IEEE J. Selected Topics in Signal Processing, Vol. 5, #2, pp. 321-334, 2011

Ultrawide Foveated Video Extrapolation

Tamar Avraham and Yoav Y. Schechner, *Member, IEEE*
Department of Electrical Engineering
Technion - Israel Inst. Technology, Haifa 32000, Israel
tammya@cs.technion.ac.il, yoav@ee.technion.ac.il

Abstract—Consider the task of creating a very wide visual extrapolation, i.e., a synthetic continuation of the field of view much beyond the acquired data. Existing related methods deal mainly with filling in holes in images and video. These methods are very time consuming and often prone to noticeable artifacts. The probability for artifacts grows as the synthesized regions become more distant from the domain of the raw video. Therefore, such methods do not lend themselves easily to very large extrapolations. We suggest an approach to enable this task. First, an improved completion algorithm that rejects peripheral distractions significantly reduces attention-drawing artifacts. Second, a foveated video extrapolation approach exploits weaknesses of the human visual system, in order to enable efficient extrapolation of video, while further reducing attention-drawing artifacts. Consider a screen showing the raw video. Let the region beyond the raw video domain reside outside the field corresponding to the viewer's fovea. Then, the farther the extrapolated synthetic region is from the raw field of view, the more the spatial resolution can be reduced. This enables image synthesis using spatial blocks that become gradually coarser and significantly fewer (per unit area), as the extrapolated region expands. The substantial reduction in the number of synthesized blocks notably speeds the process and increases the probability of success without distracting artifacts. Furthermore, results supporting the foveated approach are obtained by a user study.

Index Terms—Video extrapolation, Image and video completion, Foveated vision, Display technology

1 Introduction

Large high-quality display hardware, including television and computer monitors have become common consumer items. Thus, much effort is put nowadays into video enhancement, with the goal of a more enjoyable viewing experience. Substantial effort is invested in overcoming artifacts caused by signal noise and lossy compression, or in up-sampling low-resolution video by super-resolution techniques. The objective of all such efforts is to improve the appearance of the area encapsulated *within* the raw field-of-view (FOV).

An alternative direction for enhancing viewer experience is to create a spatial continuation of the video *outside* the raw FOV (Fig. 1). This is our goal here. We refer to such external continuation as video *extrapolation*. The video expansion is designed to enhance the viewer's sensation of being embedded in the scene rather than seeing a mundane, unrelated background. A rough step in this direction is taken by the Ambilight television system [2] (see Fig. 2). There, the wall around the television screen is backlit from behind the set. This illumination matches in realtime the azimuthal distribution of colors that exists at the margin of each frame. Related setups using varying ambient illumination are discussed in [14] and [31]. In our work, the extrapolation is an actual video. We aim for *very wide* video extrapolation, as illustrated in Fig. 1.

A related task deals with image and video completion. A part of the FOV is considered as a void to be filled by synthetic visual content. The filled content should appear realistic and consistent with the true, nearby data. Approaches for such tasks are sometimes referred to as *inpainting* (e.g., [5], [25]), *texture synthesis* (e.g., [10], [11], [15], [41]) and image *retargeting* (e.g., [34], [37]). Such methods have focused on filling in areas that occupy a rather small percentage (usually holes) of the raw FOV. One reason for this preference is that completing large areas is prone to disturbing and distracting artifacts. Another reason is that large areas take a long time to fill. Here, we address both aspects, aiming for a very wide (external) completion.

1.1 Related Work

Numerous studies deal with image and video completion. Here we mention a small subset. The most popular completion methods use patches from other locations in the image or video as source information for synthesizing data inside target holes (e.g., [8], [9], [23], [38], [42], [43]). In most of these methods (e.g., [8], [9]), the completion process is sequential, propagating the fill in from the boundary of the known data domain, into uncharted territories. At each step, a small image (or video) block is pasted or updated. With some probability, the update at a specific iteration is "wrong," i.e., it fills-in a small block that is subjectively inconsistent with the video content, according to the life-experience of the viewer. From this point on, this small error affects the consecutive iterations, leading to a large, unignorable artifact. Since small errors serve as seeds for large ones, sequential schemes are highly susceptible to the creation of distracting artifacts. To tackle this problem, Sun et al. [38] suggested that the user mark curves providing cues for the desired structure in still images. Wexler et

