STAG: Smart Tools & Apps for Graphics (2015)
Silvia Biasotti and Marco Tarini and Andrea Giachetti (Editors)

A Semi-automatic Algorithm for Applying the Ken Burns Effect

D. Allegra and F. Stanco and G. Valenti

University of Catania, Dipartimento di Matematica e Informatica, Viale A. Doria no. 6, 95125 Catania, Italy
{allegra, fstanco}@dmi.unict.it

Abstract

In historical documentaries, video material often is not available. For this reason they are mainly made by using static material such as old photographs. To make this material more endearing and dynamic an effect known as "Ken Burns Effect" can be applied to the static images. It consists in a mix of panning and zooming effect applied to different objects which belong to an image. Hence, considerable user experience with photo and video editing software is required to successfully separate the objects from the background and to animate them to produce a high quality result. In this paper, we present an algorithm to apply Ken Burns effect with a minimal user interaction. The proposed solution exploits Statistical Region Merging segmentation algorithm to support the user in the process of separation of the objects from the background. Moreover, Inpainting algorithms are employed to fill the empty regions which becomes visible when an object is moved from its original position. Finally a random video can be produced from different "animated" images.

1. Introduction

The Ken Burns effect consists in a smart combination of moving and zooming effect. The origin of the name is related to Lauren Kenneth (Ken) Burns, an American director who has been a massive use of panning and zooming effect on static historical photographs in order to produce documentaries. The effect includes a mix of different moving and scaling transformation which are gradually applied to an image. This allows creating a video with photographic material in a dynamic modality [ken10]. It can further increase the dynamism by separating the various objects of an image and by applying the panning and zooming effects to each of them. This approach, also gives the illusion of a third dimension thanks the parallax scrolling principle [GD10]. In this way, the image appears more similar to a real video.

Despite there are many software to apply Ken Burns effect to the images, they do not allow to apply the effect to single object to exploits the parallax effect. To this aim, it is necessary to use photo and video editing software to manually perform the image segmentation and the objects animation. Openshot and iMovie software for Linux SO include a transition effect called "Ken Burns", which allows to incorporate an image into a video by using a sort of slow panning and zooming. Final Cut Pro [KF03], Apple TV and Apple's iMovie video editing programs have a photo slideshow option labeled "Ken Burns Effect". On the Windows platform, 4K Slideshow Maker by 4KDownload, AVS Video Editor, Windows Movie Maker, Pinnacle Studio, Serif MoviePlus, Sony Vegas Studio (and Movie), Ulead VideoStudio, Adobe Premiere, PicturesToExe also have the pan and zoom features. Ken Burns Effect can be a native feature or can be available through third party plug-in which may be used to achieve the effect. Microsoft Photo Story is a free application which is used to create automatically videos with both random and custom pan and zooming from the selected images. Other software are ProShow Gold/Producer and PhotoFilmStrip. For the Mac platform, Final Cut Pro, Final Cut Express, iMovie, Adobe Premiere and other also have this feature. All aforementioned software, that allows an automatic application of the effect, consider the entire image only. There is no possibility to treat the objects in the image separately to produce the parallax illusion. Other programs, such as Adobe Premiere, provide the tools to separate the various objects from the original image and to apply the effect on each of them. However, the task has to be performed manually by the user. However, several works to apply a sort of $3D$ effects from $2D$ visual data exist in literature. In [ZCA*09], the authors proposed a method to simulate a sense of $3D$ parallax by using a light field with depth-

DOI: 10.2312/stag.20151296

information. The algorithm is able to create a $3D$ pan and scan cinematic effect, in accordance with typical technique used in documentary films. Świrski et al. [SRD11] implemented an algorithm for automatically applying Ken Burns effect. Nevertheless, their approach requires depth information, which is acquired through depth sensors (e.g., Kinect) or it is computed from stereoscopic images.

In this paper we propose an algorithm which leads a considerable speed-up in the user work to apply the effect on separate objects in the image. The idea, is to use segmentation and inpainting algorithms to help the user to separate the meaningful objects from the original image and fill the empty region left. The proposed approach does not need depth information, so it is possible to apply the Ken Burns effect to the historical photographs. The paper is organized as follows: in Section 2 the proposed algorithm is described in details. In Section 3 we discuss the obtained result on historical photographic material. Finally, Section 4 concludes the paper with hints for future works.

