Data-Driven Video Completion

S. Ilan[1] and A. Shamir[2]

[1]Tel-Aviv University, Israel
[2]The Interdisciplinary Center, Israel

Abstract

Image completion techniques aim to complete selected regions of an image in a natural looking manner with little or no user interaction. Video Completion, the space-time equivalent of the image completion problem, inherits and extends both the difficulties and the solutions of the original 2D problem, but also imposes new ones - mainly temporal coherency and space complexity (videos contain significantly more information than images). Data-driven approaches to completion have been established as a favored choice, especially when large regions have to be filled. In this report we present the current state-of-the-art in data-driven video completion techniques. For unacquainted researchers, we aim to provide a broad yet easy to follow introduction to the subject and early guidance to the challenges ahead. For a versed reader, we offer a comprehensive review of the contemporary techniques, sectioned out by their approaches to key aspects of the problem.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—I.4.4 [Image Processing and Computer Vision]: Restoration—I.4.9 [Image Processing and Computer Vision]: Applications—

1. Introduction

Digital image restoration and manipulation have been performed by artists using basic editing tools since the dawn of digital imaging. One of the most useful operations on images is called *image completion* or inpainting. This operation fills regions of an image that have been marked as missing or invalid. The objective of image completion is to achieve a seamless and natural looking fill of the unknown region, usually referred to as a "hole". In some cases, it may be desired that the filled content be as close as possible to some ground truth which has been damaged or concealed; in other cases, the goal is to conceal some part of the original image. In both cases, the conducted alteration is desired to be unnoticeable by an unsuspecting viewer. With modern completion algorithms, in many cases these fixes can be done completely automatically (after the damaged area has been marked).

Similar to image completion, *video completion* aims to fill missing or removed regions in a video in a seamless manner. Video completion poses more challenges than image completion both because of the larger size of videos and because the missing part can extend to more than one video frame and demands temporal coherency as well as spatial coherency. The main purpose of this state-of-the-art report is to map the key challenging aspects in video completion methods and to review the existing body of work by examining these challenges.

To fill all but the smallest of hole regions in an image, it is necessary to synthesize content for the missing areas. A common approach is to use data from different image regions called "exemplars". These could be taken from known regions of the same image, or from other images. The information around the hole must be taken into account when searching for exemplars used to complete the hole if a seamless result is to be achieved. Other considerations include the preservation of geometrical structures and texture inside the completed area. The methods used to search for good exemplars and the methods used to fuse the information from these exemplars to fill the hole, comprise the key differences between different works in this field.

While the completion of video is similar in many senses to image completion, the solutions for image completion cannot be directly applied to video completion. In video, there is much more data available for completion; exemplars for completion can be drawn from different spatial and temporal regions outside of the hole. However, the solution also becomes more complex. Significantly more data may be missing, and coherency must be kept both spatially (in ev-

DOI: 10.2312/egst.20141038

ery frame individually) and temporally (among consecutive frames) for the video to appear natural. In all but the simplest cases, applying successful image completion algorithms to video by simply performing the completion on each frame separately will result in temporal incoherencies.

In addition to its usefulness as a general video editing tool, the need for video completion naturally arises in a large variety of real-life applications. These include: restoration of damaged vintage films through the completion of damaged areas and line scratches ([JBBB99, THS*11]); post-production object removal for the television and film industries - this need often arises in cases such as public sets where unwanted beings or objects can accidentally enter the set during shooting or in cases when technical personnel or equipment are necessary in the scene for directorial needs ([KCR05]); removing logos or watermarks from videos in broadcasting ([YWK05]). Video completion is also a useful tool for other applications which inherently leave behind unknown areas after performing various types of video processing tasks on videos. Digital video stabilization is one such example; after stabilization, the outskirts of the video exhibit unknown regions which must be filled or cropped ([MOG*06, WSI07]).

In contrast to image completion methods that can already operate successfully on many types of inputs, video completion methods are far less mature. Most of the tasks involving video completion are performed manually using video editing software. Hence, such tasks may take hours or days to complete. Therefore, even a semi-automatic tool that would relieve some of the manual labor can be of great benefit to video-editing professionals. The popularity of image completion tools in photo-editing software can provide an indication to the usefulness of good video completion tools.

1.1. Overview

We have used two main aspects to classify and describe the video completion literature.

Exemplar granularity and shape. The granularity of completion determines how much information is completed in an atomic pass (iteration) of the method. Granularity can range from single pixel inpainting (where each pixel can be received from a different source), through inpainting of small or medium size patches and up to completion of larger segments and even entire objects. The shape of exemplars can be regular as in pixels or block-patches, or irregular as in segments with irregular boundaries, or whole objects.

Local vs. global techniques. Most data-driven methods perform completion by optimizing for the best exemplar sources either locally or globally. Local (or greedy) methods include methods which incrementally complete missing regions in space-time; they choose a region for completion and then find a best candidate match for that region, considering only how good source candidates match the known part of the target region at that point in the completion process. Global methods work by optimally choosing many inter-overlapping source patches together (or at least in consideration of one another), usually in an effort to minimize some global energy functional. In these methods, the decision which source exemplar is adequate to complete a certain area takes into account more information than only the known pixels which a copied source will overlap, optimizing for spatial or spatio-temporal coherency of a larger area at once. The optimization functions for the second type of methods are typically far more complex but such solutions are less prone to concentrated error regions which are prevalent among greedy algorithms.

We organize this review as follows. First, we provide some background regarding image inpainting and other related fields in Section 2. We discuss various constraints of the video inpainting problem in Section 3 and then follow our classification of the various works by the exemplar type in Sections 4, 5 and 6, where approaches working in patch, segment and object granularities are presented. Section 4 is more lengthy and partitioned as the majority of methods use patch granularity. Section 7 describes warp-driven completion methods. Section 8 discusses the problem of assessing completion outputs; Section 9 identifies some of the current challenges in the field and offers a concluding discussion.

2. Background

There are several image processing operations that are crucial to the understanding of many works in the field of video completion. We provide an overview of some here, with an emphasis on texture synthesis and image completion methods as many video completion methods are based on earlier works in these fields. We also briefly discuss common terminology and notations.

2.1. Terminology

Early on, the term *inpainting* was used to describe the act of filling and repairing scratches and other small defects in images and the term *completion* to denote the filling of larger holes. This distinction, however, was not rigorously kept and the terms are now used interchangeably. Other terms such as image or video interpolation, error concealment, repairing and falsifying are also sometimes used to describe the problem in different contexts. The missing region of the image is many times termed the *hole* or *target* region. A binary mask video (usually given as an input to the algorithm) is used to indicate the pixels included in the missing region. Figure 1 illustrates the main notations of [CPT04] which have been adopted in many image and video completion texts:

- I - The image to be completed
- Ω - The hole region to be filled

- $\delta\Omega$ - The boundary of the hole region
- Φ - The source area, the known part of the image or video used to obtain exemplars for completion. Typically $\Phi = I - \Omega$
- Ψ_p - The target patch, centered at pixel p which is to be completed
- Ψ_q - A candidate source patch, centered at pixel q which is to be used for completion

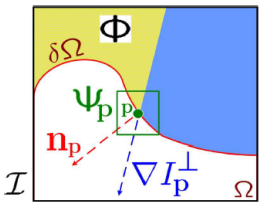


Figure 1: Visualization of the notations of [CPT04].

Variational image inpainting. In variational image inpainting, the focus of the completion process is on the correct continuation of geometrical features into the hole region. These algorithms usually perform diffusion of the data surrounding the hole, rather than use exemplars for completion. Because of this, they are usually only successful in filling relatively narrow hole regions. In [MM98], small missing areas in an image are filled by recovering image level lines (boundaries of image areas where the intensity is greater than a certain gray-scale value). Bertalmio et al. [BSCB00] studied how professionals perform manual inpainting. They followed these guidelines in their automatic algorithm: preservation of structure, continuation of contours that end on edge boundary and replication of correct color inside the inpainted area. They used an iterative diffusion process to gradually propagate isophotes (contours of equal luminance in images) into the hole region. In [BBS01] the authors use a method called Navier-Stokes, from the field of computational fluid dynamics, to perform image inpainting by describing the inpainting problem directly as a flow problem. A comprehensive survey on variational methods for inpainting is given in [CS05].

