DOI: 10.2352/ISSN.2470-1173.2016.15.IPAS-186

Spatio-temporal video background inpainting

Petr Pohl, Alexander Molchanov, Artem Shamsuarov, Victor Bucha
Samsung R&D Institute; Moscow, Russia

Abstract

In this article we present a method for inpainting background parts of scene in videos. Such inpainted background view can be used for rendering stereoscopic views of high quality. Completion of background occluded by foreground object is necessary to solve the parallax occlusion. Known inpainting algorithms, e.g. Criminisi, 2004 [1] or Telea, 2004 [2] were primarily designed for inpainting of still images. However, frame-by-frame spatial inpainting of video inevitably leads to flickering. Our hybrid inpaint algorithm utilizes the fact that some parts of background can be seen in temporally close frames, and some can only be filled-in with spatial inpainting approaches. The algorithm considers foreground masks and video frames as input, performs motion analysis to determine areas where spatial and temporal inpainting should be used and then uses both approaches to find a plausible reconstruction of background. The resulting output is a video sequence with marked foreground object(s) removed.

Introduction

Video inpainting problem

There are several possibilities to formulate video inpainting task. The most common way is as follows. We are given an input video sequence and a corresponding sequence of masks which denote areas to be inpainted. Such sequence of masks is sometimes called a space-time hole. The goal is to recover areas denoted by masks so that the result visually plausible and temporally coherent.

A lot of attention was given to the area of still image inpainting. Notable methods include diffusion-based approaches, such as [2, 3] and texture synthesis or exemplar-based algorithms, for example [1]. Video inpainting imposes additional restrictions which make it more computationally expensive and challenging.

Related work

Generally most of video inpainting approaches fall into two big groups: global and local with further filtering. Exemplar-based methods for still images have a natural extension for videos as a global optimization problem of filling the space-time hole. In [4] Wexler et al. define global method using patch similarity for video completion. The article reports satisfactory results, but the price for global optimization is extremely high complexity of algorithm. They report several hours for a video of very low resolution and short duration (100 frames of 340×120 pixels).

A similar approach was taken by the Shiratori et al. [5]. They suggest a procedure called motion field transfer to estimate motion inside space-time hole. Motion is filled-in with a patch-based approach using a special similarity measure. Found motion allows the authors to inpaint view sequence while maintaining temporal coherence. However, the performance is also quite low (40 minutes for 60-frame video of 352×240 pixels).

Another way of video inpainting was suggested by Burgeau et al. in [6]. They propose to inpaint frames independently and filter the results by Kalman filtering along point trajectories found by dense optical flow algorithm. The slowest operation of this approach is computation of optical flow. Their method produces visually consistent result, but the inpainted region is usually smoothed with occasional temporal artifacts.

Proposed algorithm

Our goal was to deliver solution that can effectively work on an ordinary PC with a single relatively powerful GPGPU. Because of that we decided not to use global optimization approach. The main problem for successful video background restoration is to achieve temporal consistency without sacrificing level of details. We tackle this problem by reconstructing background motion occluded by marked foreground objects. Our algorithm consists of three well defined steps.

Algorithm Outline

1. Motion estimation including restoration of background motion in foreground regions.
2. Temporal propagation of background image data using restored background motion from step 1.
3. Iterative spatial inpainting step with temporal propagation of inpainted image data.

For the first step we used optical flow algorithm [7] and a background motion inpainting procedure which will be described further.

The second step — temporal propagation — is the crucial part of our algorithm. Here we do a forward and backward pass through video sequence using integrated motion in occluded regions. In forward pass we integrate motion in backward direction and decide, which pixels can be filled by data in the past. The same is done for backward pass.

After forward and backward temporal pass, we still can have some areas that were not inpainted (unfilled areas). These areas were not seen during the video. It is necessary to use still image spatial inpainting to fill the missing data and propagate it using restored background motion to achieve temporal consistency.

Implementation

Motion estimation

Traditional optical flow algorithms recover motion of visible pixels. Some of them (e.g. [7]) try to recover correct motion in occlusion area. In our case it is necessary to get an estimate of motion inside frame area that can be occluded for several frames. Such task is only feasible for relatively rigidly behaving background. In this section we will describe background motion restoration based on dense optical flow.

DOI: 10.2352/ISSN.2470-1173.2016.15.IPAS-186

Background motion restoration

We start by computing dense optical flow for all consecutive pairs of input video frames. For that purpose we use dense optical flow algorithm with occlusion fixing described in [7]. After that we need to estimate background motion inside marked foreground regions. Let us refer to figure 1 for further details.