2. Proposed Method

The proposed algorithm consists in four different main phases, which require few little user interaction. In the first phase we exploit Statistical Region Merging (SRM) algorithm to perform a first raw segmentation. Moreover, two smart tools to refine the segmentation results are provided. At the end, each object will be placed in a different layers. Second, the created layers have to be ordered to give to the user a three dimensions illusion in the final video, accordingly with the parallax scrolling technique. In the third phase, all the empty regions in the background layers have to be filled. To this aim we employed two inpainting algorithms: a data estimator based on Discrete Cosine Transform (DCT) [Gar10] for the homogenous backgrounds and an inpainting algorithm based on Coherency Sensitive Hashing [KA11] for the textured backgrounds. In the last phase we finally apply the gradual linear transformation (moving and scaling) in order to create the final video. Moreover despite random video can be produced, there are several constraint to avoid unfeasible transformation. For example, an object which touch with the right border of the image, should not perform a movement along the opposite direction. This is because it is not possible rebuild the right missing part one out of the image. In Fig. 1 a flowchart of the proposed procedure is shown.

2.1. Segmentation Phase

The first step to apply the Kern Burns effect is to identify the various objects in the image, in order to process them separately. This purpose can be achieved through a segmentation algorithm, hence we decide to employ the Statistical Region Merging (SRM) described in [NN04]. Although many state-of-art segmentation algorithms exist [MRY*11, RKB04, PL08], we decide to employ SRM because is simple and fast. Hence, it is suitable to test the proposed pipeline. In future works, we are considering to test other kinds of segmentation methods. SRM algorithm takes as input an image and a integer number Q and returns as output a new image where the pixels have only Q possible values. Although the algorithm recognize Q different regions, which could be also not connected to handle the occlusions, we prefer consider two not connected segments as different regions. To obtain a good raw segmentation we suggest to use a high value for Q. However, the segments are merged in different meaningful regions as next step of the segmentation phase. In Fig. 2, it can be seen an example of application of the segmentation algorithm SRM with 8 and 1000 segments.

Given the high number of segments to obtain a good edge definition for the object, we provide a user interface to perform the merging of raw segments (Fig. 3). In this way a single object can be defined as a mix of multiple raw segments.

The merge of the segments can be done through two smart tools: the “line” tool and “box” tool. They allow to mark the segments which compose an objects with the same color, to indicate that the merging has been performed. The user is able to change the color to mark the segments of another object. The “line” tool, allows to draw a line to mark with the chosen color, all segments under the line. In Fig. 4 it can be seen an example of the “line” tool.

Through the “box” tool, a rectangle can be draw to mark with the same color all segments inside of it. The tool is shown in Fig. 5.

Each group of segments marked with the same color is placed in a different layer (one per color). The segments that are not marked with any color, will be automatically marked in black when the next phase is reached.

2.2. Layers Sorting Phase

At this point, each object marked in the previous phase is placed in a separated image or layer. In each layer, the unmarked pixels are set as “empty”. In this phase, the user have to choose the order of the produced layers, from the background to the foreground. This setting is a sort of discrete integer z coordinate for the each layer. For a segmentation with N layers, the value 1 is assigned to the background layer and the value N to the foreground one. All the other middle layers have an integer value i in $]1, N[$.

2.3. Inpainting Phase

After the layer sorting is performed, in the background layers some empty area appear. Since in the next phase, the objects in the higher layers could be moved, these empty area could become visible. To avoid this unpleasant effect, it is necessary to estimate the missing data by using some

