsymbol{B}(i,j)$,

$$(\boldsymbol{I}(i,j) = \alpha*\boldsymbol{F}(i,j) + (I-\alpha)*\boldsymbol{B}(i,j)).$$

Simply compositing this image on a new background $\boldsymbol{N}(i,j)$ with a key of α, results in a wrong composite containing $\alpha*(1-\alpha)>0$ of the old background color.

$$\boldsymbol{newI}(i,j) = \alpha*(\alpha*\boldsymbol{F}(i,j) + (I-\alpha)*\boldsymbol{B}(i,j)) + (1-\alpha)*\boldsymbol{N}(i,j).$$

Figure 10a shows a blue circle in front of a red background. Figure 10b shows the matte used for generation of this composite. Figure 10c shows a composite of the original image on a new white background, using the correct matte, but with no background color spill removal. Red residues are clearly visible around the edges of the circle.

In order to generate the correct new composite we need to recover the pure foreground color $\boldsymbol{F}(i,j)$, by removing the background color spill. Estimating the background color is an ill posed problem, since at mixed pixels we have six unknowns ($\boldsymbol{F}(i,j)$, $\boldsymbol{B}(i,j)$), but only 3 equations (for each of the components of $\boldsymbol{I}(i,j)$). The common solution used today in a chroma key scenarios, uses a background of a very limited color. These solutions cannot be applied to the general case of keying in a natural background of varying color.

Our contribution

3DV has developed and patented a robust method for live keying. The proposed keying is based on complete data of the field of view (FOV): both the color data and the distance data for each pixel. The depth information is captured by a novel depth video camera. The camera allows real time generation of a matte in natural environment, without limitation of the scene background or foreground colors. Moreover, depth distance captured by the camera, allows the parallel generation of multi-layered keys, each being set at some specific depth.
The concept behind the camera is very versatile and may be applied to many fields, such as security, robotics and gestures recognition, to name a few.

When generating a matte in a natural scene, there is a problem of fixing the color spill caused by a background of varying color. We propose a simple and fast method for color spill removal along the boundaries of foreground objects.

The remainder of the paper is organized as follows. Section 2 describes our new concept of depth keying and the depth camera developed to generate it. Section 3 outlines the algorithm for background color spill corrections for scenes with natural background. Section 4 contains conclusions.

3. DEPTH KEY

3.1 Depth camera

The basic tool for keying without resorting to a chroma-key is a special camera that delivers both color and depth (distance from the camera) for each pixel in the FOV[5]. This unique camera is capable of doing so at video rate and is compatible with all existing standards and formats.

The concept of operation is based on generating a "light wall" having a proper width moving along the FOV. The said “light wall “ can be generated, for example, as a square laser pulse of short duration having a field of illumination (FOI) equal to the FOV (Figure 1a). Non-visual illumination is used so that it does not interfere with the visual video

content. As the light wall hits the objects in the FOV, it is reflected back towards the camera carrying an imprint of the objects (Figure 1b). The imprint contains all the information required for the reconstruction of the depth map.

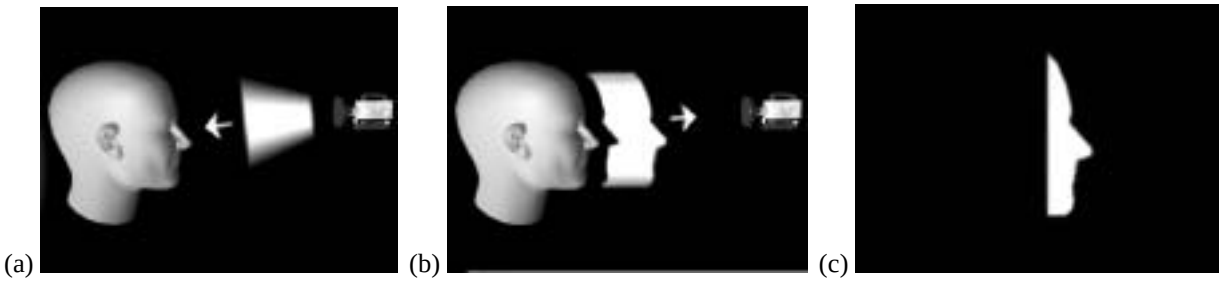

Figure 1. a) “Light wall” moving along the FOV. b) Imprinted light wall returning to camera. c) Truncated "light wall"- front cut.