Example-based texture synthesis. Example-based texture synthesis algorithms are designed to automatically create arbitrarily large output textures given a small example of a texture. The output is required to have the same statistics as the input and avoid visual artifacts or perceivable repetitions. Many of the works in this field chose to model the texture statistics as a Markov Random Field (MRF). [PP93] suggest a cluster-based kernel-estimation approach to model high-dimensional vector statistics and implement them for texture synthesis, classification and compression. [PL95, PL98] advocated the use of a non-causal neighborhood (unlike the one used in [PP93]) and a top-down approach where frequency components of the image are gradually introduced from low to high frequency.

Efros and Leung [EL99] chose to approximate an MRF generative process using non-parametric sampling. The process is initiated from a small example of texture (the input), and pixels are gradually grown around it from all sides. For each pixel to be synthesized, its neighborhood was considered as a square window (a target patch) around it; the patches were weighted by a 2D Gaussian for emphasizing importance according to the distance from the center pixel which was to be completed. The algorithm simply searched for patches similar to the known part of the neighborhood in the example image and formed the probability distribution for the missing pixel according to the intensities found in all the matching candidate source patches. The patch based non-parametric sampling approach of [EL99] had later become quite popular in the image and video completion communities.

The method of [EL99] was successful but rather slow. To improve the running time, [WL00] suggested using a scan-line directed fill and a constant size and causal-shaped neighborhood (i.e. all pixels in the neighborhood are known at completion time). This enabled to accelerate the search using tree-structured vector quantization. In a later work [EF01], Efros and Freeman performed synthesis using overlapping blocks of texture (rather than one pixel at a time). The blocks were chosen randomly from the input texture while avoiding blocks with large error in the overlap region. The special treatment given to the overlap region is discussed in Section 4.6.

While texture synthesis algorithms were successful in generating relatively regular textures, many failed to preserve complex geometrical structures. [WLKT09] gives a good survey of texture synthesis methods.

Exemplar-based image completion. Inspired by [EL99], Drori et al. [DCOY03] proposed one of the first example based image completion frameworks. In a fast approximation method, smooth color completion is achieved by down sampling and up sampling the image repeatedly. This smooth completion acts as the base for patch comparisons for an example-based completion method.

Also inspired by [EL99], Criminisi et al. [CPT04] offered a greedy approach in which the emphasis was put on the order of inpainting (priority of patches). The priority is determined by two terms: the confidence term, which signifies how much of a candidate's patch is known and what is the confidence in the knowledge (confidence decreases as completion progresses deeper into the hole); the data term, which prefers significant gradient energy, with isophotes directed explicitly into the hole. Although the method gives good results on many inputs, it has some drawbacks. A few incorrectly selected patches will encourage later incorrect patches that match them, curved structures are hard to handle and visible structure repetitions can occur. Nevertheless, the simple and effective formulation of [CPT04], compelled many researchers to follows in its footsteps.

The high importance of accurately preserving structure and the difficulty to do so automatically encouraged the use of user-assisted methods. In [SYJS05], the user is first asked to sketch object boundaries continuing into the hole, thereby giving the algorithm a strong structure prior. Patches are synthesized over the curves drawn by the user and dynamic programming is used to find the optimal combination of patches over the entire curve.

Many more exemplar-based image completion methods have been developed in the past decade. Some of these methods pertain to specific video completion methods or to aspects of the problem and are presented briefly later in the text.

3. Problem constraints

Most current video completion solutions are far from general and need to make some assumptions on their inputs. In this section we describe some of the popular assumptions and the rational behind them.

Object movement characteristics. Accounting for arbitrary movement of objects in a scene could easily turn video completion into an ill-posed problem. Therefore, hard assumptions on the types of movements are often imposed on the input. If objects only move parallel to the camera plane and the scene is not very perspective, the motion can be approximated as pure 2D translation. In more complex situations, even rigid objects which exhibit rotations, scales and perspective transforms can turn out to be hard to complete. Looking for matching patches that had undergone these transformations increases the dimensionality of the search space. Periodic and patterned movements are easier to expect and extrapolate than general ones, and are therefore also assumed in many cases (e.g. [JTPTT04, BSHK04, JTWT06]).

Background movement. Many works in the field (e.g. [PSB05, CZV06, PSB07]) assume that the background over which objects are observed is completely static and only a few objects of interest (usually termed foreground objects) are dynamic. This allows easy separation of background and foreground which in turn makes the completion process simpler.

Camera movement. Many times, assumptions are made on the movement of the camera. Simple movements such as a static camera (e.g. [CZV06]) or a camera which only exhibits pan motions parallel to scene (e.g. [PSB07]) are often assumed. More complex camera movements require special handling by the algorithm. Consequently, only few works deal with such motions (e.g. [GKT*12]).

4. Pixel and patch-based completion

Patch based completion methods are by far the most common data-driven techniques for completing images and videos. A few general components can be recognized in most patch-based approaches: the definition of a distance measure between patches, a method to search for patches, and a method to combine the patches to fill the hole. Therefore, after reviewing local and global approaches, we discuss various methods for patch comparison, search and stitching.

4.1. Local patch-based methods

The family of local patch-based methods finds its roots directly in patch-based texture synthesis and image completion literature (e.g. [EL99, EF01, CPT04]).

Patwardhan et al. [PSB05] demonstrated a greedy patch-based inpainting approach. Inspired by [CPT04], they use a priority based method for completion order where the priorities also take into account information available in other frames and motion information. A static background and a static camera are assumed so a foreground-background separation can be achieved using simple optical flow. Under these assumptions, much of the background can be filled using information from pure temporal offsets (the information is simply copied from nearby frames where the same region is unclouded). For regions where data is missing throughout the entire video, the authors follow [CPT04] to perform stationary background inpainting. Dynamic foreground inpainting is achieved one frame at a time; only the moving parts in the video are searched for matching patches (patches are searched in spatio-temporal offsets) and only the foreground (moving) pixels of a source patch are copied back to the target patch. To preserve temporal continuity, motion channels are compared along with color values during the source candidate search. This method is relatively fast (takes minutes to complete a typical video) and demonstrated plausible results. However, only cases where the motion is very cyclic and parallel to camera plane are demonstrated.

This method was later extended in [PSB07] for constrained camera motions through the use of mosaics (see Section 7.1). Recently, another extension of the original work was published in [EGLM12]. Here the K-nearest neighbors of a patch are used to jointly estimate an optimal linear combination fill patch using neighbor embedding techniques.

4.2. Global patch-based methods

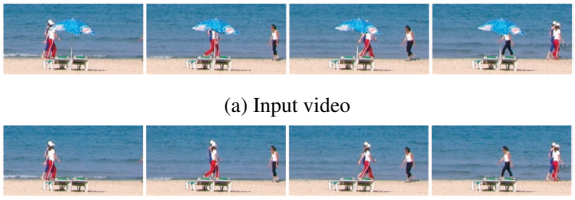
Global patch-based solutions to video completion avoid local greedy steps which may lead to unrecoverable errors in the output. Wexler et al. [WSI04, WSI07] presented a pioneering work, which used 3D spatio-temporal patches (containing information from several frames) instead of 2D patches in order to better preserve temporal coherency. A global coherence function is defined for the completion output. In order to maximize this function, each pixel must be in agreement with all patches that contain it and those patches must be similar to patches in the data-set (the known part of

the video). The algorithm works in an iterative fashion; in each iteration a nearest neighbor field (NNF) is first initialized; this field includes the closest matching patch for every target patch in the hole. The NNF's computation is accelerated using a search structure. Then, each pixel in the unknown region is initialized to the value most agreed upon by all the patches in its spatio-temporal neighborhood; this is done by applying mean-shift segmentation to the different values offered by all of its containing patches, and taking a weighted (by patch similarity) average of the values in the highest mode. Intuitively, if the chosen neighborhood size is $5 \times 5 \times 5$ then 125 cubic source patches can affect the values chosen for a pixel. To prevent over-smoothing, only patches that have a value close to the majority value will influence the result through the weighted average. The entire process is performed using a pyramid, starting at a very low resolution, initialized with a smooth completion. Using the solution of a coarser level to initialize the optimization of the level above it gives each finer resolution level a good starting point, making the convergence tractable.