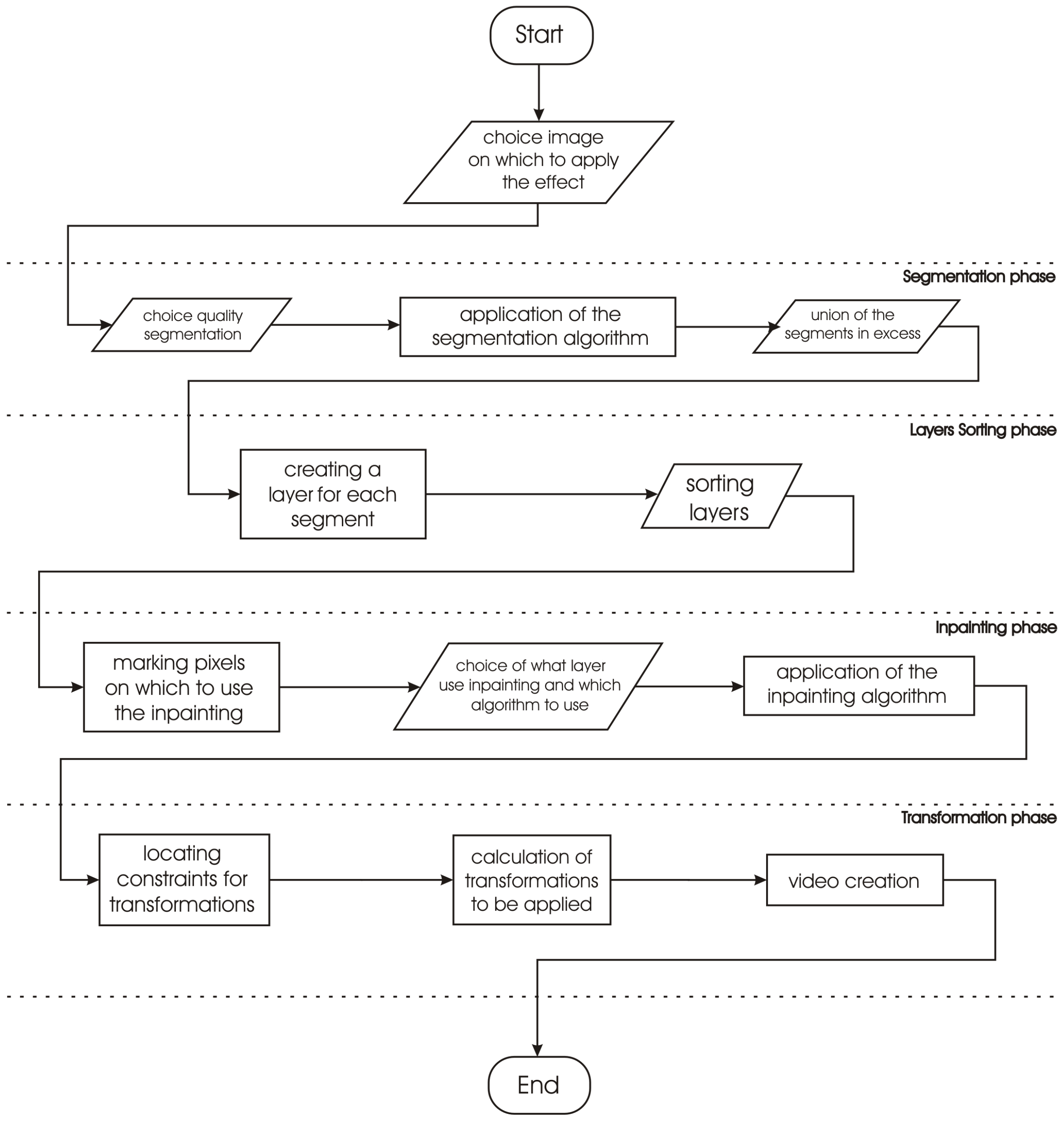
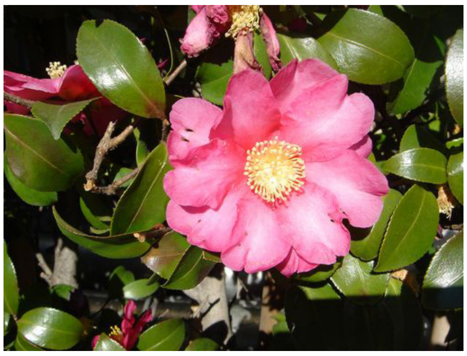

Start
choice image on which to apply the effect
Segmentation phase
choice quality segmentation
application of the segmentation algorithm
union of the segments in excess
Layers Sorting phase
creating a layer for each segment
sorting layers
Inpainting phase
marking pixels on which to use the inpainting
choice of what layer use inpainting and which algorithm to use
application of the inpainting algorithm
Transformation phase
locating constraints for transformations
calculation of transformations to be applied
video creation
End

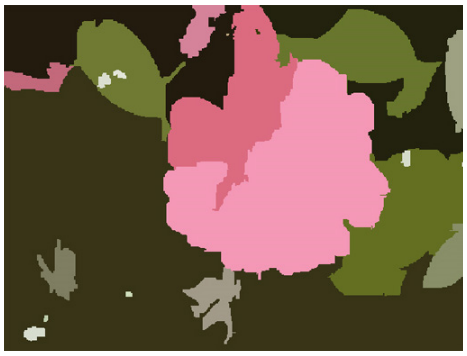
Figure 1: *The pipeline of the proposed algorithm.*

inpainting algorithms. In Fig. 6, we report an image where the inpainting should be used.