The depth information can now be extracted from the reflected deformed "wall", by deploying a fast image shutter in front of the CCD chip, and blocking the incoming light as shown in figure 1c.

The collected light by each pixel is inversely proportional to the depth of the specific pixel. As an alternative, it is possible to retain the rear section of the reflected wall, and thus to receive the “negative” of the depth.

Since reflecting objects may have any reflectivity coefficient, there is a need to compensate for this effect. Therefore a normalization procedure is introduced. The normalized depth of pixel $\boldsymbol{D(i,j)}$ can be calculated by simply dividing the front portion pixel intensity $\boldsymbol{I}_{front}(i,j)$ by the corresponding portion of the total intensity $\boldsymbol{I}_{total}(i,j)$: $\boldsymbol{D(i,j)} = \boldsymbol{I}_{front}(i,j) / \boldsymbol{I}_{total}(i,j)$.
Hence resulting reflectivity normalization procedure, which completes the capture of the depth map.

The main technologies on which the camera is based on are:
1. Fast switching of the illumination source to form the “light wall”.
2. Fast gating of the reflected image entering the camera.

A cluster of IR laser diodes and corresponding optics is used to homogenously illuminate the FOI. The diodes are switched on/off with a rise/fall time, which is shorter then 1 nsec.

For the fast image gating, our present camera incorporates a specially designed solid-state shutter. This shutter is mounted in front of the CCD, precisely controlling the exposure time of the CCD for each pulse.

An optical design and spectral separation is used, so the same shooting lens is used for both the depth sensor as well as a regular color broadcast camera, generating simultaneous video streams of color and depth videos. Figure 2 presents the block diagram design of a studio depth camera.

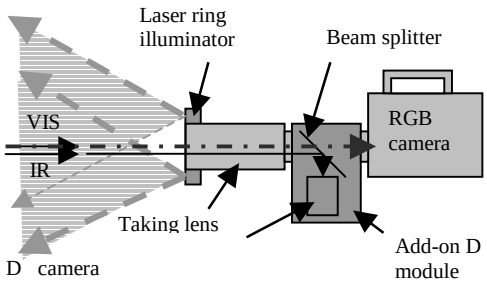

Figure 2: Main Building blocks of 3DV’s ZCam™ add-on for studio cameras

Figure 3: ZCam™ add-on to broadcast camera by 3DV Systems

As shown the laser “light wall” emerges from the illuminator to form the FOI. The taking lens gathers the reflected light forming the FOV, which can be zoomed if so desired. The D-module splits the incoming light and the visible portion of the light is sent to the color studio camera, while the IR portion is reflected towards the depth camera, which delivers the depth video.

A photograph of a ZCam™, an add-on for a broadcast camera that uses the depth imaging technology is shown in Figure 3.

3.2 Depth key setting

One of the most important features of the depth camera is the ability to change the parameters of the depth window dynamically. The depth window can be set to include certain objects, while disregarding objects that are outside the window, according to the scenario’s need.

Figure 4. Dynamically setting the depth window, controls the volume measured by the camera.

Figure 5. A color frame and its corresponding depth frame. Background objects are outside the depth measurement window.

A simple key of foreground objects can be generated by setting the depth measurement window to include the ranges, in which the foreground objects are located (see Figure 4). In such setting, the camera captures light reflected from every object inside the depth measurement window, and ignores light reflected from objects outside the window (Figure 5). A pixel is considered in the key if the amount of energy received by that pixel is above a certain threshold. All background objects (or objects that are too close to the camera) do not contribute light, and do not affect the keying process. Figure 11 shows several frames from a live weather broadcast, where the graphics is composited behind the weather caster in a natural environment using a depth key.

