How Not to Be Seen – Inpainting Dynamic Objects in Crowded Scenes

M. Granados, J. Tompkin, K. Kim,
O. Grau, J. Kautz, C. Theobalt

MPI–I–2011–4–001 February 2011

Authors' Addresses

M. Granados
MPI Informatik
Campus E14
D-66123 Saarbrücken

J. Tompkin
University College London
Malet Place
London WC1E 6BT, UK

K. Kim
MPI Informatik
Campus E14
D-66123 Saarbrücken

O. Grau
BBC Research & Development
Centre House, 56 Wood Lane
London W12 7SB, UK

J. Kautz
University College London
Malet Place
London WC1E 6BT, UK

C. Theobalt
MPI Informatik
Campus E14
D-66123 Saarbrücken

Acknowledgements

Significant thanks to Yonatan Wexler for his generous and patient help, and to Yael Pritch for answering our questions. This work was partially supported by BBC R&D, and by the EngD VEIV Centre at UCL.

Abstract

Removal of dynamic scene elements from video is an extremely challenging problem that even movie professionals often solve through days of manual frame-by-frame editing. The disoccluded regions in the video have to be inpainted in a coherent way, even if originally occluded objects or background are dynamic. To make this problem easier, we propose a new approach for video inpainting that can deal with complex scenes with dynamic backgrounds and many non-periodically moving occluding scene elements. It is built on the idea that a spatio-temporal hole created by a removed scene element can be filled by copying information from other space-time locations in the video, where objects and background are unoccluded. Inpainting is performed by solving a combinatorial optimization problem that searches for the optimal pattern of pixel shifts. Solving this problem naively, even on short videos, quickly becomes infeasible. The primary contributions of this work are a new energy functional with desirable convergence properties, an efficient hierarchical solution strategy, and an effective search space reduction strategy that restricts potential pixel shifts to regions around tracked objects in the scene. A simple interface enables the user to optionally support the algorithm in marking and tracking dynamic objects. Our approach can efficiently inpaint holes even in HD videos with many occlusions, and requires only little user input.

Keywords

video inpainting, video completion

Contents

1 Introduction

Removing unwanted people or objects from videos is a very common task in professional video and movie productions. For instance, when filming in public locations, it is often necessary to remove walking people or moving objects, such as cars, that accidentally occlude the scene. Very often it is impossible to get full control over a film set and a problem that arises is to remove unwanted people or objects from the scene. Moreover, the ability to alter the dynamic objects in a scene opens up new creative editing possibilities.

Removing such objects requires *inpainting* the resulting hole in the video in a perceptually plausible way and respecting the temporal coherence of the scene. Furthermore, it is not uncommon that there are multiple moving dynamic objects each occluding and being occluded. This is a tremendously difficult task and requires artists to spend many hours or days removing even small objects.

Consequently, a semi-automatic tool to assist users (artists) in this task would be highly desirable. However, in general video inpainting is an ill-posed problem since there is no unique solution for unobserved occluded regions. In order to make video inpainting tractable, previous work commonly makes strong assumptions about the scene structure. For instance, some algorithms assume that the occludee (the object or the person to be completed) is under cyclic motion [6, 13].

We propose an algorithm for video inpainting that is inspired by the shift-map concept of Pritch et al. [12] for 2D image inpainting, but works in the spatio-temporal domain. Our method can be applied to more general situations than those of previous approaches: We can inpaint or remove static as well as dynamic objects even in crowded scenes with many occluders and occludees, and no cyclical motions are assumed. Our algorithm only assumes that, locally, the video content to be inpainted is visible somewhere else in the video (in space and in time). This is a general assumption which we share with other methods [6, 19, 15].

Identifying the most plausible visible locations with which to inpaint a hole is

already a very challenging task in a 2D image. Methods like Shift-map [12] or Patch-match [1] typically approach this by solving a combinatorial optimization of a Markov random field-like energy. The solution assigns a shift to each pixel in an image to a different location. An energy functional penalizes visible differences between the inpainted hole and visible content, and imposes priors on acceptable shift patterns and spatial smoothness. Since the parameter spaces are extremely large, in order to give acceptable computation times these methods demand effective complexity reduction strategies such as hierarchical solvers, or energy terms that may sacrifice desirable properties such as provable convergence.

In our approach, we follow a similar strategy, but are now faced with a combinatorial energy minimization in the 3D spatio-temporal domain, which dramatically increases the problem complexity. To stand a chance to compute a plausible spatio-temporally coherent shift pattern in a video volume, we have to carefully design an energy that does not sacrifice regularity (cf. [3]). This enables us to use an efficient optimizer that delivers results in reasonable time.

The video inpainting method by Wexler et al. [19] approaches these challenges in a greedy fashion, which leads to adequate results in small sized holes, but leads to very long computation times and smoothing artifacts (cf. Fig. 6.4). We design a spatio-temporal energy functional that, in contrast to image-based shift-map editing and previous video inpainting methods, is specifically designed to consider spatial and temporal consistency in different ways, such that high-quality inpainting results are achieved under a stable set of parameters. We also present an effective shift space reduction strategy that restricts the search to tracked windows around dynamic objects, as well as a user interface in which automatically generated tracks can be easily modified.

Our main contributions are:

- A new energy functional with advantageous convergence properties (it is regular), which specifies the most suitable spatio-temporal shifts to fill in holes in a video.
- A hierarchical graph-cut-based optimization that yields plausible inpainting solutions even on large spatio-temporal holes in videos with many closely interacting occluders and occludees.
- Complexity reduction by tracking occludees across the video, and restricting the search space to the windows around them.
- A user interface enabling quick optional correction of automatically generated masks for objects to remove, and for building tracks for occludees which may be behind many occlusions.

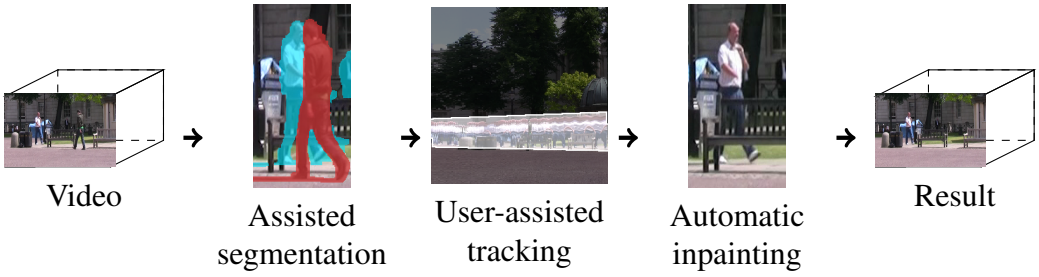


Figure 1.1: *Pipeline overview. Given a video, we first segment all dynamic objects based on an automatically computed background plate and camera noise estimations. Second, the user selects (and possibly refines) the segmentation of the object to be removed O_{rem} using a newly designed interface (Fig. 4.1). Third, the user roughly localizes each occluded object O_{compl} on two 2D spatio-temporal projections of the video (see Fig. 4.2). This drastically reduces the search space for the inpainting process. Lastly, we inpaint the background and each of the O_{compl} regions occluded by O_{rem} by solving a combinatorial optimization problem (cf. Sec. 5).*

Figure 1.2: *A semi-automatically tracked object (marked with red boundaries) is to be removed from a video sequence. To this end, the dynamic occluded object in the background needs to be inpainted. The right-hand side of each frame pair shows our result.*

In conjunction, these contributions enable one of the first approaches to remove dynamic scene elements from videos of complex crowded scenes with many interacting objects. Our video inpainting system succeeds to reconstruct non-periodically moving dynamic objects suffering non-trivial occlusions situations over complicated backgrounds. We also demonstrate that our approach compares favorably to related methods from the literature in terms of both runtime and achieved visual quality.

2 Related Work

Existing video inpainting algorithms can be broadly classified into two categories. The first category employs a *global* energy minimization framework where the necessary requirement of video inpainting plus a priori knowledge is encoded into a single energy functional, to which a globally consistent solution is found. Accordingly, these approaches can be applied even when the hole is large in space and in time. However, this minimization incurs a high computational complexity , which is commonly reduced through the use of approximate solutions. The second category of algorithms greedily propagate the available information from outside the hole into the hole, usually layer-by-layer. While this generally results in faster inpainting in comparison with energy minimization, *local* propagation of information does not guarantee any global consistency.

Wexler et al.'s [19] method is an example of the first category. A spatio-temporal patch is sampled from each observed pixel in a given input video. The collection of these patches constitute a database reflecting the statistics of a given video. Inpainting is performed by greedily assigning a patch from the database to each missing pixel such that the joint configuration of these assignments minimizes a predefined energy functional. The energy functional is configured based on the (dis-)agreement of overlapping patches, which casts inpainting as a global energy minimization. Results are usually smooth, as multiple patch candidates are combined at each pixel.