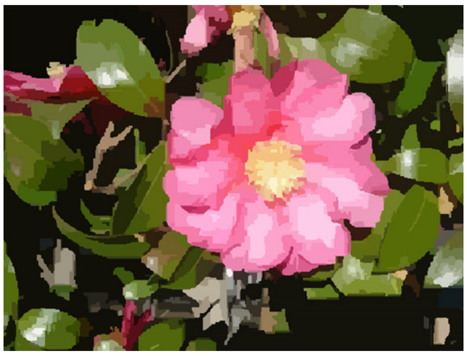
However, particular care is needed to a proper inpainting use. When two layers only has been produced, it is simple to apply the inpainting: the empty areas in the background layer, that are not empty in the foreground layer, have to be filled. In general, the rule to apply the inpainting with N different layers is the following: the empty areas in the layer $i < N$, that are not empty into at least a layer j, with $j > i$, have to be filled. The inpainting is never used in the foreground layer or $N-th$ layer. In Fig. 8 an example with three layers is shown. According to the aforementioned rule, all



(a)

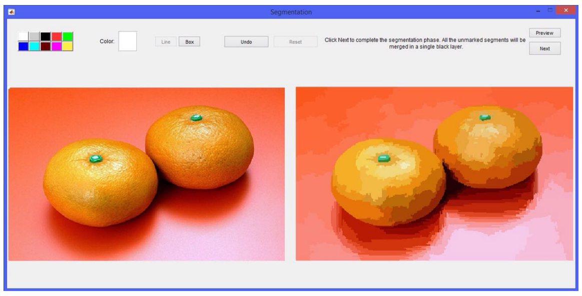


(b)



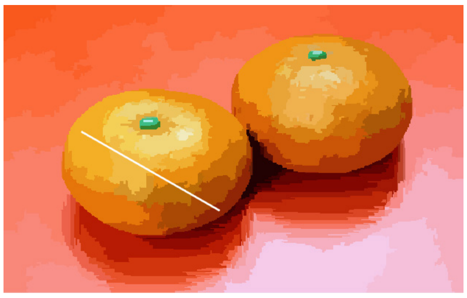
(c)

Figure 2: *(a) Original image; (b) Segmentation of the image (a) with* 8 *segments; (c) Segmentation of the image (a) with* 1000 *segments.*

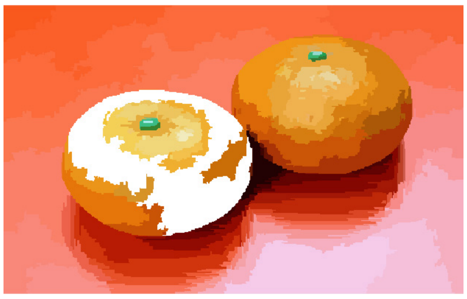


Click Next to complete the segmentation phase. All the unmarked segments will be merged in a single black layer.

Figure 3: *The interface to merge raw segments.*



(a)



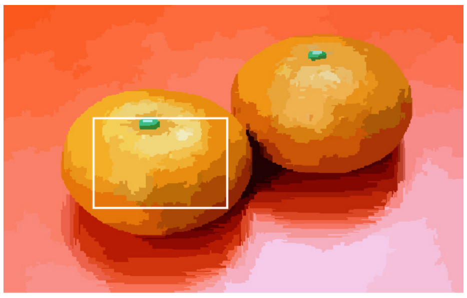
(b)

Figure 4: *(a) A line draw by "line" tool on the segmented image; (b) The marked segments under the line (b).*

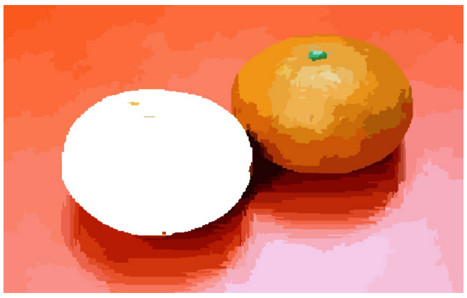
the black (empty) areas in Fig. 8(a) (layer 1) should be filled by using inpainting. In Fig. 8(b) (layer 2), only the empty area surrounded by the leaf is rebuilt. Finally, in the layer 3 (Fig. 8(c)) no operations is performed.

To guarantee a good reconstruction, two different kinds of inpainting algorithms are available for the user. The first algorithm is a data estimator in N-dimensional space based on the DCT [Gar10]. In our case, we use the algorithm for the estimation of pixel in RGB images, that is a 3-dimensional space. This algorithm is suitable for areas with homogenous content. The second one is based on Coherency Sensitive Hashing (CSH) [KA11], and is optimized for areas of picture where there is a more complex texture (e.g., grass).