In many common cases, the object to be keyed, is the object closest to the camera. In such a case, the depth key setting can be automatically set, by gradually moving the depth measurement window away from the camera, starting from zero distance from the camera, until an object is entering the measurement window. Such automatic segmentation of the scene is possible since different objects in the scene have coherent placement in space. This is not the case for chroma keyers, since objects in the scene do not have, in general, coherent color, thus a manual segmentation setting is needed.

Additional keys, that segment between foreground objects inside the depth window can be generated by further slicing the generated depth map according to some depth criterion. In Figure 6, we generate multiple keys, one for each girl, by mapping different depths to different key values.

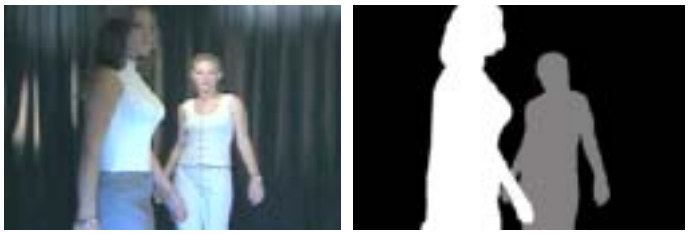

Figure 6. Multiple keys by depth slicing.

a. Handling the edges of foreground objects

Using a separate signal for key generation has great advantage: there are no limitations on the original color video. On the other hand, since the compositing procedure uses the color video, there is a need to suit the depth key to the color frame. There are several factors that contribute to possible differences between the depth matte and the color frame:

- Color smear is common in video. The smear, visible near objects boundaries, is caused due to video color compression and the difficulty of analog video to represent sharp edges.
- Depth matte may be a bit smaller than color image near the object edges, due to possible reduction in the amount of light returning from the objects at grazing angles.
- Mixed pixels around the foreground object boundary. These pixels are only partially covered by the foreground objects, letting the background to be partially visible through them. It is common to represent such pixels with an alpha value between 1 (total opaque) and 0 (full background). The depth mask generated, is a segmentation of the frame pixels into foreground or background ones according to the depth of the pixel. Although mixed pixels reflect less light back to the camera, the change in reflectance is normalized as a part of the depth computation process, just as any other change of scene albedo. In fact, the transparency quality of an object can only be recovered by looking at the interaction of an object with its background and cannot be understood by simply looking at the object by itself[1]. In the simple scenario where only foreground objects are inside the depth measurement window, there is no means to determine that an object is semi-transparent.

With color video, on the other hand, semi transparency is easily identified, once the foreground and background colors are known. Unfortunately, recovering these colors from a color frame is an ill-posed problem. Recent methods[2,3] exploit local spatial coherence of the scene color to estimate the foreground and background color using a fairly accurate trimap of the scene, created manually. This is both time-consuming and labor-intensive process.

To overcome these differences, a new matte is generated, combing the information from both the depth and the color frames. This new matte will represent both mixed pixels and compensate for color-depth differences. A real-time trimap is generated for each frame, based on the original depth matte. Guided by this trimap, the local transitions of color near the object edges will be used to generate a key closer to the color frame. The processing is taken place at a very localize area of the frame, near the edges, enabling generation of a key in real-time.

The process is based on the following steps:

a. A depth mask is generated by threshold the depth map.
b. *Recovery of the contour of the foreground objects.* A chain code representation of the boundaries of the mask is constructed. This is the only time during the entire process that a full scan of the frame is done.
c. *Definition of the region being processed*. The processing takes place in the vicinity of the foreground object boundary. A rectangle of specific dimension is centered at each chain code point. The processing region is the union of all these rectangles. A common size used is 7 by 7 pixels. Figure 7 shows the original video field (dimension of 720x288 pixels) and corresponding depth mask. Figure 8 shows the region for processing, located around the object boundary.

a  b 

Figure 7: a) Original field of the video. B) Corresponding binary depth mask.

Figure 8: Processing of the mask is done in a compact area around the object boundaries.

The rest of the processing is limited to the area of the boundary region, *R*, as defined above.