Another contribution of [WSI07], was the use of optical-flow-like channels together with color channels for patch comparisons, in order to account for motion continuity. Spatial and temporal derivatives (Y_x, Y_y, Y_t) are computed for each pixel. $u = Y_t/Y_x$ and $v = Y_t/Y_y$ are used to capture the horizontal and vertical motion components, respectively. The two local motion components (multiplied by a weighting factor) are added to the L_2-norm patch comparison, along with the RGB color channels.

Figure 2 shows an example result of [WSI07]. The method was also used for video stabilization and image completion. Due to its global nature, the computation times are long (typically hours) and only completion of low-resolution videos was demonstrated. The vast majority of time is required to compute the NNF. In [BSFG09], the method of [WSI07] is used together with the introduced PatchMatch algorithm in order to achieve quality image completion at interactive speeds. Very recently, Newson et al. [NFP*13] implemented the video completion method of [WSI07] with a spatio-temporal version of the PatchMatch algorithm and reported a significant speedup in patch search times. Their article also includes a few important implementation details, which are missing in [WSI07].

Another example of a global completion algorithm was presented by Cheung et al. [CFJ05], where video epitomes are learned from an input video and then used to perform a variety of tasks, including video completion. Epitomes are patch-based probability models in the form of small, efficient representations of larger data which still preserve a good representation of the statistical properties of the original data. In the work, Cheung et al. describe the process of unsupervised learning of a video epitome from an input video through the use of sampled spatio-temporal patches of various sizes (see Figure 3). A generative process uses epitome mapping probabilities to generate a random but consistent quilted result from different epitome patches in the missing area of the video. The iterative process encourages mappings which are agreed upon by many overlapping epitome patches.



(a) Input video

(b) Video with beach parasol removed and people completed

Figure 2: Frames from a video result of [WSI07]

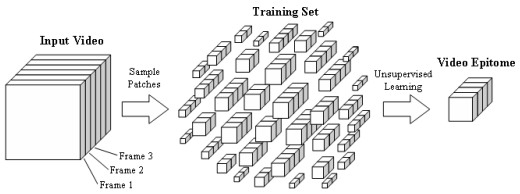


Figure 3: The process of learning a video epitome from an input video in [CFJ05].

4.3. Size of patches

The selection of patch sizes was given a lot of attention since the inception of patch based-synthesis. In [EL99] the patch size is left as a user-selected parameter for texture synthesis, and it is indicated that it should be chosen at the scale of the largest regular feature in the texture. It is also shown how small neighborhoods cause the output texture to be more random up to the point of structure corruption (see Figure 4). On the other hand, large neighborhoods can cause over regularity in the output and sometimes artifacts due to strong unsolvable constraints they may impose.

[WL00] had suggested a more automatic approach for capturing large-sized structures. Instead of using a single user-determined size, they use a fixed size neighborhood, applied in a multi-resolution fashion. Large structures are captured in the lower levels of the multi-resolution pyramid whereas smaller ones are captured later when the resolution grows. Besides avoiding the need for the user to select this parameter, this also speeds up the process, bypassing the need to compare a large neighborhood to capture large structures. To preserve the structures captured in the lower resolutions, the patch comparison in higher resolutions consists of a concatenated vector of higher and lower resolution patches. In image completion, Drori et al. [DCOY03] used a locally adaptive neighborhood size to capture structures at

various scales. The adaptive size was approximated by testing the absolute difference between extreme values within a patch in a few discrete neighborhood sizes.

In video completion, because of the higher amount of visual information, it is harder for the user to approximate the largest structure sizes as proposed in [EL99] and only few works prefer this solution. Methods using 3D patches are naturally more limited in the spatial size of the patches; for example, [WSI04, WSI07] (described earlier), use a multi-resolution approach with small $5 \times 5 \times 5$ cubic neighborhoods and complete one pixel at a time.

The non-global methods sometimes perform completion with relatively larger patches (e.g. [BLLC02]). In [BSHK04], very large (24×24) texture patches are used, but feathering is applied to blend the patches so the effective size is smaller. Many methods compare larger neighborhoods than are inpainted in the local step; [STH09] use 7×7 patches (termed patch templates) for comparison but only transfer the middle 3×3 values. Transferring of large patches during completion necessitates the use of blending or stitching techniques to avoid artifacts (e.g. [JHM05]). This is discussed in Section 4.6.

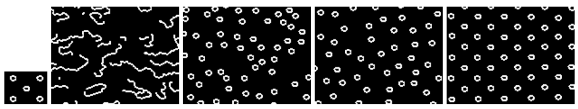


Figure 4: A sample input and outputs of synthesis from [EL99], generated with growing patch neighborhood sizes. The largest achieves the most ordered pattern.

4.4. Patch comparison

Most patch-based methods choose very simple means for comparing patches such as using a simple Sum of Squared Distances (SSD) of color values. LAB color space is sometimes preferred over RGB (e.g. [CPT04, STY*06, STH09]) because of its perceptual uniformity. Some works (e.g. [PSB05, PSB07]) follow [WSI04] (see Section 4.2) and perform comparisons using motion as well as color information.

Kumar et al. [KBBN05] manage to circumvent the exhaustive search and comparison of masked patches through a careful decomposition and treatment of L_2 norm components. They follow [CPT04] to perform image and video inpainting and extend the 2D priority scheme of Criminisi et al. into 3D video volume patches. The separate terms of the decomposed L_2 norm are calculated quickly using FFT after describing them as convolution products. A similar acceleration method for block shaped patches was previously proposed by [KDM02] and Kumar et al. extend this method to treat binary masked patches. This type of treatment is relatively unseen, since most search acceleration methods are not usable on masked or weighted patches as explained next.

Masks and weights. To avoid comparing unknown areas, it is common to use the mask directly in the comparison and compare only $\Psi_p \cap \Phi$ (the known part of the target patch) to the source. If the foreground is separated from the background, masked comparisons also allow inpainting of only one layer (foreground or background) at a time (e.g. see [PSB05]), by eliminating the background parts in foreground source patches. Some methods use weights on pixels inside the neighborhood (e.g. [BLLC02, DCOY03]), where pixels are weighted according to the confidence in their value or their distance from the center of the neighborhood.

The limitation of using weights (binary or scalar) is that they prevent the use of many acceleration methods. Comparing the two early texture synthesis works of Efros and Leung [EL99] and Wei and Levoy [WL00] illustrates this point. Efros and Leung proceeded in a less constraining fashion by comparing partially known patches, using masks and a Gaussian weighted comparison. The algorithm they present is intuitive but slow because of the need to exhaustively search for sources for each patch and compare them with a weighted kernel. On the other hand, Wei and Levoy avoid the use of weighted kernels and propose a method where all pixels in the neighborhood are valid for comparison by assigning random values to the output texture first. The tree-structured vector quantization technique Wei and Levoy used for acceleration, could not have been used if weighted patch comparisons were performed. Wei and Levoy also use a multi-resolution pyramid scheme to accelerate their algorithm; while this could also be done with masked comparisons, it is more problematic because masks need to be down sampled conservatively to avoid problems at mask edges; in practice this is rarely done.

Avoiding weighted comparisons not only speeds up the basic patch comparison task, but also allows the use of other acceleration methods such as dimensionality reduction (e.g. with PCA, see [LH05]) and fast Nearest Neighbor methods which can only handle non-weighted patch comparisons.

4.5. Searching methods

Searching for source patches usually turns out to be one of the most computationally expensive tasks in completion algorithm pipelines. To deal with this, many works use search structures or perform the search for patches on a subset of the original space or on a different space, smaller than the entire original video. For example, some earlier works (e.g. [BLLC02, STY*06]) perform the search for source patches only in the close spatio-temporal neighborhood of the target location. Besides speeding up the search, many search space reduction techniques also help in improving the quality of the output. By culling out areas in which the legitimate source patches could not reside, the probability for an error in source patch selection is reduced. Search structures for fast nearest neighbor queries on patches have been used in

several works. For example, [WSI07] employed the approximate nearest neighbor method of Arya et al. [AM93].