In Figs. 9(a) and 9(b) it can be seen the results obtained by using the DCT inpainting on the Figs. 6(b) and 7(b), respectively. As you can note, the DCT algorithm produces a smooth version of the grass field in Fig. 9(b), which is perceptually unpleasant. Actually, DCT inpainting is more suitable for low detailed area, such as the orange background in Fig. 9(a). In Figs. 10(a) and 10(b) are shown the results obtained by using the CSH inpainting on the Figs. 6(b) and

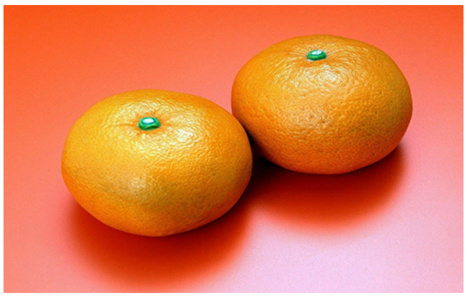


(a)

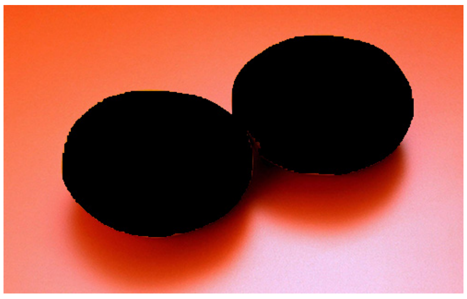


(b)

Figure 5: *(a) A box draw by "box" tool on the segmented image; (b) The marked segments into the box.*



(a)



(b)

Figure 6: *(a) The original image; (b) The black area should be fill by using inpainting.*

7(b) respectively. On the contrary, the best result by using CSH inpainting is obtained on the image in Figs. 10(b).

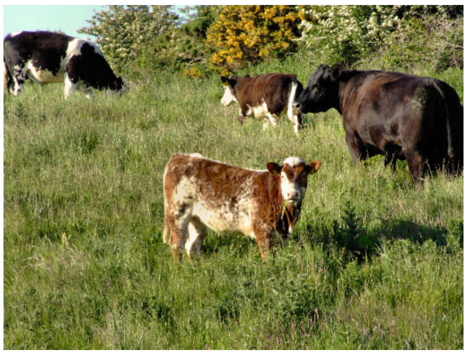
2.4. Transformation Phase

As last phase we perform the random panning and zooming transformation. However, the user could manually assign the shifting direction for some layers. To perform a reasonable random transformation, is necessary to define some constrains. For example, when an object in a certain layer touches with the right border of the image, it cannot move to the left direction. This is because the right section of the object totally misses. Hence, we define the following constrain: if a pixel of an object in a layer touch an image border, then the zooming out (shrinkage) and the movement in the opposite direction are forbidden. In Fig. 11 an example is reported.

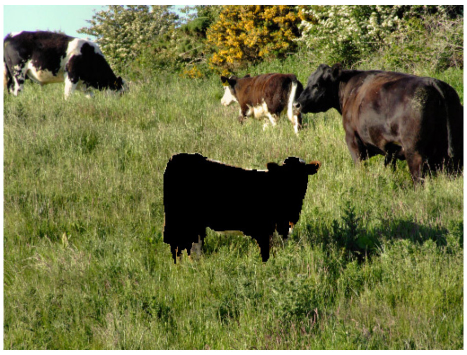
At this point, shifting and zooming are performed on each layers in accord with the aforementioned rules (Figs. 12 and 13).

The parameter, which tunes the transformation, is randomly computed within certain ranges. To obtain a parallax-like effect, the transformation of background layer are lighter than the ones on the foreground. For the layers $i > 1$, a horizontal shifting value and a vertical ones is chosen in $[-0.03w, \ +0.03w]$ and $[-0.03h, \ +0.03h]$ respectively, where w is the width of the image and h the height (in pixels).

This means, that a maximum shifting of 3% of the entire image is possible (in both directions). The zooming operation parameter is chosen in $[0.85, \ 1.15]$ for the layers with $i > 1$, while is chosen in $[1.00, \ 1.05]$ for the background layer (first layer). After the parameters are selected, each frame of the final video is created by applying a proper linear transformation at each image layers, which is performed through matrix product and inverse mapping. For each frame, we first process the top layer (the one in the foreground or layer 1) and at last the background layer. It is clear that during the transformation, only the no empty pixels are taken into account. The layers order, and the consequent overlays, has been respected by simply making the following condition: if in the final frame, a pixel has already been written, it cannot be overwritten. For this reason we choose to start by the foreground layer. To guarantee a gradual transformation for the full video, we use a proper function to weight the transformation matrix. The weight function, have been chosen to achieve no change in the first frame and the maximum change in the last frame. Let F the number of total frame to produce. If the selected zooming parameter at the previous step is Z, then the scaling coefficient in the matrix is 1 for the frame 1 and it is Z for the frame F. For the rest of the frame the coefficients change linearly from 1 to Z. The same approach is considered for the shifting transformation. If the selected shifting parameter is S, then the shifting coefficient linearly change from 0 to S in accord with the frame num-



(a)



(b)

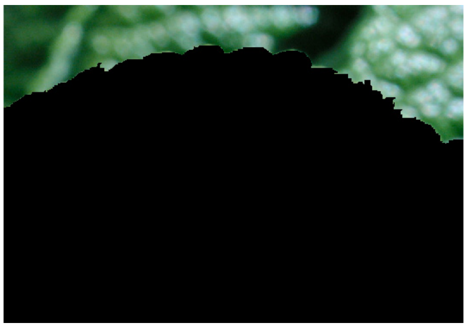
Figure 7: *(a) The original image; (b) Levels separation of the image (a).*

ber in $[1, F]$. In the final video, to reduce the artifacts due to the rounding operation of the pixel coordinates in the inverse mapping, we suggest to recompute each frame as the average of k frames. In this work, we found out that k=4 provides high quality results.