The algorithm that inspired our method is *shift-map image editing*, proposed by Pritch et al. [12]. This algorithm produces a vector field called a shift-map, which assigns to each pixel location in the input image an offset to a different pixel such that the output minimizes a given cost-functional. Pixels in the output image are encoded in the functional and shift-map values for missing pixels are constrained to be taken from pixels outside the hole. While this method has shown impressive performance in inpainting images and other related image editing applications, it cannot be directly applied to video (let alone complex videos with many mutually

occluding objects). The energy function has to be re-designed to treat spatial and temporal dimensions according to their different characteristics, as we do in our work and experimentally demonstrate. Further, their energy is not regular [3], which impairs graph-cut convergence, strongly reduces convergence speed, and may lead to artifacts. Finally, temporal redundancies should be identified and exploited to reduce computation time.

There have been several algorithms which try to retain the advantages of global approaches while reducing the computational complexity. For instance, [13] tracked each pixel in the occludee through the video. As a result, in the energy minimization stage, the search space for each pixel was reduced to a 2D manifold (rather than a 3D volume). The price for this simplification was that it can handle only pure translation of rigid objects or periodic motion.

Several techniques have been proposed that use object-based inpainting. The method by Venkatesh et al. [17] tracks and segments the dynamic object that requires inpainting. A database of segmented frames, where the object is fully visible, is created from this. Holes are now inpainted by matching segmented database frames against segmented frames at the hole boundary and against each other using dynamic programming; i.e., full (segmented) frames are used for inpainting. This required segmentation to be very accurate and motions to be mostly cyclical. This idea was extended by Ling et al. [8]. The contours of the object of interest are estimated by using motion information. The contours are then used to retrieve the object frames from a database using an approach similar to Venkatesh et al. [17]. In order to find database frames despite differences in posture, query postures are synthesized based on local segments of the object. The technique by Jia et al. [6] first segmented moving objects in the video. Assuming the periodicity of motion for the object to be inpainted, the inpainting problem was then cast as warping and aligning the visible trajectory of the object (which is referred to as a 'movel') with that of the damaged movels. Object-based systems demonstrate plausible inpainting for certain categories of moving objects, especially humans. However, for reliable estimation of correspondences between visible instances of the occludee at different times, they either impose a explicit class of possible motions (e.g., cyclic) [17, 6] or require the motion to be simple such that a dense sampling of postures is feasible [8]. Furthermore, by design, the inpainting of the object is performed independently of the background (especially when the object segmentation is very accurate). The consequence is that in the final result the person or object often looks unnatural and as if it had been pasted over the background (see project website of [8, 6]).

An example of a local approach is Patwardhan et al. [11]. They assign a priority to each pixel in the hole and, proceeding by highest priority, copy in a patch which

is visible and best matches the context of the pixel of interest. The priority is calculated based on the existence and direction of the motion across the boundary of the hole. While this algorithm enables fast inpainting, in general the results are not guaranteed to be globally coherent. This algorithm was later improved to handle simple camera motions [10].

Meanwhile, Shih et al. [14] proposed an inpainting algorithm in the context of video falsification where one changes the semantic contents of the video by manipulating the motions of people present in the video. Their inpainting component was constructed as a 3D extension of the exemplar-based image inpainting algorithm by Criminisi et al. [4]: The boundary pixels of the hole are assigned by searching for visible locations which best match the texture and motion contexts. To reduce the time complexity of the search process, each person is tracked in the video and the corresponding *skeleton* model is computed. By propagating a skeleton into the corresponding object hole and establishing the correspondences between these figures throughout different frames, the search space is reduced. However, they expect motions to be cyclic and expect that a 2D skeleton model of an object can be reconstructed from video.

A more indirect approach is motion inpainting. The idea is to derive a motion field for the hole, e.g., by gradually propagating motion vectors [9] or by using motion patch similarities [15]. This motion field is then used to propagate pixel values from outside the hole into the hole. These approaches allow inpainting only over a relatively small number of frames. Inpainting large time intervals is not straightforward, as pixel propagation tends to smooth the results too much.

3 Overview

The proposed method is inspired by several image and video inpainting algorithms. In particular, the operational structure of our algorithm is inspired by the shift-map framework [12] for image processing.

Input to our algorithm is a video sequence I that shows several dynamic scene elements, e.g., people, which occlude each other for certain periods of time. The input video sequence can be thought of as a *video volume* in which frames are stacked along the time dimension and each space-time pixel $I(x,y,t)$ is indexed with three coordinates. In a first step, we identify the moving elements of the scene throughout the video, and automatically provide spatio-temporal masks for their respective regions (cf. Fig. 1.1). Since these automatic masks may contain errors, we provide a simple interface in which the user can correct these tracked trajectories and manually refine the shape of masks (cf. Sec. 4). Given the masks, one moving scene element O_{rem} is specified by the user to be deleted from the footage. O_{rem} leaves a spatio-temporal hole $\mathscr{H}(O_{\text{rem}})$ in the 3D video volume (cf. Fig. 5.1) that needs to be filled, i.e., the background (static and dynamic), and every moving object that is occluded by O_{rem} has to be inpainted in a spatio-temporally coherent manner.

We attempt to find a spatio-temporal displacement for each missing pixel in $\mathscr{H}(O_{\text{rem}})$, such that it is an offset to a different pixel in the video volume that appropriately fills it in (cf. Sec. 5). Formally, this can be stated as follows: For a given video I and a hole $\mathscr{H}(O_{\text{rem}})$, we construct a *shift-volume* $M(x,y,t) = (d_x,d_y,d_t)^\top$ for $(x,y,t) \in \mathscr{H}(O_{\text{rem}})$. The output video R is constructed by assigning to each location $(x,y,t) \in \mathscr{H}(O_{\text{rem}})$ the color values of its shifted location, i.e., $R(x,y,t) = I((x,y,t)+M(x,y,t))$. A shift for each space-time pixel in the hole is found by minimizing a global energy functional that models the trade-off between two competing objectives: (a) The shifts should be as homogeneous as possible, which implies that the corresponding region is obtained from a spatio-temporally coherent segment in the video, and therefore appears natural; and (b)

the boundaries between different homogeneous regions in the shift-volume and across the boundary of the hole should not be objectionable, i.e., adjacent pixels should locally agree with each other.

We minimize this energy functional with a graph-cut-based algorithm [2]. In practice, a hierarchical approach is necessary, as it becomes computationally intractable to allow the algorithm to copy pixels into $\mathscr{H}(O_{\text{rem}})$ from anywhere in the video volume (cf. Sec. 6). Since we interactively follow the trajectory of each dynamic object, we can restrict the space of candidate pixels to a window around the object at time steps where it is unoccluded. We describe this process in the next section.

4 Moving Object Detection and User-guided Refinement

As described previously, the input to the proposed inpainting system is a video containing possibly many dynamic objects occluding each other on top of static or dynamic backgrounds. Our prototype preprocessing system first performs an automatic background estimation that is followed by a background subtraction yielding rough initial masks for each moving foreground object in each frame (see Fig. 1.1). Using a similar approach to [5], we first estimate a single background plate based on several frames regularly sampled from the video. Additionally, we estimate a camera noise model by sampling several non-dynamic pixel locations over the entire sequence. Then, we perform a graph-cut-based thresholding of the difference between each frame and the background. The graph-cut optimization assigns background-foreground labels based on the (lowest of) frame-background differences and the expected background noise. A simple Potts model ensures spatially consistent thresholds. The resulting masks are tracked over time, i.e., each object receives a temporally consistent label as long as they remain unoccluded. One of the objects found, O_{rem}, is selected by the user to be removed from the footage.

Automatic segmentation reliably identifies moving scene elements in most cases and provides initial masks, but some individual frame masks may contain small errors, i.e., they may contain holes or may contain other dynamic scene elements. In particular, the mask around O_{rem} should be sufficiently tight to avoid unnecessary inpainting. Therefore, we provide a graphical interface to browse through the video and to choose and refine the masks corresponding to O_{rem}. When the user refines a mask, the edit is propagated to other frames by optical flow [16]. Figure 4.1 shows a snapshot of the user interface. The readers are referred to the accompanying supplementary video for demonstration.

One can dramatically reduce the space of candidate pixels (in our case, the number

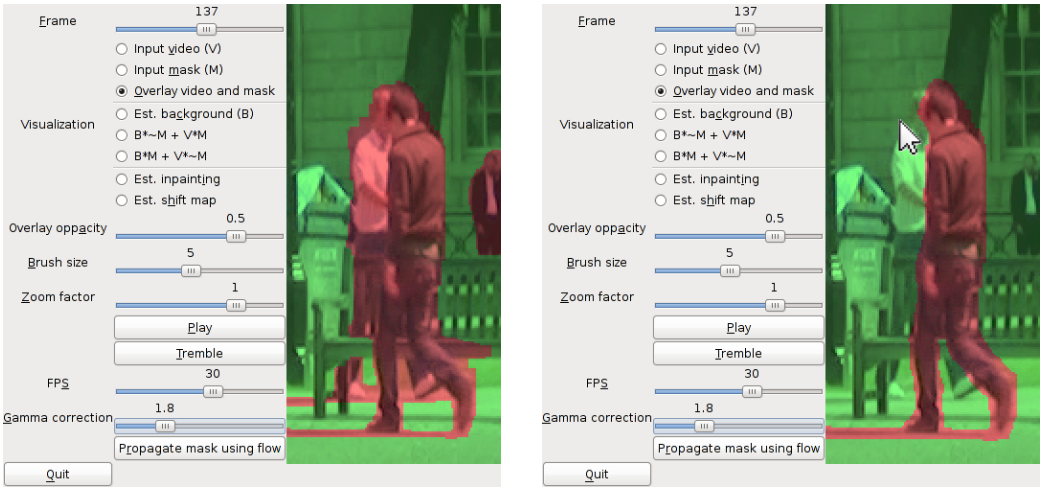


Figure 4.1: *User assisted mask editing interface. Graph-cut based segmentation (left). User refined segmentation (right).*

of labels to expand) by restricting it to moving windows that follow the dynamic objects O_{compl} to be inpainted. As an example, to fill in a person O_{compl} that is (at least partially) occluded by O_{rem}, we use pixels from time steps where the person was unoccluded. These pixels will lie in a space-time window that follows the person tightly. We provide a user interface (cf. Fig. 4.2) to quickly specify trajectories for these space-time windows on the initial segmentation result.