Layer separation. Different objects or regions in a scene often exhibit motion patterns which are coherent for a single object, but different from other objects; this is due to relative object motion or parallax (in videos where the camera is moving). A segmentation of these distinct regions forms disjoint layers of video which together compose the entire video. The layers exhibit a natural (partial) ordering based on the perceived depth occlusions between layers. A layer segmentation based on motion or depth information can prove useful for patch searching. If a patch is missing from a certain layer which represents an object or region in the video, it is logical to restrict the search for matching sources only to that layer.

Many works perform simple motion segmentation into foreground and background layers, relying on the assumption that the background of the scene is stationary [PSB05, PSB07, SLCF06]. In [XGY*11], the user manually defines a foreground and background separation in a single frame and Gaussian Mixture models are generated for different regions of the foreground and background. The models are later used for differentiating foreground and background and for limiting the search space to only the most relevant GMM region for each target patch. In [STH09], Shih et al. use a block motion estimation algorithm from video encoding ([PM96]) in order to achieve a multi-layer motion segmentation of the video. Zhang et al. [ZXS05] performed layer separation of the different moving objects in the scene using a graph-cut based method, which also estimated the layer depth order.

Motion-guided search. Motion data can be used to aid and guide the search for source patches. If the motion vectors in a video are known in advance or calculated in a preprocessing stage, the search for patches can be guided by this motion field. Such an approach is adopted in [GBBM09], searching only in past locations which contain data with movement vectors directed into the target patch, and future locations containing data with movement vectors originating from the target patch. In [EGLM12], the locations indicated by backward and forward motion vectors are used as the centers of patch search windows. Jia et al. [JHM05] use mean shift tracking to track moving features in the image. If a target selected for completion is part of a trackable element in the video, the search area for the source fragment is restricted to the tracking windows defining the space-time route of the tracked element. Bhat et al. [BSHK04] synthesize videos of particle-flow-like phenomena by first asking the user to draw an estimated flow path over the input video. The algorithm then uses this input together with constrained optical flow to track moving particles along the flow lines; the particles are represented as texture patches.

In stereo content, disparity maps can guide the search for patches in the other view in a similar way to motion vector maps. Candidate patches can be searched in the other view at the offset location given by the stereo disparity of known pixels in the target patch. This technique is used in [RK11] in order to resolve occlusions in a stereo frame.

Coherence based search. Image and video data of real life scenes usually exhibits strong visual coherency both in spatial and temporal dimensions. Such coherency is naturally found in the source information of completion algorithms and is also desired in a natural looking result. This fact is utilized by exemplar-based synthesis algorithms that copy many pixels of a patch at once (e.g. [EF01]), aiming to transfer coherent parts of the input directly to the output. This principal can also help find adequate source patches quickly; if Ψ_p is a source patch matching a target patch Ψ_q, then it is likely that neighboring patches of Ψ_p may be good candidates for completion of neighboring patches of Ψ_q.

This coherence principal was used in [Ash01] for texture synthesis at interactive rates. Given the offset of the previously completed target patch and its source, the coherence search simply picks the patch with same offset to the current target as a candidate. Bornard et al. [BLLC02] used the same principal to perform image and video inpainting. If the coherence based search failed, they turned to a regular search in a spatially restricted area around the the target. Zhang et al. [ZXS05] use an approach which is less explicit regarding relative location and simply search around the neighborhood of previous source patches if their corresponding target patches are in proximity to the currently completed patch.

Recently, a fast approximate nearest neighbor method for image patches, called PatchMatch [BSFG09], has gained much popularity because of its simplicity and effectiveness. The method defines a fast nearest neighbor patch search by exploiting image coherency and the law of large numbers. The algorithm initializes a random nearest neighbor field (NNF) and then alternates between propagation iterations which exploit image coherency to propagate good matches to neighboring patches and iterations which randomly search for better matches. The method relies on the coherency propagation of the few randomly found improvements for the quick convergence of the process. Barnes et al. demonstrated the effectiveness of the method in several applications including image completion, retargeting and reshuffling.

Searching a lower dimensional space. A few methods for video completion refrain from searching the dense 3D space of video pixels altogether and perform searches only in lower dimensional spaces. Some early methods (e.g. [STY*06]) simply chose to search in space or time offsets, but not both together. Other methods found less lenient ways to reduce the dimensionality of the search space.

In [PSB07], a preprocessed video mosaic is used in order to search for patches of the moving foreground layer in the two dimensional space of the mosaic rather than the entire foreground layer of the video (see Figure 5). The target patch

is mapped into the mosaic and the patch in the corresponding mosaic location is used to search for candidate sources in the undamaged part of the mosaic. For each of the candidates found in the mosaic, a full comparison is performed using the original target patch and the source patch from the frame which mapped to the relevant location in the mosaic.

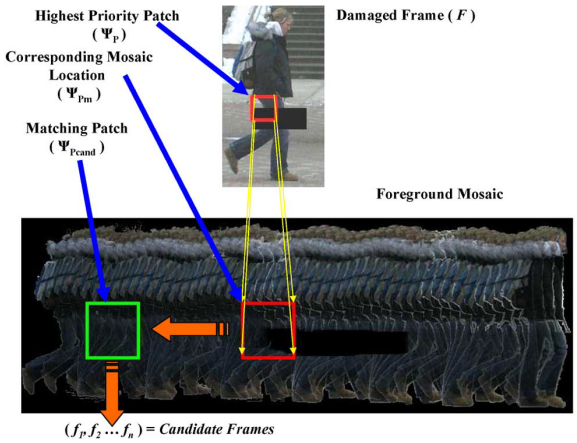


Figure 5: The use of foreground mosaics for matching source patches in [PSB07]. The target patch is mapped into the mosaic and a candidate search is done. Found candidate patches are then mapped back and original patches are compared.

The method of [SLCF06] performs completion on XT video volume slices one at a time. To this end, the video volume is initially rectified so that object movement is made mostly parallel to the X plane throughout the volume. In each XT slice, curve fitting is used to create curves following the movement trajectory. The curves are used as guidance paths for a completion method similar to the one presented in [SYJS05]. The search for patches is done along these curves so the search space is further reduced. In order to preserve spatial coherency in the Y direction, first and second derivatives are considered during the optimization process (a dynamic programming approach).

4.6. Patch stitching

To achieve a good completion result for image or video, the completion process must control the output at the pixel level. While many methods choose to complete one pixel at a time, other methods choose to complete larger patches and therefore have to selectively decide what part of the source is copied back to the target. One way to control this is through layer-separation, which was discussed earlier. Some works, especially those performing local patch optimization, focus on how different patches are stitched together to achieve a seamless result. Efros et al. [EF01] perform texture synthesis using large square patches. To relieve the patch edge artifacts they perform a dynamic programming optimization in order to find the seam of minimum error between patches where the stitching will be least noticeable.

In video completion, Zhang et al. [ZXS05] formulate an energy term for achieving a least noticeable seam and use a graph cut algorithm to compute it. In [JHM05], a similar approach to patch merging is taken, but the graph-cut operates over volumetric patches rather than 2D ones. This method is also claimed to be faster than pixel-by-pixel completion. When the exemplars used are segments or irregular patches, their shapes are often selected so that their seam areas exhibit minimum overlap error. This will be discussed in more details in Section 5.

4.7. Preserving temporal coherency

One of the most important aspects of video completion is the preservation of temporal coherency. The human eye is very sensitive to motion discontinuities; even when each frame of a video is completed successfully and seamlessly, the played video may look unnatural and exhibit flickering or ghost-like artifacts if temporal coherency is not taken into account.

Many patch based methods simply use 3D (volumetric) patches when performing completion [WSI07, JHM05, KBBN05]. 3D patches inherently capture movement in a few consecutive frames and so good overlaps of 3D patches should already encourage temporal coherency. Nevertheless, some methods take extra measures to ensure temporal continuity. One such example is the addition of local motion channels to the patch comparison ([WSI07] - see Section 4.2). Shiratori et al. [SMTK06] compute for the known part of the video the motion field at a preprocessing stage. They then use 3D patches to inpaint the motion field itself, rather than the color information. The color information is later propagated from areas surrounding the hole using the completed motion field. The resulting motion is therefore inherently smooth but the method can only deal with temporally short holes.