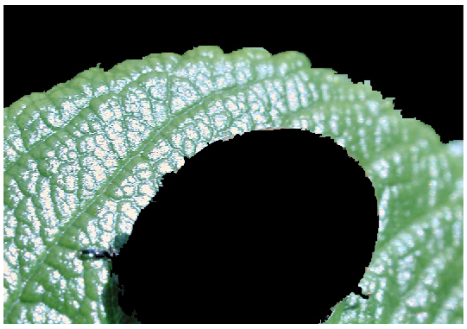
3. Results

To implement the proposed algorithm and the user-interface, MATLAB language has been exploited. We have made a friendly-interface to create a video by applying Ken Burns effect to a maximum of five images. The software, is able to guide the user from the load of the image to the video creation. To present the result, we create a sort of historical video documentary by using some old photographs. This video can be found in the website of Archeomatica Project[†] [BCVS06, SAM13, SAM14, STG*12]. However, we present a study case about one of the image, which appears in the produced video. In the Fig. 14(a), we show the gray scale image which has been processed. We decide to select three main layers: the child, the explosion and the background

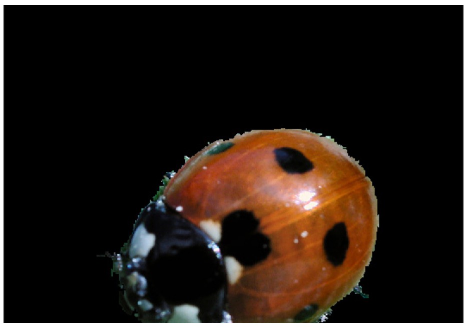
[†] http://www.archeomatica.unict.it/



(a)



(b)



(c)

Figure 8: *An example with three different layers. (a) The background layer* 3*; (b) The middle layer* 2*; (c) The foreground layer (a).*

(sky and land). The result of the segmentation phase are shown in Fig. 14(b). In Figs. 14(c) and 14(d) the two foreground layers has been separated from the background layer (Fig. 14(e)). According to the rule defined in section 2.3 for the inpainting, the two empty areas in the background layer has been reconstruct through the DCT inpainting algorithm 14(f).

4. Conclusions

In this paper, we have presented a procedure to automate as much as possible all the phases required to obtain the "Ken

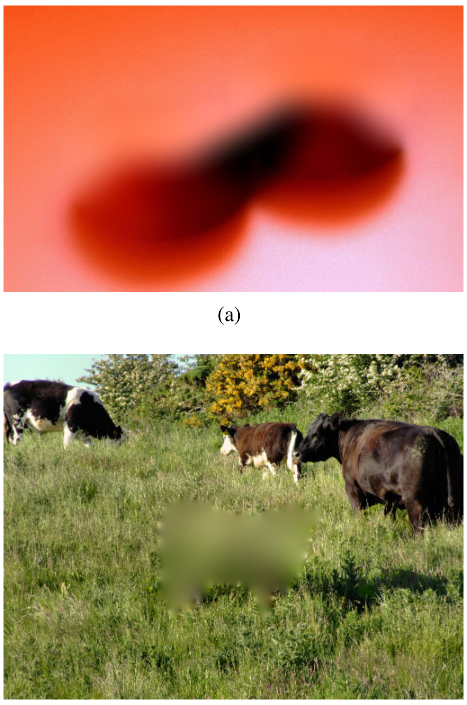


Figure 9: *(a) The DCT inpainting applied to the missing region (in black) of the image 6(b); (b) The DCT inpainting applied to the missing region (in black) of the image 7(b).*