At the boundaries of the defined region, pixels are assumed to be either completely inside the foreground object (opaque alpha, pure foreground color), or completely outside the object (transparent alpha, pure background colors). Inside the region, there are pixels that might be a mix of those colors that should be represented by a semi-transparent alpha value. The definition of this alpha is propagated iteratively across the region, using the assumption that pixels, which have similar color values, will probably have a similar alpha value.

d. *Calculate color distances between neighboring pixels*:
For each a pixel $p \in R$ and for each pixel q of its 3x3 neighborhood, N_p, we define a weight

$$W_i = \frac{1}{\sqrt[4]{D_i}},$$

where D_i is the distance (in RGB color space) between p and q. The 4th root was found suitable to allow for similar colors to be considered close in the presence of video noise. The weights are normalized so that the sum of the weights between p and its 8 neighbors is 1. The initial alpha value of p is either 0 or 1 according to the binary mask.

e. For each a pixel $p \in R$, the new value of the alpha corresponding to the pixel p, is the weighted sum of the alpha values of its neighborhood:

$$\alpha_p = \sum_{q \in N_p} W_i \alpha_q$$

f. Step e is repeated several times until the mask distribution is similar to the color transitions. Given the basic mask quality, 3 iterations where found to be sufficient in most cases.

Figure 13 shows an inset of an original mask (b) and the mask after processing (d). Each boundary pixel of the processed mask contains an alpha value according to the amount of its color similarity to the local foreground colors. Notice that sharp edges, such as those visible at the bottom right side of the frame, are retained although the colors of the foreground and the background color are quite close.

Discussion:

It is very important that the processing is held on a region R of minimal area. A smaller region would contribute to the speed of the procedure, but even more important – it keeps the validity of spatial coherence assumption. All methods that use trimaps rely on the assumption, that both the foreground colors and the background colors do not change inside the area being processed. The only factor that is assumed to contribute to the change of image color in that area is the change of alpha value.

When the trimap is defined manually, as being done by other methods, an operator can assure that this is indeed the situation. Whenever there is a change of foreground color near the edge of the object, the operator will define the complete-opaque contour closer to the object edge, forcing the change of foreground color to be outside the processed zone. The same is done for the background. In this paper, the entire processing is done in real-time without any manual intervention, so a special care is taken to minimize the transition area between foreground and background. Sharp edges have small transition from foreground to background (1-2 pixels), thus there is no need for a large processing area, on the other hand, fuzzy edges require processing larger area since the transition is more gradual. The sharpness of the edge is measured by calculating the derivative of the color image intensity at a direction orthogonal to the edge. Figure 9 shows the processed area around the contour of a foreground object.

Figure 9: Unknown zone width is proportional to the amplitude of the gradient at the object boundary. The lower the gradient is, the slower the transition from foreground to background, and therefore, a larger unknown zone is defined. (The zone width is exaggerated for this figure).

The suggested process produces good results for objects with moderate fuzziness: transition between foreground and background should be limited to the processed area only. Current implementation has a maximum process area of 7 pixels, centered on the position of the original mask. Objects with fuzziness greater than that will not generate a correct mask.

The processing is computationally efficient: it is preformed on a limited area of the frame, around the object boundaries, and it is highly parallel.

The process can be improved using Bayesian methods[3,10], as long as it is valid from performance point of view.

4. COLOR SPILL CORRECTION

The new generated matte, as described in the previous section, represents mixed pixels at the edges of the foreground objects. Mixed pixels color is a result of a mix of the foreground object color and the color of the background used in the original video. Before the video can be used to be composed on a new background, pure foreground color should be recovered; otherwise residues of the old background colors will be visible in the composite (Figure 10).

Recovery of the foreground color of a single pixel is ill posed: each pixel contributes 3 equations of 6 unknowns (3 of the unknown foreground color and 3 of the unknown background color). To solve this problem we assume local spatial coherence of the foreground and background colors.

Suppose p is a mixed pixel, for which a prior foreground color (F_p) will be recovered. The visible (composite) color of p is a mix of the foreground and background colors at p :

$$C_p = F_p * \alpha_p + B_p * (1 - \alpha_p).$$

Where α_p is the alpha value at that pixel, and B_p is the background color.

Let N_p be the *k* x *k* neighborhood of p .