In methods using 2D patch completion, there is usually a need for more explicit temporal coherency preservation. Patwardhan et al. [PSB05, PSB07] perform frame by frame inpainting but compare patches using the same 5-channel metric used by [WSI04]. In [ZXS05], temporal consistency is maintained by warping the completion result in a reference frame to other frames (see Section 7). Shih et al. [STH09] also maintain temporal continuity using motion parameters, but transfer completed content from frame to frame consecutively. Shen et al. [SLCF06] complete XT patches in the video volume rather than XY patches (see description above), thereby trading the temporal coherency issue with a Y-axis spatial coherency issue. In [BGD*10], Kalman filters applied to pixel values along motion trajectories are used to perform smoothing in the temporal domain after frame-by-frame image inpainting has been done.

5. Segment-based completion

Segment-based completion methods make up a small family of methods in which exemplars are segmented from the source with a least noticeable result seam in mind. These methods resemble patch-based methods which use large patches and stitching (see Section 4.6). In contrast, in segment-based completion methods, the exemplar shapes and sizes are determined implicitly and can be arbitrarily large or small. For segment-based methods, graph cut [BVZ01] is usually used to find an optimal seam, which determines the irregular shape of the exemplars.

5.1. Local segment-based methods

Extending the ideas of [EF01] (see Section 2), [KSE*03] present a synthesis framework focusing on finding minimally noticeable seams. By using graph cuts, the limitation of grid-like patch synthesis imposed by the dynamic programming framework of [EF01] is relieved. The algorithm works iteratively, picking an offset for the source input texture (or a sub-patch of it) in the output and then finding the optimal seam between the original and offset using graph cut. The process is repeated even after the entire output area has been completed; in each iteration the source segment is pasted over an area with high seam error in the output in order to reduce it.

Graph-cuts extend naturally to the spatio-temporal domain where seams become 2D surfaces. [KSE*03] demonstrate their algorithm on inputs ranging from very regular textures to real-life images and videos of natural phenomena. On videos, they show how their framework can facilitate looping of sequences, extending short sequences and enlarging small ones. The effects of using temporal vs. spatio-temporal offsets of sources is demonstrated. While temporally stationary textures (such as a waterfall) can expect sufficiently good results with only pure temporal translations, spatio-temporally stationary textures (e.g. small waves) achieve better quality by using a spatio-temporal search for sources.

5.2. Global segment-based methods

In Shift-Map image editing [PKVP09], several image editing tasks (completion, retargeting, re-arrangement and composition) are demonstrated under one unified segment-based framework. The authors define a global multi-label graph-cut based optimization, in which each label corresponds to an offset into a known area in the scene, which will be used as the source for a pixel in the output. Graph-cut optimizes for continuous segments of source offsets and for low-energy seams between adjacent areas with different source offsets. The smoothness component of the graph cut energy term penalizes color and gradient discontinuities across the boundary and the data term is used to constrain the solution according to the specific application. In image completion the data term is simply set to infinity for pixels in the hole and to zero elsewhere to make sure that no hole pixels are chosen in the synthesis. The solution can be slow so a hierarchical solution approach is necessary to speed up the process. In each level of the hierarchy, the solution of the coarser level is used as an initial guess to the optimization of the level above it. A completion result of the algorithm is displayed in Figure 6.

(a) Input (b) Output (c) x coordinate shift map (d) y coordinate shift map

Figure 6: Completion result of [PKVP09] along with shift regions of x and y coordinates.

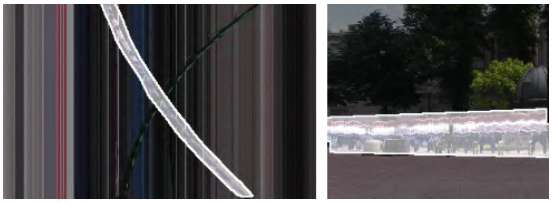


Figure 7: User assistance is employed in [GTK*12] in order to reduce the search space. The user marks the occluded object along its entire path in an XT slice of the video volume (left) and then in YT sections (right) and thereby restricts the source region of possible shifts to a region around the occluded object.

Granados et al. extended the completion method of [PKVP09] to video in [GTK*12]. Segments become volumetric and shifts are extended to space-time offsets in the video volume. The optimization functional is weighed to increase cost of temporal discontinuities over spatial ones. In addition, to avoid a degenerate solution where a single source offset is used for the entire hole and all the error is concentrated on the boundary, pixel pairs closer to the boundary are penalized stronger than ones further inside the hole. User assistance helps reduce the search space by indicating approximate foreground object extents quickly in XT and YT cross-sections of the 3D video cube (see Figure 7). Like in [PKVP09], a pyramid scheme is used to accelerate the solution. Relatively high-resolution outputs are demonstrated but their computation times are high (some reach an order of a few days).

6. Object based methods

Object based completion methods are developed around the notion of objects in the video. The focus in this family of works therefore lies in the correct calculation of continuous object motion. Some of the works focusing on object completion perform completion of a single object using sources which contain a complete copy the object (sometimes termed *template*) and optimize for good temporal continuity between these sources. This sometimes proves somewhat less flexible than working with smaller patches and can result in temporal artifacts around hole edges. Other object based methods attempt to improve motion estimation and synthesis by using more specialized models that are geared towards human or skeletal motion.

6.1. Template-based methods

The method of [CZV06] commences by tracking and segmenting humans (represented as moving blobs) using simple background subtraction (stationary camera and static background are assumed). The segmentation process also separates multiple objects from one another and stores object templates - short snips of object movement, into a database for later use. Missing static background areas are inpainted using information from temporal offsets. In places where such information is not available, the method of [CPT04] is used for completion. The dynamic missing parts are completed using the stored templates, by overlapping them temporally. This is done using dynamic programming which minimizes the overlap cost, thereby preserving temporal coherency. The motion in the output is generally smooth but in some cases inconsistencies can be observed around the temporal edges of the hole. The removal of shadows in the segmentation stage is also noticeable in the result.

The authors improve the work in [VcSCZ09] by treating the frames where objects are partially occluded separately and matching object templates to the partial objects before applying the dynamic programming optimization. The static camera assumption is also relaxed and parallel-to-scene motions are allowed.

6.2. Explicit human modeling

In [LLS*11], Ling et al. propose a contour-based method for occluded object completion. Their method proceeds as follows: first, the moving object in the video is analyzed and postures of the object are extracted for later use. Postures go through a key-posture selection phase where representative postures are selected, saved and labeled. In the next step, virtual contours are constructed. These will later be used to guide the posture-based completion. To facilitate their construction, the XT spatio-temporal slices of the video are separately completed (using [CPT04]) after warping the frames to make the motion purely horizontal. This is reminiscent in nature to the processing done in [SLCF06], discussed earlier. The completed 2D slices are then combined to form the virtual contours for the missing regions. Using local shape matching descriptors, postures from the database created earlier are matched to the virtual contours. Postures are represented as strings and string matching is used to match database postures to postures detected using virtual contours. If matching fails, the algorithm can synthesize new posture sequences by combining postures from the database using a skeletal analysis process.

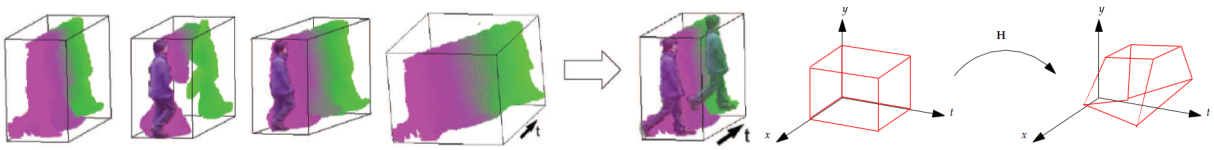
The work of Shih et al. [STTZ08], focuses on layer-based re-composition of videos. Motion interpolation based on patch-driven synthesis is used to alter the speed and location of motion layers extracted in preprocessing. The authors extract stick figures from the moving object layers. Stick figures in missing areas are then matched from other examples in the video or interpolated from other stick figures. The completed stick figures guide the motion interpolation by copying source patches using the prior knowledge inherent to the model (i.e. source patches are taken from correct relative location on stick figure). The initial patches completed using the stick figure knowledge guide a patch-based completion algorithm for the rest of the missing pixels. Edited and interpolated video motion layers are fused seamlessly using graph-cuts.