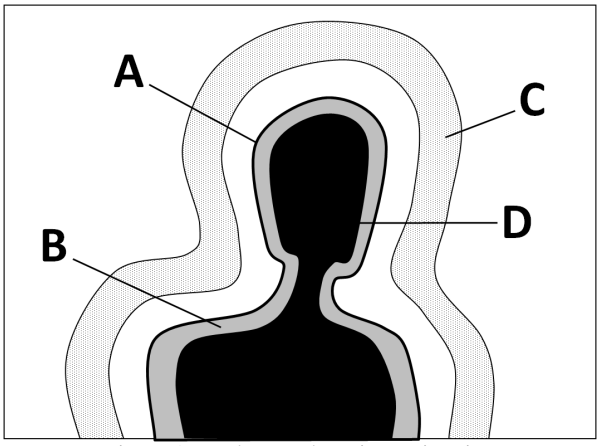


Figure 1: Background motion estimation

Our main idea is to compute and combine several motion estimates based on optical flow results. Let black contour **A** be the edge of a marked foreground region. Outside **A** we will be using motion produced by optical flow algorithm, $M_0(x,y)$. In area **B** we obtain a *local* background motion estimate $M_1(x,y)$. Said local estimate can be produced for example by running diffusion-like inpainting algorithm (such as [2]) on $M_0(x,y)$ inside region **B**.

Next, we use $M_0(x,y)$ from area **C** to fit parameters $a_0,\dots,a_5$ of an affine global motion model

$$M_2(x,y) = \begin{pmatrix} a_0 + a_1 x + a_2 y \\ a_3 + a_4 x + a_5 y \end{pmatrix}. \quad (1)$$

Motion $M_2(x,y)$ is used inside area **D**. We choose area **C** to be separated from foreground object contour **A**. This is to avoid introducing artifacts which might come from optical flow estimation around object edges into global motion estimates.

To improve motion smoothness we blend motions M_1 and M_2 inside **A**. Let $0 \le W(x,y) \le 1$ be a weighting function, then the resulting motion vector field is computed as

$$M(x,y) = (1 - W(x,y)) \cdot M_1(x,y) + W(x,y) \cdot M_2(x,y).$$

$W(x,y)$ is defined as exponential decay (with reasonable decay rate parameter in pixels) of distance from foreground area edge **A**. It equals to 1 outside contour **A** and the distance parameter is computed by effective distance transform algorithm.

As a result, we obtain full-frame per-pixel estimate of background motion that generally has properties of global motion inside previously missing regions, but does not suffer from discontinuity problems around foreground region edges. Figure 2 shows example of background motion restoration.

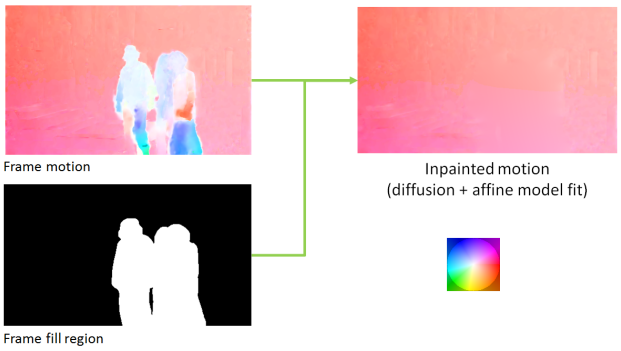


Figure 2: Background motion estimation

Temporal passes

The forward and backward temporal passes are symmetrical and they fill in the areas which were visible on other frames of video. After motion estimation we have background motion vectors for all consecutive pairs of frames in both directions. Let us introduce the following notation:

- $I(n,x)$ is the n-th input frame
- $M(m,n,x)$ is restored background motion from frame m to frame n
- $F(n,x)$ is input foreground mask for frame n (area to be inpainted)
- $Q_F(n,x)$ is inpainted area mask for frame n
- $I_F(n,x)$ is inpainted frame n
- T is temporal window size (algorithm parameter)

We will also use the following symbols for binary masks operations:

- $\&$ — intersection of binary masks (pixelwise AND)
- $\vee$ — union of binary masks (pixelwise OR)
- $\sim$ — negation (inverse) of binary mask (pixelwise NOT)