It is assumed that $F_q = F_p$ and $B_q = B_p$ are constant for each pixel *q* of *Np*. Since α is given, each pixel will contribute 3 equations and so 2 pixels of different colors are enough in order to solve the equations. In practice, a larger neighborhood is used and a foreground color is found by minimizing least square differences. A special care is taken, when the entire neighborhood has a similar alpha value, since it may lead to a degenerate equations system. This situation rarely occurs near object edges, but in such a case it is better to use the original depth key mask. The same is done when the foreground and the background colors are very similar, which makes any matte based on a chroma key highly unreliable.

A neighborhood of 5 x 5 was found to be adequate: a smaller neighborhood might reduce computation accuracy due to small variance of α . Yet, on the other hand, the assumption of constant foreground and background colors might not hold for larger neighborhoods.

A new frame is generated, where each mixed pixel, along the matte edge, has its color replaced by the recovered foreground color.

From a computational point of view the locality of the processing enables a highly parallel implementation. The main load of this algorithm is a high number of read calls for each pixel (25 calls as the size of the neighborhood), each consist of 4 channels (RGB and alpha). On the other hand, it is highly suitable for parallel computation. Using the SIMD (Single Instruction /Multiple Data) instructions set, a real-time computation of this algorithm was implemented on a Pentium 3 machine. The MLS (Mean Least Square) calculation uses a moving window summing which accelerates the computation.

Figure 13 (e) shows the compositing of a foreground object on top of a black background. Notice the residues of the wall gray-brown color.

5. SUMMARY

A new type of keying was introduced: the *depth key*. Depth key is based on the assumption of space-coherence of scene objects. Objects are mostly located in a compact volume of space, and can be segmented according to different distances from the camera. This assumption is much more fundamental than the assumption of color coherence of objects, on which all chroma-key methods are based.

To generate a depth key we presented a novel camera that measures both color and depth information of the visible scene in real-time. ZCam™, which is a commercial product, was used to generate depth keying during live broadcasts in different environments. Cameras, using depth-sensing technology are used in a variety of fields, such as man-machine interactions, game input devices, gesture recognition, video conferencing, automotive industry and more. Additional information may be found on our web site (http://www.3dvsystems.com).

The depth keying method is based on the use of a separate signal for keying, rather than using the color signal. The method segment objects regardless of the object color or textures, while preserving original colors. The key can be generated without any limitation on the keyed object background: it can be outside, in a natural surrounding. In fact, the depth key can segment an object even if it is completely similar to its background color. On the other hand, since

the key is needed for compositing the color video, additional processing is done to assure compatibility of the depth matte edges to the edges of the color frame.

Using the depth information, it is possible to generate more than one matte, while each is related to a different distance from the camera. In fact, when compositing a three-dimensional object, it is possible to generate a matte by comparing the depth of the object and the depth of the scene on a pixel basis. This enables the generation of complex visibility composition, such as seen in figure 12. The girl is both occluding the virtual object and at the same time is being occluded by it.

A depth key can be generated in a natural environment, thus can be used for compositing graphics on part of the background or graphics with semi transparencies, as seen in figure 11. The depth keying is done in real-time without the need of operator supervision. In some cases, such as keying the object closest to the camera, it is possible to locate the target and set a key automatically.

6. ACKNOLEDGMENTS

The authors would like to thank Boris Epshtein for his help and remarks, and would like to thank KPIX CBS channel 5, San Francisco for the permission to use their footage.