In [WLL07], human motions are estimated using simple feature points marked manually on the human body. Motion state vectors are computed for undamaged frames according to these descriptors and motion state prediction is done according to a simple cyclic motion model. Graph cut is used to merge the undamaged frames corresponding to the predicted motion states.

7. Warp-driven completion methods

In many cases, completion is desired for videos with a moving camera. In such cases, transformations which consider the camera motion are sometimes required to correctly transfer background source information in one frame to target information in another. Inter-frame warping or background mosaicing can be effectively used in order to overcome changes in camera position, rotation and scale between source and target. In addition, warping can be an effective tool for transferring patch data when the objects themselves had undergone perspective, scale or other transformations between source and target frames.

In the work of Jia et al. [JTPTT04], two separate strategies are employed in order to perform video completion of foreground and static background. Some camera movements are allowed in the scene: zooming, rotation about a point and panning. To facilitate this, the background is modeled using several layers (achieved through coarse segmentation by the user and mean shift tracking). Background completion starts by constructing mosaics for each of the layers in a pre-chosen reference frame. Homographies to the reference



(a) Sampled movel (b) Damaged movel (c) Repaired movel (d) Movel is wrapped and normalized (e) Movel volumes are aligned using a 4×4 homography

Figure 8: Movel sampling and alignment in [JTPTT04].

frame are then calculated for the different layers in other frames and blended at the seam areas between layers. Holes left in the mosaics are filled using image repairing ([JT03]) and the completed layer mosaics can be projected back to any video frame using the homographies.

The moving foreground completion assumes cyclic motions and spatio-temporal object samples termed *movels*, are sampled along a single cycle of motion (such as a full step of a walking person). Damaged movels (ones that include holes) are completed using sample movels in a two-phase process which includes sampling and alignment. In the sampling phase, movels go through three steps of preprocessing. Movel wrapping alters the first and last frames of a movel so it can be wrapped (cycled) without temporal flickers. Movel regularization computes a smooth trajectory (of movel frame centroids) for the damaged and sample movels. In damaged movels, this step also recovers a smooth trajectory for the missing frames. Movel normalization translates the sample and damaged movel centroids to the image center in each frame in order to simplify the alignment search (see Figure 8). In the alignment phase, sample movels are aligned with damaged movels using a spatio-temporal (4×4) homography. The volume homography is found using a Levenberg-Marquardt optimization, and is used to warp the sample movel to the damaged movel; the latter is then repaired using the former. Finally, repaired frames are de-normalized (offseted back to original frame positions) and matting is used to fuse foreground and background inpainted results together in order to obtain the final video result. Figure 12 displays a frame completed using the method.

Granados et al. [GKT*12], focus on completing the background in videos taken with a free moving camera which exhibits much movement. Their completion process requires specifying two masks: the desired part of the moving foreground and the part to remove. The algorithm proceeds in three general steps. In the first step, frame pairs are aligned to each other. Sparse feature matching is done to establish point pairs and the RANSAC algorithm is used to prune for consistency and obtain the fundamental matrix of the two frames and establish a homography. The outliers from the previous RANSAC step are then used again iteratively to find more homographies with lesser support. A graph cut optimization is done to find the best homography for each pixel region.

Figure 9: Background completion in a frame is composited from different source frames (signified by different colors) using a graph-cut optimization in the method of [GKT*12].

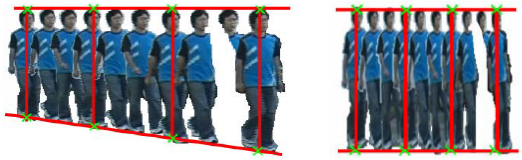
In the second step graph-cuts are used again to composite source regions from different frames for completing the hole region in a target frame; Figure 9 gives a visualization of source frame composition. Distance transform is used to penalize misalignments at the boundary of the hole stronger than inside and avoid a degenerate "copied" solution. In the last step, illumination differences are resolved using Poisson blending [PGB03] and a special regularizer is introduced to eliminate differences in the temporal domain.

In [ZXS05], a motion model is used to find motion of layers between frames. A compensated reference frame (including multiple layers) is created by warping layers in other frames to it, according to their motion parameters. Remaining hidden areas are completed using a patch-based scheme. After completing all of the layers in a single frame, the motion data is used to project the synthesized layers to all other frames according to their motion parameters. This helps preserve motion consistency.

7.1. Mosaics

Mosaics of image sequences ([SS97, Dav98]) are composites of frames captured at different overlapping viewpoints of a scene. The frames are warped and stitched together correctly to display a larger consistent panorama of the scene.

Mosaics are intended to build a complete background of a scene using overlapping images of the scene taken from various viewpoints; this makes mosaics a great tool for the completion of missing background. As long as the missing back-



(a) Foreground mosaic (b) Rectified foreground mosaic

Figure 10: Foreground mosaics in [SLCF06] are used for rectification of the foreground volume. Point sets on head and feet are used to estimate lines parallel to the world's ground plane. Vanishing points are found and used to obtain a rectifying transformation.

ground area can be seen in some other frame of the video, the result mosaic will contain the necessary background information. This approach has been used in several works (e.g. [PSB07, LLS*11]) for separately completing the background layer. The background is usually treated separately from the foreground, and the avoidance of foreground information in mosaicing also helps avoid problems due to large parallax ([SS97]). In [JTPTT04, JTWT06], the background mosaics are built separately for different background layers as explained earlier in this section. In [PSB07] (discussed earlier), foreground and optical flow mosaics are used in addition to background mosaics. In [SLCF06], foreground mosaics are used to guide the rectification of video frames, so that character movement in the video becomes purely horizontal (see Figure 10). In [KS12], a similar foreground mosaic equalization procedure is used in order to achieve scale-robust inpainting. In [LC10], completion of dynamic background is achieved using a linear dynamic system model for dynamic texture analysis and synthesis. In moving camera cases, a background mosaic is built and the formed correspondence maps are used for the training of the model.

8. Output quality assessment

In the case of small hole inpainting, algorithm output quality can sometimes be tested against an original undamaged version of the image; this is rarely the case in exemplar based image and video inpainting where large portions are removed and completed results are not necessarily expected to adhere with a ground truth. This makes the assessment of output quality an inherent problem. Even when comparing results using a single overlap error measure, a smaller error can still result in a less plausible synthesized output. The quality of completion outputs is therefore usually left to the subjective judgment of human beings. Consequently, only a few works in the field went beyond publishing video outputs (sometimes along with comparisons to other methods).

In [HE07], Hays and Efros conducted a user study to assess and compare the performance of their method. In the study, subjects were shown images in one of three categories. Untouched images, images completed using [HE07] and images completed using [CPT04]. The subjects were asked to designate photos as real or fake. Through the user study, Hays and Efros quantifiably showed their method to give superior results compared to [CPT04]. The issue of assessing completion quality is addressed differently in [Mah10], where an experiment using eye-tracking is conducted in order to suggest quality of completion outputs by observing human gaze patterns in the completed region and outside of it. The experiment shows how completion artifacts influence gaze of viewers which tend to concentrate on observable or suspected artifacts.

In [KKDK12], the authors try to predict the quality of image completion in different regions of an image automatically in order to crop low-quality inpainted areas of panorama edges. They run a user study in which they ask to mark completed image blocks as good or bad completion results. Pixels in the blocks are associated with source regions restricted to them and a descriptor which encompasses visual measures of these source regions is calculated for each pixel. Using these descriptors along with the user input from the study, a function which predicts the quality of completion is learned.

9. Discussion

At present time, it seems that state-of-the-art methods for video completion still fall short of becoming practical video-editing tools. Most examples deal only with low resolution, small holes or impose other types of limiting constraints on the input. Further advancements in the field are hence necessary to tackle this hard problem. There are several important challenges to overcome and also some territory which has been left relatively unexplored. We identify some of these challenging and unexplored aspects below.

Solution scalability. Most current video completion algorithms operate on limited resolution, limited length videos. Even if the area to be completed is temporally short, allowing long sequences or even multiple video sources as input could benefit the completion process. Of course, such a growth in the source data would necessitate much more efficient ways to query video data such as intelligent low level descriptors and indexing methods. Working on high-resolution videos is also important for any practical video completion tool. Since methods are struggling with long computation times even for low and medium resolution videos, it would make sense to strive for multi-resolution methods which can show relatively quick low resolution previews, before taking the time to render a completed high resolution output.