In our interface, the user can examine all $O_{compl,i}$ ($i \in \{0,\ldots,k\}$, k being the number of dynamic objects ever occluded by O_{rem}) from two diagrams. These diagrams are projections of the video volume onto the xt-plane and ty-plane, respectively. In these diagrams the path of each $O_{compl,i}$ traces a "tunnel" through the space-time slice. For each $O_{compl,i}$, the user marks this tunnel with a polyline in each projection. From the two projections, the moving (rectangular) window around the object in all time steps is now determined. It should be noted that since $O_{compl,i}$ can also occlude each other, their windows may intersect. As such, regions in the video volume in which trajectories intersect correspond to two or more people. As each window is inpainted independently, intersecting regions may be inpainted twice more. If two or more objects are occluded behind a removed object, the user has to decide on their depth ordering before compositing.

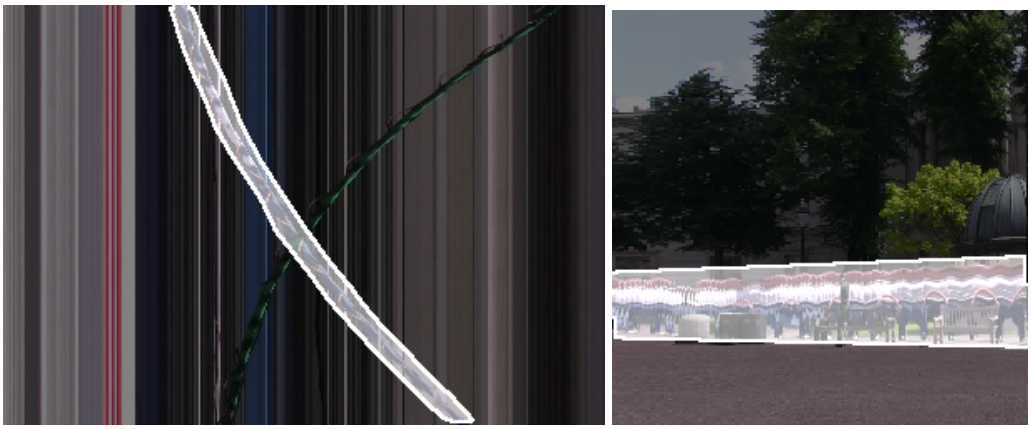


Figure 4.2: *User interface for occludee tracking: in order to reduce the storage and computation time, we restrict the space of possible shifts to the spatio-temporal region spanned by the occluded object O_{compl}. Here we display the region drawn by the user on top of an xt projection (left) and a ty projection (right) of the input video. Please note that we don't specify a tight bound on the object, but rather a fairly large window around it that can be quickly marked.*

5 Inpainting

From the previous processing step, we obtain relatively tight masks around O_{rem}, as well as bounding boxes for the scene elements to be filled in, $O_{compl,i}$. The latter do not need to be tight as they are required for bounding box tracking only. Removing O_{rem} means finding a shift volume (cf. Sec. 3), which we obtain by minimizing an energy functional described below. When inpainting a dynamic object, we make use of the observation that the set of candidate pixels should only contain those where the object was unoccluded, i.e., from a relatively narrow tracked bounding box around it. We therefore reduce the space of possible shifts by restricting them to the tracked bounding boxes. This increases run time performance by orders of magnitude without sacrificing quality. We can adapt this strategy, since the preprocessing stage provides all the mask and tracking information needed to constrain the shift space accordingly. More specifically, inpainting $\mathscr{H}(O_{\text{rem}})$ is performed by subsequently solving sub-inpainting problems as follows:

1. First, fill in the entire $\mathscr{H}(O_{\text{rem}})$ with background. This is achieved by solving an additional (constrained) inpainting problem, where each pixel is only allowed to shift along the temporal axis.
2. For each $O_{compl,i}$ specify the candidate database to be the unoccluded tracked window area of $O_{compl,i}$. Here, overlaps between different $O_{compl,i}$ may have to be resolved. Both defining the tracks and resolving overlaps can be conveniently done in the user interface (cf. Sec. 4).
3. Solve a separate inpainting problem for the occlusion time span of each $O_{compl,i}$.

Once the individual inpainting sub-problems are solved, the final video result is obtained by compositing the inpainted regions back into the video, such that occludee inpaintings replace background inpaintings where necessary. The energy we minimize for each sub-problem is defined in the remainder of this section.

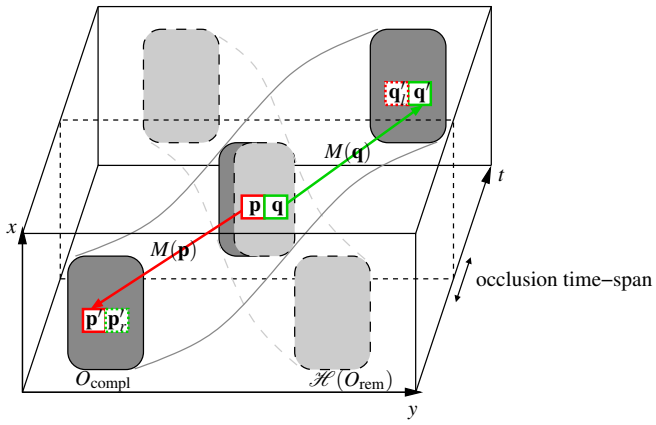


Figure 5.1: *Video volume inpainting. A pair of missing pixels* $\mathbf{p},\mathbf{q}$ *(white squares) inside the spatio-temporal hole* $\mathscr{H}(O_{\text{rem}})$ *(pink rectangles) can be replaced by pixels* $\mathbf{p}',\mathbf{q}'$ *outside the hole provided that their neighborhoods are color and gradient consistent, i.e., that the color and gradient mismatch between the pairs* $(\mathbf{p}',\mathbf{q}'_l)$ *and* $(\mathbf{p}'_r,\mathbf{q}')$ *is sufficiently low. For an inpainting to be satisfactory, such neighborhood consistency has to be maintained for all spatio-temporally adjacent pixels inside* $\mathscr{H}(O_{\text{rem}})$.

5.1 Energy functional

For notational simplicity, we use vectorial notations for denoting indices of spatio-temporal pixels, i.e., pixels $\mathbf{p},\mathbf{q} \in \Omega \subset \mathbb{Z}^3$. Here Ω is the index set for the entire input video. The shift-volume M is obtained as a minimizer of the cost functional

$$\mathscr{E}(M) = \sum_{\mathbf{p}\in\mathscr{H}(O_{\text{rem}})} \sum_{\mathbf{q}\in\mathscr{N}(\mathbf{p})} \mathscr{V}\left((\mathbf{p},M(\mathbf{p})),(\mathbf{q},M(\mathbf{q}))\right), \tag{5.1}$$

with the condition that all shift origins should be outside of the hole, i.e., $\forall \mathbf{p} \in \mathscr{H}(O_{\text{rem}}) : \mathbf{p}+M \in \Omega\backslash\mathscr{H}(O_{\text{rem}})$. Here, $\mathscr{N}(\mathbf{p})$ denotes the set of pixels adjacent (6-neighborhood in both space and time) to $\mathbf{p}$. For $\mathbf{q} \in \Omega\backslash\mathscr{H}(O_{\text{rem}}) \cap \mathscr{N}(\mathbf{p})$, $M(\mathbf{q})$ is not explicitly defined. In this case, we assign $\mathbf{0}$ to it.