7. REFERNCES

1. M. Ben-Ezra, “Segmentation with invisible keying signal”, *Proc. Of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp 32–37, 2000
2. A. Berman, A. Dadourian, and P. Vlahos, “Method for removing from an image the background surrounding a selected object”, *US Patent 6,134,346*, 2000
3. Y. Chuang, A. Agarwala, B. Curless, D.H. Salesin, and R. Szeliski, “Video matting of complex scenes”, *Proc. Of SIGGRAPH ‘02*, pp 243-248, 2002
4. K. Fukui, M. Hayashi, and Y. Yamanouchi, “A virtual studio system for TV program production”, *SMPTE Journal*, June 1994.
5. G.J. Iddan, and G. Yahav, “3D imaging in the studio (and elsewhere…)”, *Proc. Of SPIE 4298: Videometrics and Optical Methods for 3D Shape Measurements*. pp 48-55, 2001
6. R.F.H. McCoy, “Television images positioning and combining systems”, *U.S. patent #4,393,394*,1983.
7. A.R. Smith, and J.F. Blinn, “Blue screen matting”, *Proc. Of SIGGRAPH ‘84*, pp 259-268, 1996.
8. M.A. Ruzon, and C. Tomasi, “Alpha estimation in natural images”, *Proc. Of CVPR 2000*, pp 18–25, 2000

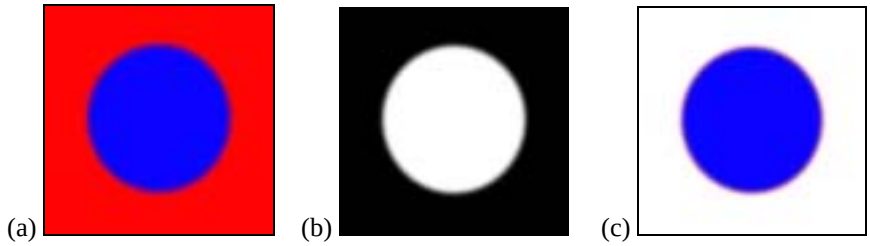

Figure 10: a) Original image. b) Matte. c) Composite on a white background reveals red spill.

Figure 11. Depth keying in a natural environment.

Figure 12: Depth key of 3D object.

.

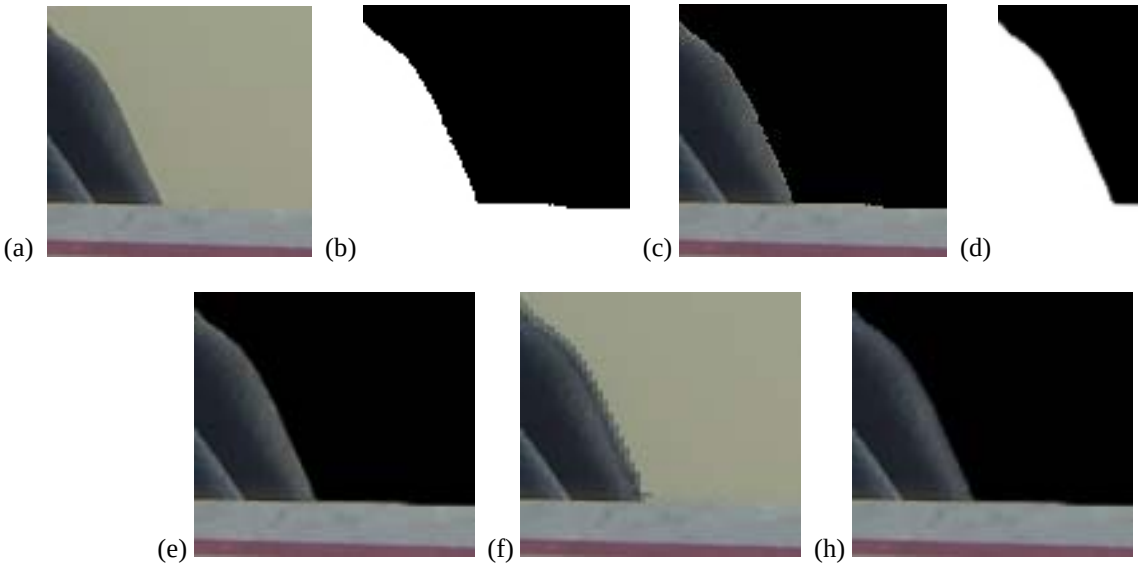

Figure 13: a) An enlarged inset of a frame b) original binary depth mask. c) Foreground object composite onto black background using the binary mask. d) Mask after processing. e) Composite using processed mask with background brown color residues on the edge of the blue coat. f) Foreground color is extrapolated cleaning out background color. h) Composite using processed mask and cleaned color.