High-level models. Most current methods have no means to resolve semantic ambiguities in the scene during the completion process. It seems that at least some of these ambiguities

could be reconciled if more high-level knowledge models were used in the completion process. Until now, only human motion completion seemed to have received such specialized treatment. Knowledge of scene geometry can help properly complete disoccluded regions of a scene correctly. High-level analysis of objects in and around the hole regions might help prevent completion results which include crude semantic discontinuities in space or time. Figure 11 shows an example of such a result.

(a) Original image (b) Frame after scene completion

Figure 11: An otherwise successful and interesting image completion result of [HE07] is hindered by the introduction of only the lower part of a pedestrian into the scene (in circled area).

User-aided methods. Even though only little work has been done towards user-aided algorithms in video, it seems plausible that semi-automatic video completion techniques that rely more on human assistance may help alleviate some of the current challenges in the field. Specifically, it would seam logical to give users motion guidance control by allowing them to specify how structure in non-occluded frames should temporally propagate into a space-time region marked for completion. Also, user interactions such as restriction of source regions for the completion of a certain missing region could be helpful in improving both quality and speed of algorithms. Unfortunately, since current video completion methods do not work at interactive rates, the user's ability to conveniently intervene in the completion process is limited.

Solution speeds. Many successful video completion methods take hours, sometimes even days, to complete a single video sequence of several seconds. Such long computation times render methods unusable for most applications. Even quicker methods working on the order of minutes are problematic as they still cannot allow interactive editing with instant adjustments and reiteration. Automatic image completion methods entered the mainstream only when they became quick enough to perform completions within seconds. The same should be expected for video. To this end, algorithmic advances such as fast patch search algorithms tailored for video are an important goal.

Shadows, reflections and indirect lighting. Problems caused by shadows, reflections and indirect lighting have been avoided in much of the video completion literature. In scenes with a significant amount of direct lighting, shadows of objects cannot be avoided. This poses two challenges; the shadows or reflections of removed objects must also be removed; necessary shadows or reflections in hole areas must be synthesized in the completion process. An example of missing reflection is displayed in Figure 12. Removing moving shadows (especially soft ones), in a videos can prove quite difficult because they are hard to discriminate correctly using binary masks, while pieces of outer penumbra which are left untreated will cause ghost-like artifacts. Correct reflections can also be very difficult to restore, especially in non-flat surfaces.

(a) Original frame (b) Frame after video completion

Figure 12: A completed frame from [JTWT06]. The result video is missing reflections and shadows over the glossy surface floor.

9.1. Conclusion

Data-driven completion methods have received much attention in the past decade. The usefulness of these methods, especially in completing large missing areas in images and video has been well established. While image completion has received plenty of attention from the research community, resulting in the recent development of fast and robust methods that have also found much practical use, the problem of video completion is not as mature. The additional time dimension in video completion, makes the video version of the problem both more difficult to solve plausibly and harder to compute. Nevertheless, we believe that the huge potential of editing tools that could be developed with the evolution of robust video completion methods is a strong driving force for research. Hopefully, this survey of the subject will help spread the knowledge regarding the current state-of-the-art and drive new research efforts in the field.

References

[AM93] ARYA S., MOUNT D. M.: Approximate nearest neighbor queries in fixed dimensions. In *Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms* (1993), Society for Industrial and Applied Mathematics, pp. 271–280. 7

[Ash01] ASHIKHMIN M.: Synthesizing natural textures. In *Proceedings of the 2001 symposium on Interactive 3D graphics* (New York, NY, USA, 2001), I3D '01, ACM, pp. 217–226. 7

[BBS01] BERTALMIO M., BERTOZZI A., SAPIRO G.: Navier-stokes, fluid dynamics, and image and video inpainting. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 1, pp. I–355 – I–362 vol.1. 3

[BGD*10] BUGEAU A., GARGALLO P., D'HONDT O., HERVIEU A., PAPADAKIS N., CASELLES V.: Coherent background video inpainting through kalman smoothing along trajectories. In *Vision, Modeling, and Visualization (2010)* (2010), The Eurographics Association, pp. 123–130. 8

[BLLC02] BORNARD R., LECAN E., LABORELLI L., CHENOT J.-H.: Missing data correction in still images and image sequences. In *Proceedings of the tenth ACM international conference on Multimedia* (New York, NY, USA, 2002), MULTIMEDIA '02, ACM, pp. 355–361. 6, 7

[BSCB00] BERTALMIO M., SAPIRO G., CASELLES V., BALLESTER C.: Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 417–424. 3

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3 (Aug. 2009). 5, 7

[BSHK04] BHAT K. S., SEITZ S. M., HODGINS J. K., KHOSLA P. K.: Flow-based video synthesis and editing. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 360–363. 4, 6, 7

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 23*, 11 (2001), 1222–1239. 9

[CFJ05] CHEUNG V., FREY B., JOJIC N.: Video epitomes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 1, pp. 42–49 vol. 1. 5

[CPT04] CRIMINISI A., PEREZ P., TOYAMA K.: Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing 13*, 9 (2004), 1200–1212. 2, 3, 4, 6, 10, 12

[CS05] CHAN T. F., SHEN J. J.: Variational image inpainting. *Communications on Pure and Applied Mathematics 58*, 5 (2005), 579–619. 3

[CZV06] CHEUNG S.-C. S., ZHAO J., VENKATESH M. V.: Efficient object-based video inpainting. In *Proc. IEEE Int Image Processing Conf* (2006), IEEE, pp. 705–708. 4, 10

[Dav98] DAVIS J.: Mosaics of scenes with moving objects. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on* (1998), IEEE, pp. 354–360. 11

[DCOY03] DRORI I., COHEN-OR D., YESHURUN H.: Fragment-based image completion. *ACM Trans. Graph. 22*, 3 (July 2003), 303–312. 3, 5, 6

[EF01] EFROS A. A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001* (August 2001), 341–346. 3, 4, 7, 8, 9

[EGLM12] EBDELLI M., GUILLEMOT C., LE MEUR O.: Examplar-based video inpainting with motion-compensated neighbor embedding. In *Image Processing (ICIP), 2012 19th IEEE International Conference on* (2012), pp. 1737–1740. 4, 7

[EL99] EFROS A., LEUNG T.: Texture synthesis by non-parametric sampling. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on* (1999), vol. 2, ICCV, pp. 1033–1038 vol.2. 3, 4, 5, 6

[GBBM09] GUNNEWIEK R. K., BERRETTY R.-P., BARENBRUG B., MAGALHAES J.: Coherent spatial and temporal occlusion generation. In *IS&T/SPIE Electronic Imaging* (2009), International Society for Optics and Photonics, pp. 723713–723713. 7

[GKT*12] GRANADOS M., KIM K. I., TOMPKIN J., KAUTZ J., THEOBALT C.: Background inpainting for videos with dynamic objects and a free-moving camera. In *ECCV (1)* (2012), Fitzgibbon A. W., Lazebnik S., Perona P., Sato Y., Schmid C., (Eds.), vol. 7572 of *Lecture Notes in Computer Science*, Springer, pp. 682–695. 4, 11

[GTK*12] GRANADOS M., TOMPKIN J., KIM K., GRAU O., KAUTZ J., THEOBALT C.: How not to be seen - object removal from videos of crowded scenes. *Computer Graphics Forum 31*, 2pt1 (2012), 219–228. 9

[HE07] HAYS J., EFROS A. A.: Scene completion using millions of photographs. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 12, 13

[JBBB99] JOYEUX L., BUISSON O., BESSERER B., BOUKIR S.: Detection and removal of line scratches in motion picture films. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* (1999), vol. 1, IEEE. 2

[JHM05] JIA Y.-T., HU S.-M., MARTIN R. R.: Video completion using tracking and fragment merging. *The Visual Computer 21*, 8-10 (2005), 601–610. 6, 7, 8

[JT03] JIA J., TANG C.-K.: Image repairing: Robust image synthesis by adaptive nd tensor voting. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on* (2003), vol. 1, IEEE, pp. I–643. 11

[JTPTT04] JIA J., TAI-PANG W., TAI Y.-W., TANG C.-K.: Video repairing: Inference of foreground and background under severe occlusion. In *In Proc. Computer Vision and Pattern Recognition* (2004), pp. 364–371. 4, 10, 11, 12