The pair-wise smoothness term $\mathscr{V}$ represents the cost of discontinuities in the shift map. When such discontinuities occur, we measure the discrepancy of the corresponding pixel values, which in the following we will refer to as *observations*. Specifically, observations measure the distance of the color and gradient values within the local neighborhoods [7] from which the color values for $\mathbf{p}$ and $\mathbf{q}$ are drawn:

$$\mathscr{V}\left((\mathbf{p},M(\mathbf{p})),(\mathbf{q},M(\mathbf{q}))\right) = \begin{cases} 0 & \text{if } M(\mathbf{p}) = M(\mathbf{q}), \\ h(\mathbf{p},\mathbf{q}) \cdot \gamma(\mathbf{p},\mathbf{q}) & \text{otherwise,} \end{cases} \tag{5.2}$$

where

$$
\begin{aligned}
h(\mathbf{p},\mathbf{q}) = \; & \Big(\| \; I(\mathbf{p}+M(\mathbf{p})) - \; I(\mathbf{q}+M(\mathbf{p})) \|_2^2 \quad + \\
& \quad \| \; I(\mathbf{q}+M(\mathbf{q})) - \; I(\mathbf{p}+M(\mathbf{q})) \|_2^2 \Big)^{\psi} + \\
& \beta \Big(\| \nabla I(\mathbf{p}+M(\mathbf{p})) - \nabla I(\mathbf{q}+M(\mathbf{p})) \|_2^2 \quad + \\
& \quad \| \nabla I(\mathbf{q}+M(\mathbf{q})) - \nabla I(\mathbf{p}+M(\mathbf{q})) \|_2^2 \Big)^{\psi} \; ,
\end{aligned}
\tag{5.3}
$$

where I is the input color video, and β is the weight balancing the contribution of gradient and color values. We fix β at $(2\sqrt{2})^{-1}$ throughout all experiments.[1]

In our approach, the exponent ψ in Eq. (5.3) is fixed to $\frac{1}{2}$ to make sure that the resulting cost function is *regular*. If the cost function is not regular, it is not guaranteed that our graph cut-based optimization finds a solution close to the global minimum [3]. Empirically, we also observe slower convergence with irregular energies.

Pixels that are close to the boundary of the hole deserve special treatment. As currently specified, $\mathscr{V}$ is undefined for pixels $\mathbf{q}$ on the boundary, i.e., $\mathbf{q} \in \partial(\Omega \backslash \mathscr{H}(O_{\text{rem}}))$ (see Fig. 5.2). Since $\mathbf{q}$ is outside the hole, it is not allowed to be inpainted, i.e., we have a hard constraint $M(\mathbf{q}) = \mathbf{0}$. In that case the values $I(\mathbf{p}+M(\mathbf{q}))$ and $\nabla I(\mathbf{p}+M(\mathbf{q}))$ for $\mathbf{p} \in \mathscr{H}(O_{\text{rem}})$ would be drawn from inside the hole, and therefore would be invalid as pixels inside the hole should not be used as reference. We solve this issue by relaxing the hard constraint on the boundary so that pixels $\mathbf{q}$ in that boundary can be inpainted, as illustrated in Fig. 5.2.

Any inpainting algorithm is required to ensure both consistency at the boundary of the missing region and self-consistency inside it, two criteria that are equally important. However, since usually there are many more pixels in the interior of a hole than on its boundary, the current version of the energy function implicitly weighs the latter criterion higher. Note also that pixels close to the boundary usually contain a lot of information for correct inpainting. Usually, occludees are only partially occluded, and, for instance, a leg or an arm of a person is often still visible outside the hole. That information should preferably be propagated to the inpainted region.

In order to encode these two observations, we introduce the factor $\gamma(\mathbf{p},\mathbf{q})$ as a weight that is inversely proportional to the distances of $\mathbf{p}$ and $\mathbf{q}$ to the boundary of the hole $\partial\mathscr{H}(O_{\text{rem}})$. For the construction of γ, we firstly assign to each pair

[1] The range of gradient differences is twice as large that of pixel differences (hence the $1/2$ factor), and their variance (assuming linear camera response) is also twice as large (hence the $1/\sqrt{2}$ factor).

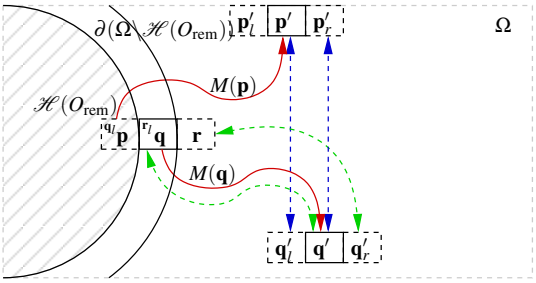


Figure 5.2: *In addition to inpainting pixels* $\mathbf{p} \in \mathscr{H}(O_{\text{rem}})$*, we also allow pixels* $\mathbf{q}$ *that are adjacent to the hole, i.e.,* $\mathbf{q} \in \partial(\Omega\backslash\mathscr{H}(O_{\text{rem}}))$*, to be inpainted. This prevents their neighbors which are inside the hole (*$\mathbf{q}_l$ *in the figure) from becoming undefined. Here, red lines denote shift origins, and blue and green lines denote the pixel differences evaluated in our energy Eq. (5.1), namely* $\mathscr{V}((\mathbf{p},M(\mathbf{p})),(\mathbf{q},M(\mathbf{q})))$ *and* $\mathscr{V}((\mathbf{q},M(\mathbf{q})),(\mathbf{r},\mathbf{0}))$*, respectively.*

$(\mathbf{p},\mathbf{q})$ the average of distances of $\mathbf{p}$ and $\mathbf{q}$ from the boundary

$$d_0(\mathbf{p},\mathbf{q}) = \frac{1}{2}(d(\mathbf{p},\partial\mathscr{H}(O_{\text{rem}})) + d(\mathbf{q},\partial\mathscr{H}(O_{\text{rem}}))), \tag{5.4}$$

where the distance $d(A,\mathscr{B})$ between a point A and a set $\mathscr{B}$ is defined as the minimum of the city-block distance between A and the elements of $\mathscr{B}$. The contour lines of d_0 define an equivalence relation in $\mathscr{H}(O_{\text{rem}})$. For the pair $(\mathbf{p},\mathbf{q})$, the corresponding equivalence class $[\mathbf{p},\mathbf{q}]$ is defined as

$$[\mathbf{p},\mathbf{q}] = \{(\mathbf{m},\mathbf{n}) | d_0(\mathbf{m},\mathbf{n}) = d_0(\mathbf{p},\mathbf{q})\}.$$

Furthermore, by straightforwardly extending d_0 to equivalence classes, we can induce a linear ordering for the equivalence classes:

$$[\mathbf{p},\mathbf{q}] < [\mathbf{m},\mathbf{n}] \text{ if } d_0([\mathbf{p},\mathbf{q}]) < d_0([\mathbf{m},\mathbf{n}]). \tag{5.5}$$

The next step is to make sure that the distinct equivalence classes are sufficiently separated. Therefore, we define a weight γ_1 which guarantees that:

$$2\gamma_1([\mathbf{p},\mathbf{q}])|[\mathbf{p},\mathbf{q}]| \geq \gamma_1([\mathbf{m},\mathbf{n}])|[\mathbf{m},\mathbf{n}]| \text{ if } [\mathbf{p},\mathbf{q}] \leq [\mathbf{m},\mathbf{n}],$$

where $|[A]|$ is the number of elements in the equivalence class $[A]$. Let us denote $i \in [0,1,\dots]$ as the index in the linear ordering of equivalence relation and $[i]$ the corresponding equivalence class. Then, γ_1 can be explicitly defined by the recurrence relation

$$\gamma_1(i+1) = \gamma_1(i)\frac{|[i+1]|}{2|[i]|}, \tag{5.6}$$

with $\gamma_1(0) = 1$.

Lastly, it is crucial for video inpainting to properly balance the relative importance of spatial mismatches and temporal mismatches. To allow control of the temporal importance, one could replace the metric d in Eq. (5.4) by a metric of the form:

$$d_1^2(\mathbf{p},\mathbf{q}) = (p_x - q_x)^2 + (p_y - q_y)^2 + \alpha(p_t - q_t)^2,$$

with a parameter α weighting the relative importance of the time domain. For 6-neighbor system in space and in time, like ours, this has a similar effect to defining γ as

$$\gamma(\mathbf{p},\mathbf{q}) = \begin{cases} \alpha\gamma_1(\mathbf{p},\mathbf{q}) & \text{if } \mathbf{p}-\mathbf{q} = (0,0,\pm 1) \\ \gamma_1(\mathbf{p},\mathbf{q}) & \text{otherwise.} \end{cases} \tag{5.7}$$

The parameter α is fixed at 2 throughout all our experiments, which equalizes the contributions of the temporal dimension and the combined two spatial dimensions.

5.2 Optimization

The energy functional (Eq. (5.1)) is non-convex and finding a global minimum is difficult. Instead, we find an approximate solution using graph cuts [3, 2], where each individual node corresponds to a pixel in $\mathscr{H}(O_{\text{rem}})$, and the set of potential labels for each node is the set of all possible shifts.

Directly minimizing the cost functional using graph-cuts is still a challenging problem due to the very large size of the corresponding graph. Similar to [12], we adopt a multi-resolution approach. First, a multi-resolution video pyramid is generated by iterating through the process of reducing the resolutions of spatial dimensions by half such that the lowest level of the pyramid, the minimization of Eq. (5.1) is tractable. It should be noted that down-sampling is not performed along the temporal dimension as it can introduce discontinuities. Once the solution (the shift-volume) is found at the smallest scale, it is up-sampled to an initial guess for the next level (i.e., a higher-resolution) using nearest neighbors interpolation. The magnitudes within the shift-volume are doubled to match the higher resolution. This step is repeated until we reach the original resolution.