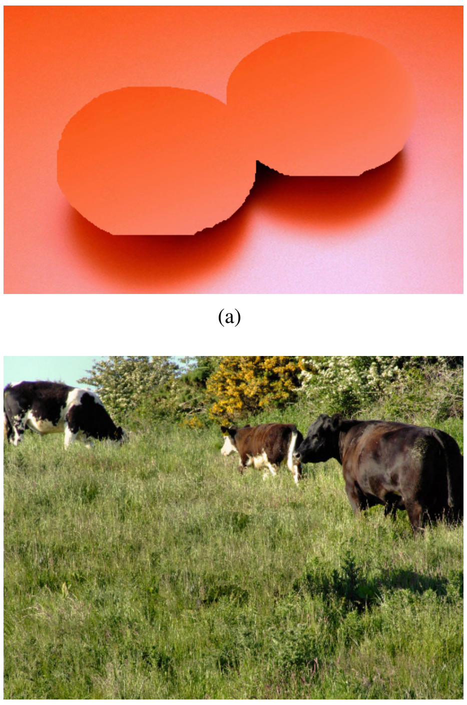


Figure 10: *(a) The CSH inpainting applied to the missing region (in black) of the image 6(b); (b) The CSH inpainting applied to the missing region (in black) of the image 7(b).*

Burns Effect". However, a little user interaction is required, because not all the steps to produce the effect are easily to automate. Therefore, the user interaction is essential to make the program more flexible and to remove some constrains which would make the program too limited. The manual merging, after the raw segmentation, is the interaction that takes more time for the user. Actually, to obtain a segmentation which captures the exact boundary of the objects into the image, it is necessary to create a high number of regions through the segmentation algorithm. Then, all these regions have to merge by using the provided tools. Nevertheless, our method achieves a substantial speed-up compared with a manual segmentation where the user draw the boundary for each object. Moreover, the user should choose the favourite object to segment, while the state-of-art segmentation algorithms detected all the meaningful regions in accord with a specific mathematical criterion. In future works, we are considering to employ a more precise segmentation algorithm or implement a suitable one for the discussed problem. Inpainting phase also plays a keyrole in the proposed method. This is because the information to reconstruct a missing part of the image, is not always enough. Moreover, the presence of some edges or a particular kind of texture can worsen the performance of the inpainting algorithm. For this reason, we allow the user to select two kind of inpainting algorithms. Each of them is suitable for different kind of images. As future improvement, we think to refine the inpainting phase by exploiting some strategy based on sematic content.