Algorithm 1 forward temporal pass

for $N_{curr} = 1 \dots N_{frames}$ **do**
  Intitialize forward pass image and mask:
  $I_F(N_{curr},x) = I(N_{curr},x)$
  $Q_F(N_{curr},x) = 0$
  Iterate backwards for temporal data:
  for $N_{src} = (N_{curr}-1) \dots \max\{1, N_{curr}-T\}$ **do**
    Integrate motion:
    $M(N_{curr},N_{src},x) = M(N_{curr},N_{src}+1,x) + \dots$
    $\dots + M(N_{src}+1,N_{src},x + M(N_{curr},N_{src}+1,x))$
    New inpainted mask:
    $Q_F^{new}(N_{curr},x) = F(N_{curr},x) \& \sim Q_F(N_{curr},x) \& \dots$
    $\dots \& F(N_{src},x + M(N_{curr},N_{src},x)$
    Update inpainted image and mask:
    for all points x marked on $Q_F^{new}(N_{curr},x)$ **do**
      $I_F(N_{curr},x) = I_F(N_{src},x + M(N_{curr},N_{src},x))$
      $Q_F(N_{curr},x) = Q_F(N_{curr},x) \vee Q_F^{new}(N_{curr},x)$

This is a kind of greedy algorithm, which inpaints background with data from temporally least distant frame. Backward temporal pass algorithm is doing the same operations only with reversed order of iterations and motion directions.

DOI: 10.2352/ISSN.2470-1173.2016.15.IPAS-186

Some areas can be filled from both sides. In this case we need to find single inpainting solution. We used temporally less distant source. In case of the same temporal distance a blending procedure was done based on distance from non-inpainted area.

Spatial pass

The goal of spatial pass is to inpaint regions, which were not filled by temporal passes. To achieve temporal stability, we inpaint on a selected frame and propagate inpainted data temporally. We found, that reasonable strategy is to find the largest continuous unfilled area, use spatial inpainting to fill it in and then propagate through whole sequence using background motion estimate. It is necessary to perform spatial inpainting with propagation iteratively until all unfilled areas are inpainted.

Any spatial inpainting algorithm can be used, in our work we experimented with exemplar based and diffusion based methods. Our experience shows, that its better to use diffusion algorithm for filling small or thin areas and exemplar based one for larger unfilled parts.

Let us denote $Q_{FB}(n,x)$ and $I_{FB}(n,x)$ as temporally inpainted mask and background image respectively after forward and backward passes are blended together. $Q_S(n,x)$ and $I_S(n,x)$ are mask and image after temporal and spatial inpainting. $|D|$ stands for number of pixels in image domain D.

Algorithm 2 spatial pass

Intitialize spatially inpainted image and mask:
$Q_S(n,x) = Q_{FB}(n,x)$, $I_S(n,x) = I_{FB}(n,x)$
while $|F(n,x)\& \sim Q_S(n,x)| > 0 \quad \forall (n,x)$ **do**
  Find N_{curr} s.t. $|F(N_{curr},x)\& \sim Q_S(N_{curr},x)|$ is maximal
  Let $R(N_{curr},x) = F(N_{curr},x)\& \sim Q_S(N_{curr},x)$
  Inpaint area $R(N_{curr},x)$ and store pixel data into $I_S(N_{curr},x)$
  $Q_S(N_{curr},x) = Q_S(N_{curr},x) \vee (F(N_{curr},x)\& \sim Q_S(N_{curr},x))$
  Propagate inpainted area forward and backward:
  for all frames $N_{dst} : |F(N_{dst},x)\& \sim Q_S(N_{dst},x)| > 0$ **do**
    Integrate motion to obtain $M(N_{dst},N_{curr},x)$
    Get new inpainting mask:
    $Q_S^{new}(N_{dst},x) = F(N_{dst},x)\& \sim Q_S(N_{dst},x)\& \ldots$
    $\ldots \& Q_S(N_{curr}, x + M(N_{dst},N_{curr},x)$
    Update inpainted image:
    for all points x marked on $Q_S^{new}(N_{dst},x)$ **do**
      $I_S(N_{dst},x) = I_S(N_{curr}, x + M(N_{dst},N_{curr},x))$
      $Q_S(N_{dst},x) = Q_S(N_{dst},x) \vee Q_S^{new}(N_{dst},x)$

Experimental Results

Synthetic Dataset

For quantitative evaluation we generated a set of synthetic examples with known ground truth background and motion. There are two kinds of object motion in test sequences:

- Simple motion. Background and foreground are moved by two different randomly generated motions which include only rotation and shift.
- Affine motion. Background and foreground are moved by two different randomly generated affine motions.