On the lowest pyramid level, the size of the label space is $(\frac{2w}{n} - 1)(\frac{2h}{n} - 1)(2t - 1) \sim \frac{w \cdot h \cdot t}{n^2}$, where w, h, t are respectively the width, height, and length of the tracked window of a given O_{compl}, and n is the reduction factor of the coarsest pyramid level. For background inpainting, the size of the label space is $(2t - 1)$, as we do not allow spatial shifts. In this work, we set n such that the number of nodes in the

graph is smaller than 100^3. This way optimization remains feasible on standard computing hardware.

On all levels above the lowest one, only small shifts relative to the initial estimate are examined. In our implementation, we use three relative shifts $(-1,0,1)$ in each spatial and temporal coordinate. Unfortunately, on these pyramid levels our energy function is no longer regular. Any two adjacent pixels $\mathbf{p},\mathbf{q}$ that have different labels in the coarsest level might be assigned the same refinement label $(-1,0,1)$. This makes the energy $\mathscr{V}(\mathbf{p},\mathbf{q})$ zero, and the energy non-regular as $\mathscr{V}$ is no longer metric. In practice, this does not pose a problem as global shifts are already decided in the coarsest resolution, where the regularity condition is upheld. Please note that the energy used by [12] is irregular at every resolution (a quadratic penalizer is used), and irregularity leads to overly smooth solutions in our setting (see also Sec. 5.3).

5.3 Design Validation

Figure 5.3e shows that both the distance weighting function γ, as well as the proper control of the relative importance of time α are important to obtain high-quality inpainting results. If only the time domain is given higher impact ($\alpha = 2$), but distances to the boundary are not used for weighting ($\gamma_1 = 1$), inpainting the background erroneously turns into a solution with low energy, Fig. 5.3c. In this case, the interior of the hole is coherent, but the clearly visible discontinuities at the hole boundaries were not weighted sufficiently high.

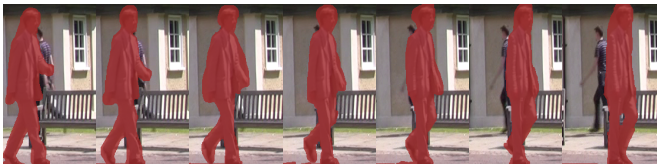
On the other hand, if we employ distance weighting γ_1 but do not increase the importance of temporal mismatches ($\alpha = 1$), we get spatially coherent (for individual frames) but temporally incoherent inpainting results, as can be seen, for instance, at the incorrectly located head in Fig. 5.3d. The importance of properly considering the time domain is also documented in the literature on visual perception. Wandell [18] states that the human sensitivity to temporal contrast changes peaks roughly between 5 Hz to 10 Hz. In other words, temporal aliasing at about half the video frame rate is very objectionable to the viewer. If the time domain is not re-weighted, as in this example, we can suffer temporal aliasing in that range.

Please note that Wexler et al. [19] also use distance weighting in their energy. However, their weighting strategy is much simpler and equivalent to ours only for the case of $\alpha = 1$ and a spherical hole, which is rather uncommon in reality.

We have also implemented the energy function of Pritch et al. [12], extended it from 2D to 3D, and used it to inpaint the hole (the only difference to the original

paper is that we work on individual pixels and not on patches). Figure 5.3b shows that this does not lead to pleasing inpainting results, as that energy is irregular and does not incorporate distance and time weighting.

Finally, we would like to point out that the irregular variant of our proposed energy, which we obtain by using a quadratic penalizer ($\psi = 1$), does not produce the desired outcome and leads to over-smoothing and washed out inpainting results, Fig. 5.3h. In contrast, our proposed regular energy function that features distance weighting and properly trades off spatial versus temporal mismatches is important to achieve high quality results.



(a) Input with occluder mask

(b) extension of [12] to 3D

(c) $\gamma_1 = 1$, $\alpha = \{1,2\}$

(d) γ : Eq. (5.7), $\alpha = 1$

(e) γ : Eq. (5.7), $\alpha = 2$ (proposed)

(f) γ : Eq. (5.7), $\alpha = 2$, $\psi = 1$ (irregular)

Figure 5.3: *The effect of the distance weighting γ and the temporal weight α. (a) Input video and overlaid mask. (b) Straightforward extension of shift-map to 3D with irregular smoothness term. (c) No distance weighting ($\gamma_1 = 1, \alpha = 1$) but using a regular term (the result for $\gamma_1 = 1, \alpha = 2$ is identical). (d) Using γ as defined in Eq. (5.7) and $\alpha = 1$, the result is spatially consistent but shows slight temporal misalignments. (e) Proposed method: With γ as defined in Eq. (5.7) and $\alpha = 2$, the balance between spatial and temporal consistency is kept. (f) Same as (e) but using an irregular functional, where optimality guaranties are lost. The video corresponds to the fifth occlusion of* park complex.

6 Results

We tested our algorithm on four videos, which will henceforth be referred to as *beach-umbrella, park-simple, park-complex*, and *museum*. Complete results are shown in the supplementary video. Each video exhibits different scene complexity in terms of the number of moving objects, the number of occlusions, and illumination conditions (see Table 6.1). The beach umbrella sequence is of comparably low resolution (271×80), whereas the other sequences are shot in HD (1920×1080 or 1440×1080). The sequence length varies between 100 and 450 frames. In our experiments, we focused on inpainting people, which is very relevant in real productions. However, it should be noted that our algorithm is generic and can be applied to any type of objects under any motion.

The removal tasks performed on the different sequences vary in difficulty, in particular due to the different numbers of moving objects and respective occlusions. In the *park-simple* sequence, we remove a person that only occludes one other moving person (Figs. 1.2 and 6.4). In contrast, *park-complex* is a much more crowded scene (Fig. 6.1). Here, we also remove one person that walks through

Dataset	Video size	Missing pixels	Occlusions	Missing pixels per occlusion	Search space per occlusion
park-simple	Full HD$\times$251	10^6	3	10^4 – 10^6	10^6
park-complex	Full HD$\times$459	10^6	8	10^4 – 10^6	10^6 – 10^7
museum	Full HD$\times$200	10^7	8	10^5 – 10^6	10^6 – 10^7
beach-umbrella	$271 \times 80 \times 98$	10^5	3	10^5	10^6

Table 6.1: *Summary of our experimental dataset. Due to its sufficiently small size, we treat the* beach-umbrella *sequence as a single occlusion, i.e. without tracking the occludee, thus solving a single inpainting problem with the complete video volume as search space.*

the entire scene from left to right, but now we have to inpaint 8 people occlusions (a couple sitting on a bench and chatting is inpainted in one window) in different motions. *museum* is the most crowded and most challenging scene with many occlusions in multiple layers (Fig. 6.2). Here, we also remove a person crossing the entire scene from right to left and we have to inpaint eight occluded people that are standing or strolling in different directions. Algorithmically removing people from as complex scenes as *park-complex* and *museum* has not been demonstrated in the literature before. The beach-umbrella sequence was originally used in [19]. The task is to remove a largely static object (umbrella) which is occluding three persons moving in front of a dynamic background (ocean). Unlike other videos, we treat this sequence as a single occlusion, i.e. without tracking the occludee, thus solving a single inpainting problem with the complete video volume as candidate space, just as it was done in [19]. This can be regarded as an application example of the proposed algorithm for dynamic background. For all the videos, the size of objects to be inpainted range between 64×44 to 384×512. The actual search space of our algorithm is in order of 10^6 – 10^7 *per* pixel (see Table 6.1).

The occlusion in *park-simple* is inpainted without noticeable artifacts. The algorithm plausibly synthesizes the motion of the occludee where it was originally unseen, Figs. 1.2 and 6.4.

In the museum scene, a person walking across the camera's view is removed, Fig. 6.2. This required inpainting 8 people at different distances from the camera, with different motions, and with different paths across a reflective floor. It is noteworthy that the algorithm successfully inpainted the person walking away from the camera (the 2nd person in the figure) which exhibits strong size variation due to perspective foreshortening. Also, the method succeeds in inpainting reflections from the floor which is especially noticeable in the 2nd and 3rd person.

During inpainting, our algorithm successfully masters very challenging scene conditions, such as changing lighting and non-periodic motions that are non-parallel to the image plane. We would like to discuss this in more detail for *park-complex*. Figure 6.1 shows the individual 8 inpainting cases solved for the entire video. Inpainting cases 1 and 6-8 deal with people that are non-periodically moving in the background, e.g., person 8 stands up while being half-occluded, and person 1 starts to walk right after the occlusion. In all these cases, the inpainting results are very convincing with almost no visual artifacts. In case 5 a couple in conversation sitting on a bench is inpainted. The final result is highly realistic. Case 4 is a demanding situation since the person to be inpainted is walking very fast and during the occlusion by $\mathscr{H}(O_{\text{rem}})$ is himself occluded by a static obstacle, namely the bench. Still, our algorithm successfully inpainted the person as well as the bench, even though both these scene elements are never seen in

the video in the exact same combination. Another challenging case is 2, where a person walking towards the camera is inpainted. While the body and motion of the person is realistically inpainted, the method is challenged by the fact that the occludee lifts his arm wile being occluded, but that motion is never seen somewhere else. Therefore, slight inaccuracies are seen in the arm. Similarly, in case 3, the person is turning on the spot and walking away from the camera. During occlusion, the person shifts their weight from the left to the right leg, a movement that is never seen unoccluded at any other time instant. The appearance of the person also changes after the occlusion since he walks into a shadowed part of the scene, which presents an additional challenge. Consequently, a slight discontinuity appears in the inpainted person. Please note that the latter two cases could not easily be solved by any data-driven inpainting approach, since the parts where the inpainted person is unoccluded are too different from what would be needed to correctly inpaint the hole. Nonetheless, in such cases the automatic inpainting result only exhibits small artifacts that could be manually corrected from our result.