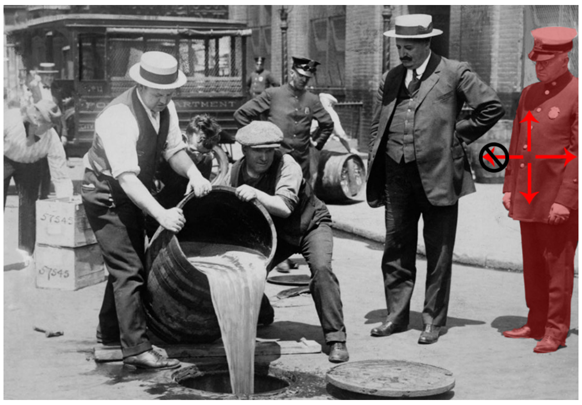


Figure 11: *The object in red cannot be moved to the left direction.*

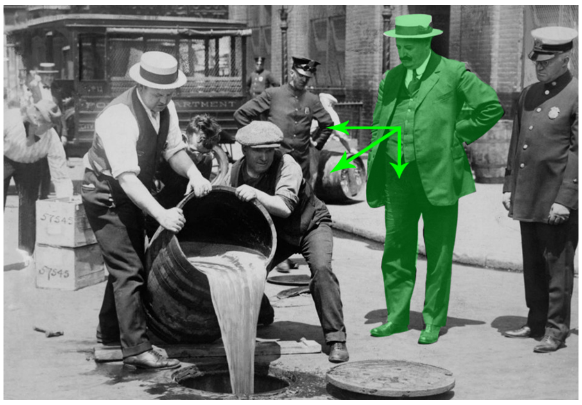


Figure 12: *A possible shifting transformation for the green object.*

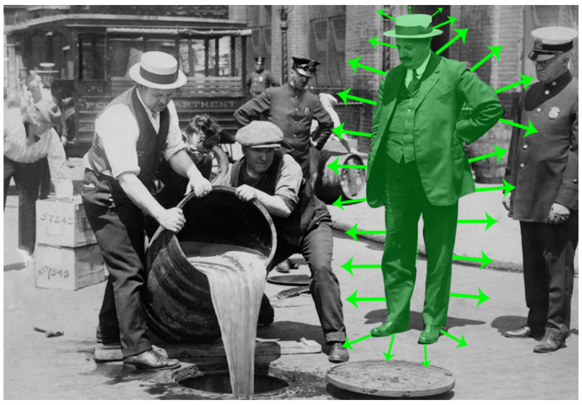


Figure 13: *A possible zooming transformation for the green object.*

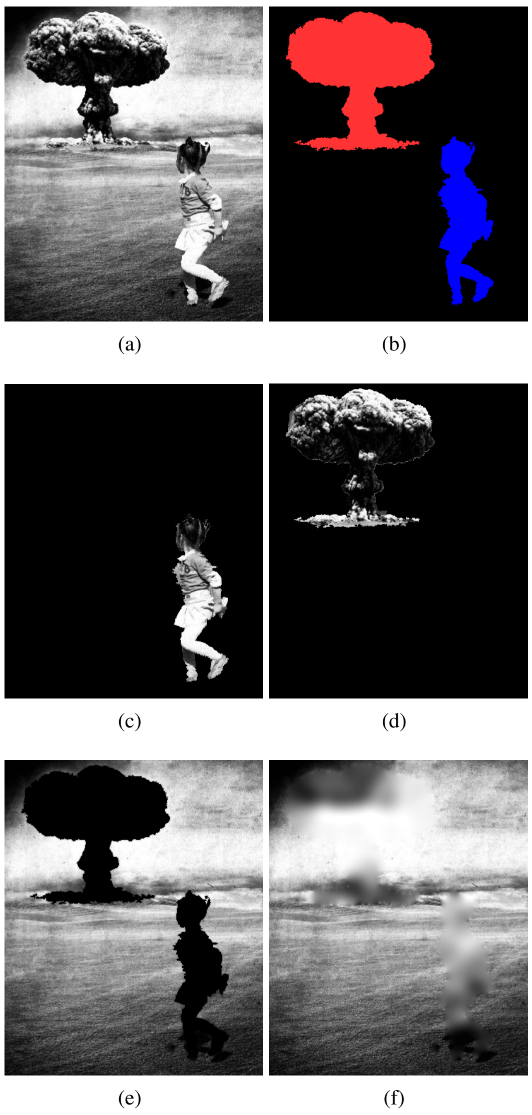


Figure 14: *(a) A case study image; (b) The output of the segmentation phase: in blue the "explosion" layer, in red the "child" layer and in black the background one; (c) The pixels of the red layer. In black the empty areas; (d) The pixels of the blue layer. In black the empty areas; (e) The background layer; (f) The new background layer, after that the empty areas are reconstructed by inpainting.*

References

[BCVS06] Bruni V., Crawford A., Vitulano D., Stanco F.: Visibility based detection and removal of semi-transparent blotches on archived documents. vol. 1, pp. 64–71. 6

[Gar10] Garcia D.: Robust smoothing of gridded data in one and higher dimensions with missing values. *Computational Statistics and Data Analysis 54*, 4 (2010), 1167 – 1178. 2, 4

[GD10] Green T., Dias T.: *The parallax effect:traveling through space*. New York: Springer, 2010. 1

[KA11] Korman S., Avidan S.: Coherency sensitive hashing. In *Proceedings of the International Conference on Computer Vision* (2011), pp. 1607–1614. 2, 4

[ken10] *Documentary Film Techniques: Ken Burns Effect, Interview, Documentary Swarm*. General Books LLC, 2010. 1

[KF03] Koble H., Fahs C.: *Final Cut Pro 4 Dummies. For Dummies*. 2003. 1

[MRY*11] Mobahi H., Rao S. R., Yang A. Y., Sastry S. S., Ma Y.: Segmentation of natural images by texture and boundary compression. *International Journal of Computer Vision 95*, 1 (2011), 86–98. 2

[NN04] Nock R., Nielsen F.: Statistical region merging. *IEEE Transaction IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 11 (2004), 1452–1458. 2

[PL08] Pablo A., Laurent D. C.: Constrained image segmentation from hierarchical boundaries. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2008). 2

[RKB04] Rother C., Kolmogorov V., Blake A.: "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Transanction on Graphics 23*, 3 (2004), 309–314. 2

[SAM13] Stanco F., Allegra D., Milotta F. L. M.: Detection and correction of mistracking in digitalized analog video. vol. 8158 LNCS, pp. 218–227. 6

[SAM14] Stanco F., Allegra D., Milotta F. L. M.: Tracking error in digitized analog video: automatic detection and correction. *Multimedia Tools and Applications* (2014). 6

[SRD11] Świrski L., Richardt C., Dodgson N. A.: Layered photo pop-up. In *ACM SIGGRAPH 2011 Posters* (2011), ACM, pp. 36:1–36:1. 2

[STG*12] Stanco F., Tanasi D., Gallo G., Buffa M., Basile B.: Augmented perception of the past - the case of hellenistic syracuse. *Journal of Multimedia 7*, 2 (2012), 211–216. 6

[ZCA*09] Zheng K. C., Colburn A., Agarwala A., Agrawala M., Salesin D., Curless B., Cohen M. F.: Parallax photography: Creating 3d cinematic effects from stills. In *Proceedings of Graphics Interface 2009* (2009), pp. 111–118. 1