Evaluation was done by running the proposed algorithm with default parameters on test sequences. We decided to use publicly available implementation of exemplar-based inpainting algorithm by Criminisi et al. from [1] to provide a baseline for our approach. Our goal was to test both the quality of background inpainting for each frame and temporal stability of the resulting sequence. For evaluation of background inpainting quality we use PSNR between algorithm output and ground truth background. Results are shown in figure 4.

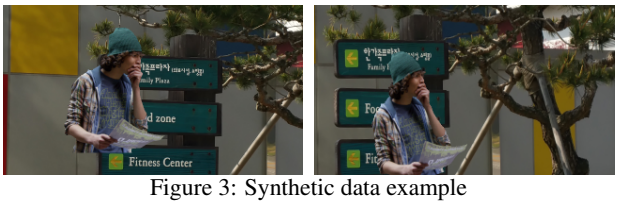
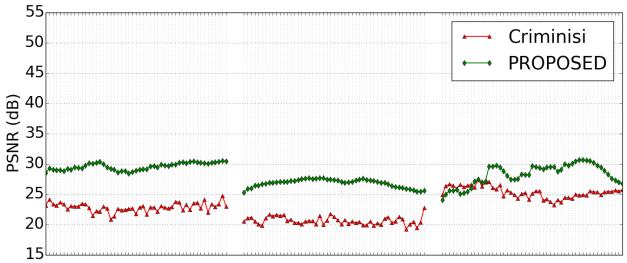
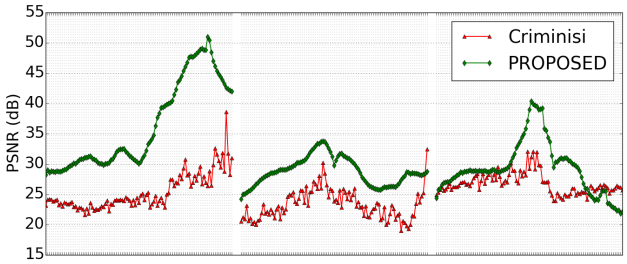


Figure 3: Synthetic data example



(a) Simple motion



(b) Affine motion

Figure 4: Background restoration quality

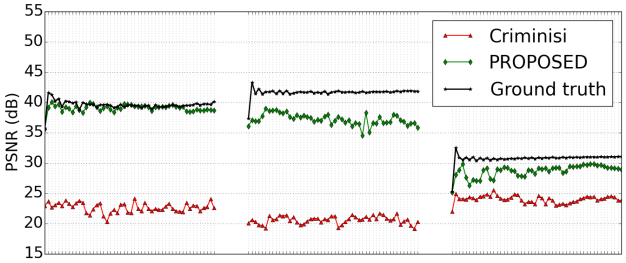
To measure temporal stability we use the following procedure. Let $I(n,x)$ and $I(n+1,x)$ be a pair of consecutive frames with inpainted background and $M^{GT}(n,n+1,x)$ be ground truth motion from frame n to $n+1$. We compute PSNR between $I(n,x)$ and $I(n+1,x+M^{GT}(n,n+1,x))$ (sampling is done using bicubic interpolation). Results are shown in figure 5.

As we can see, our inpainting algorithm provide usually slightly better quality of background restoration than exemplar-based inpainting when analyzed statically. However, it produces far more temporally stable output which is very important for video inpainting.

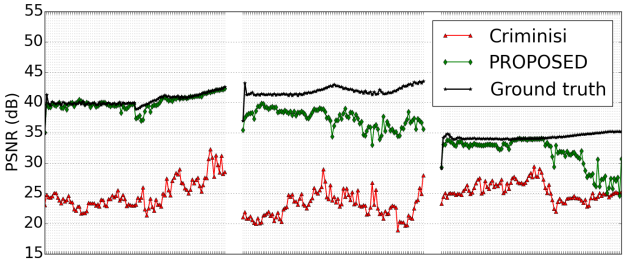
Real Data

We also tested our algorithm on a proprietary database of videos with two types of resolution: 1920×1080 (FHD) and 960×540 (quarter HD, qHD). The method shows reasonable quality for scenes without changes of scene properties (brightness change, different focus, changing fog or lights) and with rigid scene structure. Few examples of our algorithm outputs are shown in Figure 6.