6.1 Comparison

We compared our algorithm to the approach of [19] on two sequences, *beach-umbrella* and *park-simple* (Figs. 6.3 and 6.4). This algorithm refrains from significantly restrictive assumptions (e.g., cyclic motion and directions of motion with respective to camera's view, etc.) and accordingly can be directly compared to ours.[1]

For the *beach-umbrella* sequence, we use the mask for the hole provided by Wexler et al. on their project web page. For *park-simple*, we use the masks generated during our segmentation and refinement step such that both [19] and our algorithm benefit from restricting the search space (or database) for each object to be inpainted. Accordingly, here we only compare the energy functional and corresponding optimization algorithms.

We use our own implementation of the method by Wexler et al. that we prepared after extensive conversation with the authors of the paper. In their paper, they describe two options for voting on a final pixel value from candidates in a spatio-temporal neighborhood, namely using mean shift or using a weighted average, which we both implemented. In the end, we only use the weighted averaging variant since the mean shift variant did not terminate on the umbrella sequence even

[1] We also compared our method with a hypothetical 3D extension of shift-map in Sec. 5.3.

after 9 days of CPU time on a 2.66GHz Xeon X5560 CPU (using a single core). We are confident that our implementation is accurate, since Wexler et al.'s approach inpaints the videos hierarchically, and the mean shift variant terminated on lower resolution levels with plausible results, but not on the full resolution level. Wexler's result with weighted averaging took 26 hours to compute on the same machine, whereas our approach finished after 14 hours. Unfortunately, even after exhaustive grid search of parameters for the weighted averaging variant, we were not able to find settings that would plausibly inpaint the moving ocean in the background. This area always looked washed out and static (the most visually plausible result was obtained with $\gamma = 1.3$, see Fig. 6.3c and supplementary video). We expect the mean shift variant to perform better on dynamic backgrounds, but it seems it requires dramatically longer computation times. Nonetheless, we downloaded the final video result that was computed with the mean shift variant from the project web page of Wexler as a reference, see Fig. 6.3b. Both in our result and the downloaded result, the people and the dynamic background are very realistically inpainted. However, the above evidence suggests that our approach produces such high quality results much faster.

We performed the same comparison on the *park-simple* scene using our implementation of the weighted averaging variant. Figure 6.4 shows that the results of our method look sharper and crisper, and are more complete. In Wexler et al.'s result parts of the arm are missing in the reconstruction and the bench is very blurry, whereas in our approach the moving arm is faithfully filled in and the bench looks sharp and realistically inpainted. This example illustrates that the implicit preference of our energy function to shift large coherent regions in the shift volume is a conceptual advantage. Second, by construction, our inpainted pixel values consist solely of pixel values visible elsewhere in the video. That is, our method does not create new pixel values (e.g., by averaging) in favor of minimizing Eq. (5.3). This might have resulted in blurred images as the only null space of the cost (Eq. (5.3)) is a constant image. In Wexler et al.'s approach, the final pixel values are weighted combinations of existing pixel values and, in combination with L^2 norm, this may lead to blurring in the final results.

6.2 Parameter Selection

For all our results we use the same set of parameters, namely $\alpha = 2$, $\beta = (2\sqrt{(2)})^{-1}$, and $\psi = 1/2$ (see Sec. 5.1 and Sec. 5.3 for a discussion on these specific choices). This shows the robustness and stability of our approach across different types of scenes. In addition, for all experiments, we restricted the possible values of

M in the direction of the y-coordinate to be within $[-16, 16]$, i.e., occludees are not rapidly approaching or moving far away from the camera. This additional constraint was introduced to speed up the experiments and is not a fundamental requirement of applying the proposed algorithm.

6.3 Timings

The examples that we present in the next section took approximately 16 hours to inpaint per occlusion (unoptimized code), with candidate sets in the order of 10^6–10^7 shifts (see Table 6.1) on a single core of a Xeon X5560 CPU.

For each of our examples, it took at most 60 minutes of manual touch up to remove errors in the tracked mask of $\mathscr{H}(O_{\text{rem}})$. Marking the trajectories of people in the volume projections took less than a minute per occlusion.

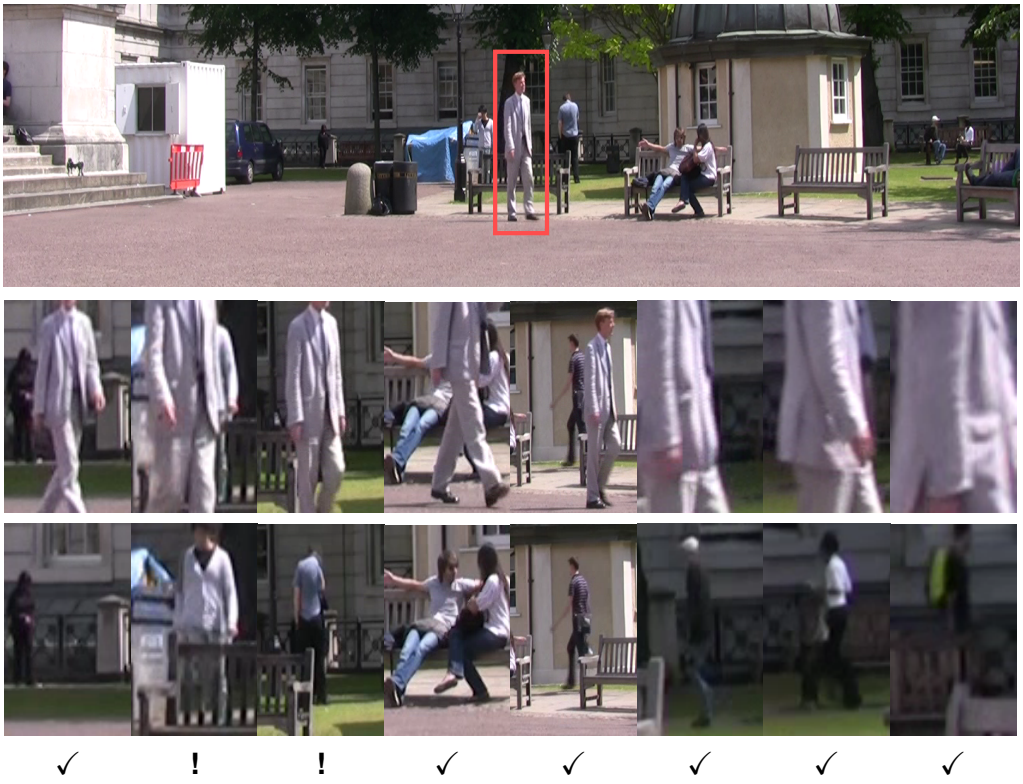


Figure 6.1: *Inpainting of the* park-complex *video: Crop of the the input video, where the person to be removed is marked in red (top); sample frame from each of the eight people occlusions in the input video (middle); result of our inpainting algorithm (bottom). Successful and challenging inpaintings are marked with* ✓ *and* **!***, respectively.*