[JTWT06] JIA J., TAI Y.-W., WU T.-P., TANG C.-K.: Video repairing under variable illumination using cyclic motions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 28*, 5 (2006), 832–839. 4, 12, 13

[KBBN05] KUMAR S., BISWAS M., BELONGIE S., NGUYEN T.: Spatio-temporal texture synthesis and image inpainting for video applications. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on* (2005), vol. 2, pp. II–85–8. 6, 8

[KCR05] KOKARAM A. C., COLLIS B., ROBINSON S.: Automated rig removal with bayesian motion interpolation. *IEE Proceedings-Vision, Image and Signal Processing 152*, 4 (2005), 407–414. 2

[KDM02] KILTHAU S. L., DREW M. S., MOLLER T.: Full search content independent block matching based on the fast fourier transform. In *Image Processing. 2002. Proceedings. 2002 International Conference on* (2002), vol. 1, IEEE, pp. I–669. 6

[KKDK12] KOPF J., KIENZLE W., DRUCKER S., KANG S. B.: Quality prediction for image completion. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 131:1–131:8. 12

[KS12] KOOCHARI A., SORYANI M.: Video object inpainting: a scale-robust method. *Imaging Science Journal, The 60*, 5 (2012), 272–284. 12

[KSE*03] KWATRA V., SCHÖDL A., ESSA I., TURK G., BOBICK A.: Graphcut textures: image and video synthesis using graph cuts. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 277–286. 9

[LC10] LIN C.-W., CHENG N.-C.: Video bsckground inpainting using dynamic texture synthesis. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on* (2010), IEEE, pp. 1559–1562. 12

[LH05] LEFEBVRE S., HOPPE H.: Parallel controllable texture synthesis. *ACM Trans. Graph. 24*, 3 (July 2005), 777–786. 6

[LLS*11] LING C.-H., LIN C.-W., SU C.-W., CHEN Y.-S., LIAO H.-Y.: Virtual contour guided video object inpainting using posture mapping and retrieval. *Multimedia, IEEE Transactions on 13*, 2 (2011), 292–302. 10, 12

[Mah10] MAHALINGAM V. V.: *Digital inpainting algorithms and evaluation*. Doctoral dissertation, University of Kentucky, 2010. 12

[MM98] MASNOU S., MOREL J. M.: Level lines based disocclusion. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on* (1998), pp. 259–263 vol.3. 3

[MOG*06] MATSUSHITA Y., OFEK E., GE W., TANG X., SHUM H.-Y.: Full-frame video stabilization with motion inpainting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 28*, 7 (2006), 1150–1163. 2

[NFP*13] NEWSON A., FRADET M., PÉREZ P., ALMANSA A., GOUSSEAU Y.: Towards fast, generic video inpainting. In *Proc. Conf. Visual Media Production (CVMP)* (June 2013). 5

[PGB03] PEREZ P., GANGNET M., BLAKE A.: Poisson image editing. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 313–318. 11

[PKVP09] PRITCH Y., KAV-VENAKI E., PELEG S.: Shift-map image editing. In *ICCV'09* (Kyoto, Sept 2009), pp. 151–158. 9

[PL95] PAGET R., LONGSTAFF D.: Texture synthesis via a nonparametric markov random field. *Proceedings of DICTA–95, Digital Image Computing: Techniques and Applications, Anthony Maeder and Brian Lovell, Eds., Brisbane, Australia 1* (1995), 547–552. 3

[PL98] PAGET R., LONGSTAFF I.: Texture synthesis via a noncausal nonparametric multiscale markov random field. *Image Processing, IEEE Transactions on 7*, 6 (1998), 925–931. 3

[PM96] PO L.-M., MA W.-C.: A novel four-step search algorithm for fast block motion estimation. *Circuits and Systems for Video Technology, IEEE Transactions on 6*, 3 (1996), 313–317. 7

[PP93] POPAT K., PICARD R. W.: Novel cluster-based probability model for texture synthesis, classification, and compression. In *In Visual Communications and Image Processing* (1993), pp. 756–768. 3

[PSB05] PATWARDHAN K. A., SAPIRO G., BERTALMIO M.: Video inpainting of occluding and occluded objects. In *Proc. IEEE Int. Conf. Image Processing ICIP 2005* (2005), vol. 2. 4, 6, 7, 8

[PSB07] PATWARDHAN K. A., SAPIRO G., BERTALMIO M.: Video inpainting under constrained camera motion. *IEEE Transactions on Image Processing 16*, 2 (2007), 545–553. 4, 6, 7, 8, 12

[RK11] RAIMBAULT F., KOKARAM A.: Stereo video inpainting. In *IS&T/SPIE Electronic Imaging* (2011), International Society for Optics and Photonics, pp. 78631B–78631B. 7

[SLCF06] SHEN Y., LU F., CAO X., FOROOSH H.: Video completion for perspective camera under constrained motion. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* (2006), vol. 3, IEEE, pp. 63–66. 7, 8, 10, 12

[SMTK06] SHIRATORI T., MATSUSHITA Y., TANG X., KANG S. B.: Video completion by motion field transfer. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 1, IEEE, pp. 411–418. 8

[SS97] SZELISKI R., SHUM H.-Y.: Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (1997), ACM Press/Addison-Wesley Publishing Co., pp. 251–258. 11, 12

[STH09] SHIH T. K., TANG N. C., HWANG J.-N.: Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity. *IEEE Transactions on Circuits and Systems for Video Technology 19*, 3 (2009), 347–360. 6, 7, 8

[STTZ08] SHIH T. K., TAN N. C., TSAI J. C., ZHONG H.-Y.: Video falsifying by motion interpolation and inpainting. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), IEEE, pp. 1–8. 10

[STY*06] SHIH T. K., TANG N. C., YEH W.-S., CHEN T.-J., LEE W.: Video inpainting and implant via diversified temporal continuations. In *Proceedings of the 14th annual ACM international conference on Multimedia* (New York, NY, USA, 2006), MULTIMEDIA '06, ACM, pp. 133–136. 6, 7

[SYJS05] SUN J., YUAN L., JIA J., SHUM H.-Y.: Image completion with structure propagation. In *ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), SIGGRAPH '05, ACM, pp. 861–868. 4, 8

[THS*11] TANG N., HSU C.-T., SU C.-W., SHIH T., LIAO H.-Y. M.: Video inpainting on digitized vintage films via maintaining spatiotemporal continuity. *Multimedia, IEEE Transactions on 13*, 4 (2011), 602–614. 2

[VcSCZ09] VENKATESH M. V., CHING SAMSON CHEUNG S., ZHAO J.: Efficient object-based video inpainting. *Pattern Recognition Letters 30*, 2 (2009), 168 – 179. <ce:title>Video-based Object and Event Analysis</ce:title>. 10

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 479–488. 3, 5, 6

[WLKT09] WEI L.-Y., LEFEBVRE S., KWATRA V., TURK G.: State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report, EG-STAR* (2009), Eurographics Association. 3

[WLL07] WANG H., LI H., LI B.: Video inpainting for largely occluded moving human. In *Multimedia and Expo, 2007 IEEE International Conference on* (2007), IEEE, pp. 1719–1722. 10

[WSI04] WEXLER Y., SHECHTMAN E., IRANI M.: Space-time video completion. In *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition CVPR 2004* (2004), vol. 1. 4, 6, 8

[WSI07] WEXLER Y., SHECHTMAN E., IRANI M.: Space-time completion of video. *IEEE Transactions on Pattern Analysis and Machine Intelligence 29*, 3 (2007), 463–476. 2, 4, 5, 6, 7, 8

[XGY*11] XIA A., GUI Y., YAO L., MA L., LIN X.: Exemplar-based object removal in video using gmm. In *Multimedia and Signal Processing (CMSP), 2011 International Conference on* (2011), vol. 1, pp. 366–370. 7

[YWK05] YAN W.-Q., WANG J., KANKANHALLI M. S.: Automatic video logo detection and removal. *Multimedia Systems 10*, 5 (2005), 379–391. 2

[ZXS05] ZHANG Y., XIAO J., SHAH M.: Motion layer based object removal in videos. In *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on* (jan. 2005), vol. 1, pp. 516 –521. 7, 8, 11