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.15.IPAS-186



(a) Simple motion



(b) Affine motion

Figure 5: Temporal stability

Figure 6: Algorithm output

Typical visible artifacts include misalignments on edges of temporal inpainting direction (presumably in cases when motion integration is not precise enough), and mixing same part of scene which changed appearance properties with time. Examples of such are shown in figure 7 and figure 8 respectively.

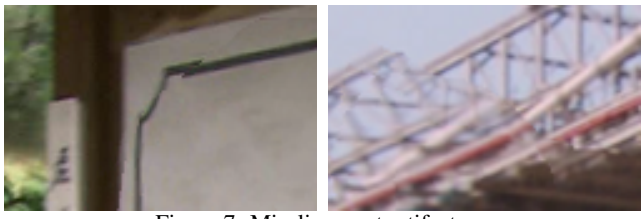
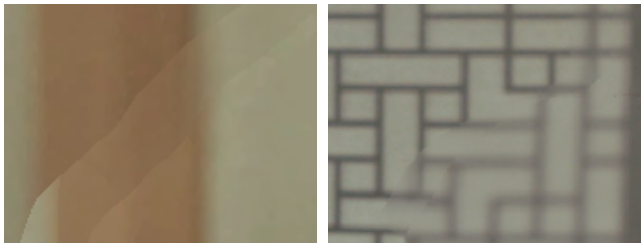


Figure 7: Misalignment artifacts



(a) (b)

Figure 8: Scene change artifacts

Running time of the algorithm (not including optical flow estimation) is around 1 s/frame for qHD and 3 s/frame for FHD sequences on a relatively modern PC with single GPU (Core i7, 8GB RAM, GTX 760 GPU in our case). Despite the mentioned drawbacks we believe results to be acceptable for restoration of areas, occluded due to stereoscopic parallax for a limited range of scenes.

For wider applicability it would be necessary to extend the algorithm to decrease the level of artifacts. We see two main improvements. One of them is more advanced analysis of propagation reliability for better decision between spatial and temporal inpainting and temporal inpainting direction using analysis of the level of scene changes. The other is improve alignment of temporally inpainted parts from different time moments, or apply Poisson seamless stitching approach similar to [8] in case there are overlapping parts.

References

[1] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama, Region filling and object removal by exemplar-based image inpainting. IEEE Transactions on Image Processing, 13, 9 (2004).

[2] Alexandru Telea, An image inpainting technique based on the fast marching method. Journal of graphics tools 9, 1 (2004).

[3] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, Coloma Ballester, Image inpainting. Proc. of the 27th annual conference on Computer graphics and interactive techniques, pg. 417. (2000).

[4] Yonatan Wexler, Eli Shechtman, and Michal Irani, Space-time completion of video. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29, 3 (2007).

[5] Takaaki Shiratori, Yasuyuki Matsushita, Xiaoou Tang, Sing Bing Kang, Video completion by motion field transfer. Proc. 2006 IEEE

DOI: 10.2352/ISSN.2470-1173.2016.15.IPAS-186

Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pg. 411. (2006).

[6] Aurlie Bugeau, Pau Gargallo I. Piracés, Olivier D'Hondt, Alexandre Hervieu, Nicolas Papadakis, Vicent Caselles, Coherent background video inpainting through Kalman smoothing along trajectories. Proc. VMV 2010-15th International Workshop on Vision, Modeling, and Visualization, pg. 123. (2010).

[7] Petr Pohl, Michael Sirotenko, Ekaterina Tolstaya, Victor Bucha, Edge preserving motion estimation with occlusions correction for assisted 2D to 3D conversion. IS&T/SPIE Electronic Imaging (2014).

[8] Patrick Pérez, Michel Gangnet, Andrew Blake, Poisson image editing. ACM Transactions on Graphics (TOG), 22, 3, (2003).

Author Biography

Petr Pohl received MSc. in Technical Cybernetics from Czech Technical University (CTU) in Prague (2002). From 2002 until 2011 he worked at Neovision sro., a CTU startup working on industrial applications of computer vision including precise measurement and laser welding navigation. From 2011 until now he works in Samsung R&D Institute Russia on projects related with video augmentation and frame rate conversion.

Alexander Molchanov received specialist in applied mathematics from the Moscow State University (2012). From 2011 till 2012 he worked as computer vision developer at Redmadrobot LLC, Moscow. Since 2012 until now he works in Samsung R&D Institute Russia. His work is focused on motion estimation and image enhancement.