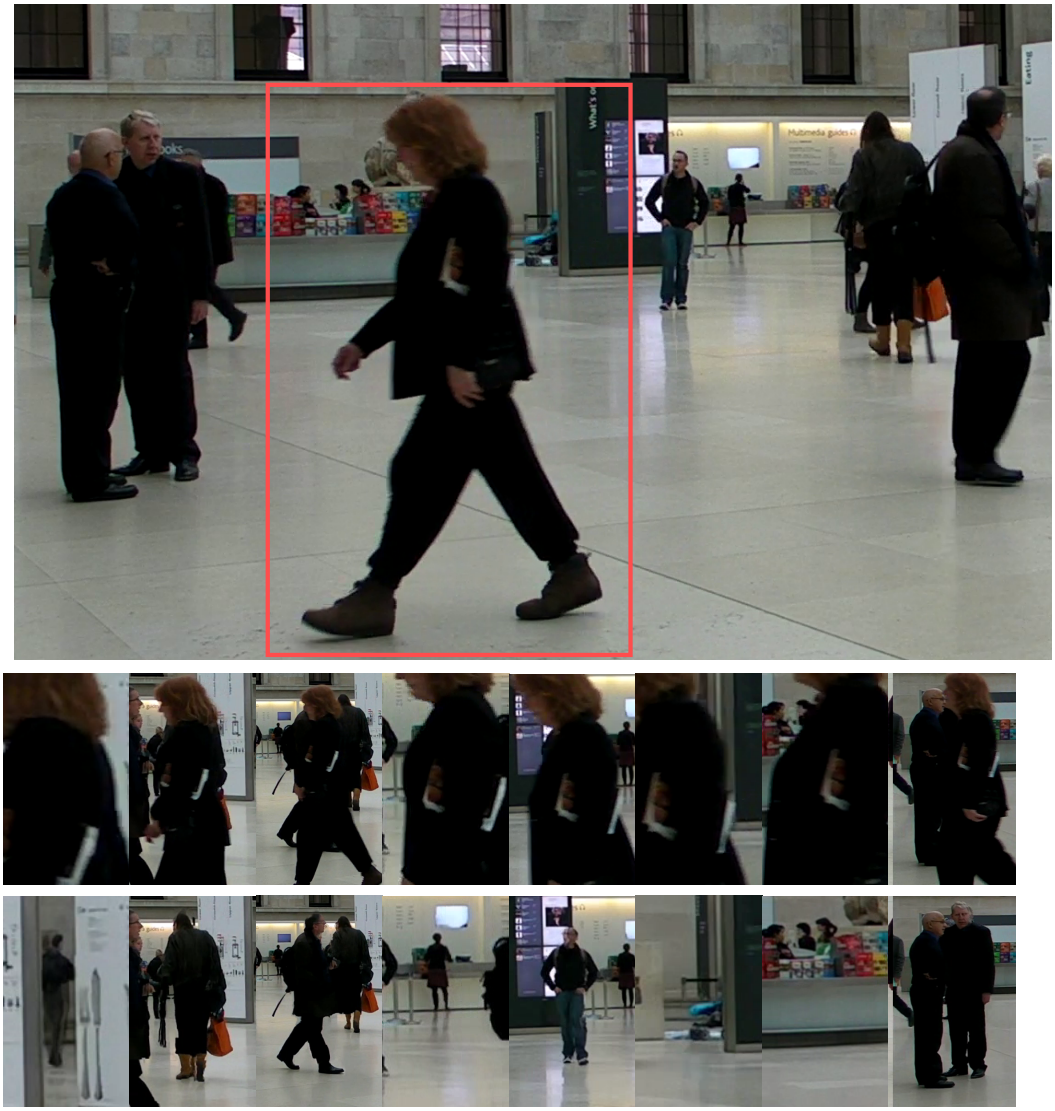


Figure 6.2: *Inpainting of the* museum *video. Crop of the the HD video, with the inpainted person marked in red (top). Sample frames of each of the eight occluded moving objects (middle). Our inpainting result (bottom).*

(a) (b) (c) (d)

Figure 6.3: *Comparison on the* umbrella *sequence by Wexler et al.: example of a full frame where the hole is shown overlaid in red (top row); (a) input occlusions, (b) result by Wexler et al. with mean shift, (c) Our implementation of Wexler et al. with weighted average, and (d) our result.*

Figure 6.4: *Subset of input video and overlaid mask (top). Result by our implementation of Wexler et al. (middle). Inpainting result of our method (bottom, β=0.25, 3 pyramid levels). Images shown are two frames apart.*

7 Discussion

This paper presents a shift-volume-based method for video inpainting. The underlying idea is to assign to each pixel in a given hole the color values of a visible pixel, such that the resulting inpainted hole looks natural and is consistent with the hole boundary. By encoding these requirements in a global energy functional, inpainting can be regarded as selecting, for each pixel in the hole, the most compatible visible pixel in a way that the resulting energy is minimized. One innovation that sets us apart from prior work is a very efficient reformulation of the energy minimization problem: by tracking each moving object that is occluded by the hole, we can significantly reduce the search space, and essentially reduce it from the entire video to the trajectories of objects. This is facilitated by automatic segmentation which can optionally be corrected in our newly designed user interface.

We use graph-cuts for minimizing the energy functional. What is different from closely related prior art is that the regularity of our energy functional guarantees that the solution obtained is close to a global minima. Further on, we experimentally validated that the distance weighting and proper balancing of temporal and spatial mismatches are crucial in the context of video inpainting.

In combination, this produces one of the first approaches to remove people in such complex crowded scenes with many occlusions and arbitrarily moving dynamic occludees. Our results indicate that the performance of the proposed method is significantly better than the method of [19] and than a direct extension of 2D shift-map editing to the video domain [12].

7.1 Limitations and Future work

A limitation of our algorithm is that by design, the synthesized color values filling in the holes are coming from the visible regions of the input video (cf. the second and third occlusion in Fig. 6.1). Accordingly, the system might fail if at the time of occlusion, the objects have different appearances (due to lighting changes, motion or deformation) than in regions where they are visible. We have discussed these case already in Sec. 6 and the video. To our knowledge, no approach that is designed to work on general scenes and refrains from strict model assumptions about scene content can fully-automatically handle such cases. One possibility to overcome this limitation is to use additional user input: e.g., the user could explicitly mark the spatio-temporal regions from where to inpaint. Also, for specific applications, e.g. the inpainting humans, one may resort to stronger model assumptions, such as motion and shape models that could be tracked and used to constrain the origins of pixel shifts in a semantically meaningful way. However, this would sacrifice flexibility.

Currently, we are not using motion or acceleration information from the video when inpainting a hole. As such, we cannot explicitly encourage preservation of coherent motion patterns in the inpainting result. The energy function could be extended to enforce similarity between source and target in terms of higher order derivatives. Especially when inpainting larger holes this information might be useful. However, computing higher order derivatives is potentially unstable and further increases computation time.

Due to the chosen background estimation approach we currently expect the camera to be static. Please note that this is due to our specific implementation of the pre-processing, and not a limitation of the actual inpainting approach which would succeed equally well with a dynamic camera. Segmentation approaches for dynamic backgrounds exist. Also, in the *beach-umbrella* sequence we successfully inpainted moving ocean waves in the background. This confirms that moving cameras/backgrounds can be handled.

In the current system, we have tried to keep the amount of user interaction at a moderate level. Except for the process of masking the object to be removed (which is required for any inpainting system), the only required user interactions are for refining the spatio-temporal tracks for each object to inpaint. We expect that in general the performance of the system will improve if more user inputs are supplied. For example, one could ask users to provide tighter mask for objects to be inpainted. Future work should investigate various possibilities for user interactions.

Our energy function as defined in Eq. (5.1) is not scale invariant. This makes it difficult to cope with objects moving away from or towards the camera and for cases where the missing scale was not observed. We can handle such situations provided that the scale does not change significantly during the occlusion, as demonstrated in Fig. 6.2 and the video. This limitation could be overcome, for instance, by allowing shifts along the video scale-space.

In the future, we would like to use video inpainting as a building block for other applications. For instance, one could inpaint every occlusion of every person in a video sequence, such as the *museum* clip, yielding loop-able tracks for every single person. This would enable the creation of novel, infinitely long sequences, by overlaying the tracks in varying order.

8 Conclusion

This paper presented a shift-volume-based method for video inpainting. We assign each pixel in a given hole the color values of a visible pixel, such that the resulting inpainted hole looks natural and consistent with the hole boundary. By encoding these requirements in a global energy functional, inpainting can be regarded as selecting for each pixel in the hole the most compatible visible pixel, such that the resulting energy is minimized. This lead us to a very efficient reformulation of the problem: By following the trajectory of each occluded object, we can reduce the search space from the entire video volume to tracked windows around occludees. This is facilitated by an interface through which the user can refine occluder masks and mark occludee trajectories. We use graph-cuts for minimizing the energy functional. The regularity of our energy functional guarantees that the solution obtained is close to the global minimum. We have evaluated our system on a set of videos with include two very complex situations. The results indicate that the performance of the proposed method is significantly better than the method of [19] and than a naive extension of the shift-map method to the video domain.

Bibliography

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 28(3):24:1–24:11, 2009.

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI*, 26(9):1124–1137, 2004.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23(11):1222–1239, 2001.

[4] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE TIP*, 13(9):1200–1212, 2004.

[5] M. Granados, H.-P. Seidel, and H. P. A. Lensch. Background estimation from non-time sequence images. In *Proc. Graphics Interface*, pages 33–40, 2008.

[6] J. Jia, Y.-W. Tai, T.-P. Wu, and C.-K. Tang. Video repairing under variable illumination using cyclic motions. *IEEE TPAMI*, 28(5):832–839, 2006.

[7] V. Kwatra, A. Schödl, I. A. Essa, G. Turk, and A. F. Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 22(3):277–286, 2003.

[8] C.-H. Ling, C.-W. Lin, C.-W. Su, H.-Y. M. Liao, and Y.-S. Chen. Video object inpainting using posture mapping. In *Proc. ICIP*, pages 2785–2788, 2009.

[9] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum. Full-frame video stabilization with motion inpainting. *IEEE TPAMI*, 28(7):1150–1163, 2006.

[10] K. Patwardhan, G. Sapiro, and M. Bertalmio. Video inpainting under constrained camera motion. *IEEE TIP*, 16(2):545–553, February 2007.

[11] K. A. Patwardhan, G. Sapiro, and M. Bertalmio. Video inpainting of occluding and occluded objects. In *Proc. ICIP*, pages 69–72, 2005.

[12] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *Proc. ICCV*, pages 151–158, Kyoto, Sept 2009.

[13] Y. Shen, F. Lu, X. Cao, and H. Foroosh. Video completion for perspective camera under constrained motion. In *Proc. ICIP*, volume 3, pages 63–66, 2006.

[14] T. K. Shih, N. C. Tan, J. C. Tsai, and Z. H.-Y. Video falsifying by motion interpolation and inpainting. In *Proc. IEEE CVPR*, pages 1–8, 2008.

[15] T. Shiratori, Y. Matsushita, X. Tang, and S. B. Kang. Video completion by motion field transfer. In *Proc. IEEE CVPR*, pages 411–418, 2006.

[16] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *Proc. IEEE CVPR*, pages 2432–2439. IEEE, 2010.

[17] M. V. Venkatesh, S. S. Cheung, and J. Zhao. Efficient object-based video inpainting. *Pattern Recognition Letters*, 30(2):168–179, 2009.

[18] B. A. Wandell. *Foundations of Vision*. Sinauer Associates, Inc., 1995.

[19] Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *IEEE TPAMI*, 29(3):463–476, 2007.

Below you find a list of the most recent research reports of the Max-Planck-Institut für Informatik. Most of them are accessible via WWW using the URL `http://www.mpi-inf.mpg.de/reports`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
– Library and Publications –
Campus E 1 4

D-66123 Saarbrücken

E-mail: `library@mpi-inf.mpg.de`

MPI-I-2009-RG1-002	P. Wischnewski, C. Weidenbach	Contextual rewriting
MPI-I-2009-5-006	S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, G. Weikum	Scalable phrase mining for ad-hoc text analytics
MPI-I-2009-5-004	N. Preda, F.M. Suchanek, G. Kasneci, T. Neumann, G. Weikum	Coupling knowledge bases and web services for active knowledge
MPI-I-2009-5-003	T. Neumann, G. Weikum	The RDF-3X engine for scalable management of RDF data
MPI-I-2008-RG1-001	A. Fietzke, C. Weidenbach	Labelled splitting
MPI-I-2008-5-004	F. Suchanek, M. Sozio, G. Weikum	SOFI: a self-organizing framework for information extraction
MPI-I-2008-5-003	F.M. Suchanek, G. de Melo, A. Pease	Integrating Yago into the suggested upper merged ontology
MPI-I-2008-5-002	T. Neumann, G. Moerkotte	Single phase construction of optimal DAG-structured QEPs
MPI-I-2008-5-001	F. Suchanek, G. Kasneci, M. Ramanath, M. Sozio, G. Weikum	STAR: Steiner tree approximation in relationship-graphs
MPI-I-2008-4-003	T. Schultz, H. Theisel, H. Seidel	Crease surfaces: from theory to extraction and application to diffusion tensor MRI
MPI-I-2008-4-002	W. Saleem, D. Wang, A. Belyaev, H. Seidel	Estimating complexity of 3D shapes using view similarity
MPI-I-2008-1-001	D. Ajwani, I. Malinger, U. Meyer, S. Toledo	Characterizing the performance of Flash memory storage devices and its impact on algorithm design
MPI-I-2007-RG1-002	T. Hillenbrand, C. Weidenbach	Superposition for finite domains
MPI-I-2007-5-003	F.M. Suchanek, G. Kasneci, G. Weikum	Yago : a large ontology from Wikipedia and WordNet
MPI-I-2007-5-002	K. Berberich, S. Bedathur, T. Neumann, G. Weikum	A time machine for text search
MPI-I-2007-5-001	G. Kasneci, F.M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum	NAGA: searching and ranking knowledge
MPI-I-2007-4-008	J. Gall, T. Brox, B. Rosenhahn, H. Seidel	Global stochastic optimization for robust and accurate human motion capture
MPI-I-2007-4-007	R. Herzog, V. Havran, K. Myszkowski, H. Seidel	Global illumination using photon ray splatting
MPI-I-2007-4-006	C. Dyken, G. Ziegler, C. Theobalt, H. Seidel	GPU marching cubes on shader model 3.0 and 4.0
MPI-I-2007-4-005	T. Schultz, J. Weickert, H. Seidel	A higher-order structure tensor

MPI-I-2007-4-004	C. Stoll, E. de Aguiar, C. Theobalt, H. Seidel	A volumetric approach to interactive shape editing
MPI-I-2007-4-003	R. Bargmann, V. Blanz, H. Seidel	A nonlinear viseme model for triphone-based speech synthesis
MPI-I-2007-4-002	T. Langer, H. Seidel	Construction of smooth maps with mean value coordinates
MPI-I-2007-4-001	J. Gall, B. Rosenhahn, H. Seidel	Clustered stochastic optimization for object recognition and pose estimation
MPI-I-2007-2-001	A. Podelski, S. Wagner	A method and a tool for automatic veriication of region stability for hybrid systems
MPI-I-2007-1-003	A. Gidenstam, M. Papatriantafilou	LFthreads: a lock-free thread library
MPI-I-2007-1-002	E. Althaus, S. Canzar	A Lagrangian relaxation approach for the multiple sequence alignment problem
MPI-I-2007-1-001	E. Berberich, L. Kettner	Linear-time reordering in a sweep-line algorithm for algebraic curves intersecting in a common point
MPI-I-2006-5-006	G. Kasnec, F.M. Suchanek, G. Weikum	Yago - a core of semantic knowledge
MPI-I-2006-5-005	R. Angelova, S. Siersdorfer	A neighborhood-based approach for clustering of linked document collections
MPI-I-2006-5-004	F. Suchanek, G. Ifrim, G. Weikum	Combining linguistic and statistical analysis to extract relations from web documents
MPI-I-2006-5-003	V. Scholz, M. Magnor	Garment texture editing in monocular video sequences based on color-coded printing patterns
MPI-I-2006-5-002	H. Bast, D. Majumdar, R. Schenkel, M. Theobald, G. Weikum	IO-Top-k: index-access optimized top-k query processing
MPI-I-2006-5-001	M. Bender, S. Michel, G. Weikum, P. Triantafilou	Overlap-aware global df estimation in distributed information retrieval systems
MPI-I-2006-4-010	A. Belyaev, T. Langer, H. Seidel	Mean value coordinates for arbitrary spherical polygons and polyhedra in $\mathbb{R}^3$
MPI-I-2006-4-009	J. Gall, J. Potthoff, B. Rosenhahn, C. Schnoerr, H. Seidel	Interacting and annealing particle filters: mathematics and a recipe for applications
MPI-I-2006-4-008	I. Albrecht, M. Kipp, M. Neff, H. Seidel	Gesture modeling and animation by imitation
MPI-I-2006-4-007	O. Schall, A. Belyaev, H. Seidel	Feature-preserving non-local denoising of static and time-varying range data
MPI-I-2006-4-006	C. Theobalt, N. Ahmed, H. Lensch, M. Magnor, H. Seidel	Enhanced dynamic reflectometry for relightable free-viewpoint video
MPI-I-2006-4-005	A. Belyaev, H. Seidel, S. Yoshizawa	Skeleton-driven laplacian mesh deformations
MPI-I-2006-4-004	V. Havran, R. Herzog, H. Seidel	On fast construction of spatial hierarchies for ray tracing
MPI-I-2006-4-003	E. de Aguiar, R. Zayer, C. Theobalt, M. Magnor, H. Seidel	A framework for natural animation of digitized models
MPI-I-2006-4-002	G. Ziegler, A. Tevs, C. Theobalt, H. Seidel	GPU point list generation through histogram pyramids
MPI-I-2006-4-001	A. Efremov, R. Mantiuk, K. Myszkowski, H. Seidel	Design and evaluation of backward compatible high dynamic range video compression
MPI-I-2006-2-001	T. Wies, V. Kuncak, K. Zee, A. Podelski, M. Rinard	On verifying complex properties using symbolic shape analysis

MPI-I-2006-1-007	H. Bast, I. Weber, C.W. Mortensen	Output-sensitive autocompletion search
MPI-I-2006-1-006	M. Kerber	Division-free computation of subresultants using bezout matrices
MPI-I-2006-1-005	A. Eigenwillig, L. Kettner, N. Wolpert	Snap rounding of Bézier curves
MPI-I-2006-1-004	S. Funke, S. Laue, R. Naujoks, L. Zvi	Power assignment problems in wireless communication
MPI-I-2005-5-002	S. Siersdorfer, G. Weikum	Automated retraining methods for document classification and their parameter tuning
MPI-I-2005-4-006	C. Fuchs, M. Goesele, T. Chen, H. Seidel	An emperical model for heterogeneous translucent objects
MPI-I-2005-4-005	G. Krawczyk, M. Goesele, H. Seidel	Photometric calibration of high dynamic range cameras
MPI-I-2005-4-004	C. Theobalt, N. Ahmed, E. De Aguiar, G. Ziegler, H. Lensch, M.A. Magnor, H. Seidel	Joint motion and reflectance capture for creating relightable 3D videos
MPI-I-2005-4-003	T. Langer, A.G. Belyaev, H. Seidel	Analysis and design of discrete normals and curvatures
MPI-I-2005-4-002	O. Schall, A. Belyaev, H. Seidel	Sparse meshing of uncertain and noisy surface scattered data