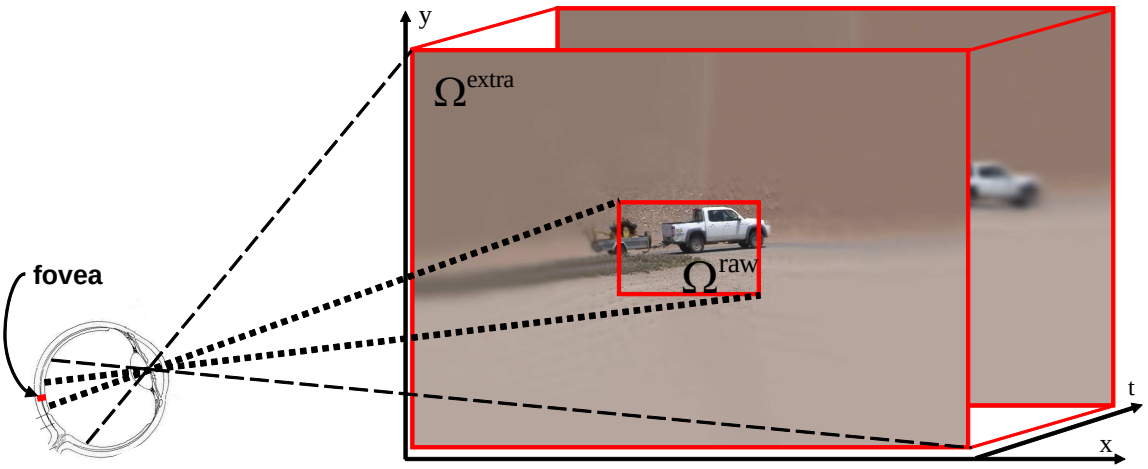


Fig. 1. A raw input video resides in a spatiotemporal set Ω^{raw}. It is extrapolated to a video in domain Ω^{extra}. Wherever the viewer looks on in Ω^{raw}, the extrapolated region is outside the fovea. Thus, the extrapolated video may be synthesized and displayed at a scale that become gradually coarser as the extrapolated region expands, following the natural foveated resolution. The marked reduction in the need to synthesize details, significantly speeds up the process and significantly increases the probability of success without distracting artifacts.

Fig. 2. Philips Ambilight TV. The wall around the television screen is illuminated with colors matching the margins of the video frames. Video extrapolation should extend the scene by actual video [2].

al. [42] and Komodakis et al. [23] suggest methods whose goal is global optimization. However, we found that this incurs a high cost in terms of computation time.

Source patches can become larger and fewer when using additional photos [43]. A recent appealing method [16] suggests relying on a database of millions of still photos. Then, just a single, large patch from a database photo can replace the hole. This approach reduces the number of artifacts and offers nice results. However, using this approach for *video* would require a huge database.

Some video completion methods suggest using mid-level image analysis, such as segmentation, motion analysis, and tracking (e.g., [20], [21], [22], [26], [28]). However, it requires the analysis to have high-reliability. Nevertheless, incorporating such analysis in completion methods is beneficial.

1.2 Our Approach

We tackle the runtime and artifacts problems, in a quest for very wide extrapolation. The context of the task in this work is enhancement of the viewer's experience, while the viewer is *aware* of the true-data region and its boundary. The context of this display task can exploit the *foveated* nature of the human visual system: outside the true data region, the viewer is much less sensitive to fine spatial details; hence, as we will show, these can be gradually sacrificed for the sake of speed and reliability. We assume that the viewer's focus is on the raw data domain at all times. This cannot be guaranteed. However, we make an effort to create the extrapolation without attention-drawing artifacts.

Extrapolation can be useful also inside the main display screen. Internet streaming video and short amateurish videos, taken e.g., by digital cameras integrated in cellular phones, may have a resolution much lower than display screens and projectors. Thus, watching such video in its raw resolution wastes significant display real-estate in return for void regions, or (worse) distracting objects. Extrapolating the video to the full screen could exploit these unused regions, in a way that enhances the view even if the extrapolation resolution is rough. Additional display configurations that can benefit from extrapolation are discussed in Section 8.1.

We thus use a foveated approach. It relies on the tolerance of the visual system to spatial coarsening in the periphery. This enables very large areas to be filled in with far fewer blocks (iterations) than implied by the native display resolution. This reduction in the number of blocks has several benefits. First, it leads to a faster, efficient process, which makes its use in video more feasible than standard image completion. Second, it greatly reduces the number or potential error-seeds.

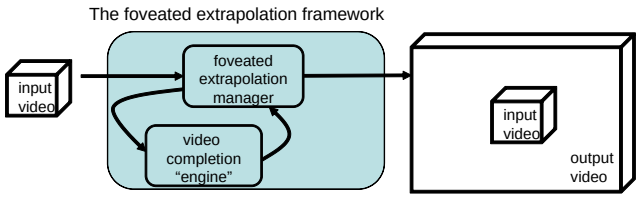


Fig. 3. The foveated extrapolation algorithm provides a general framework, into which a wide range of full resolution video completion algorithms can be integrated as an "engine".

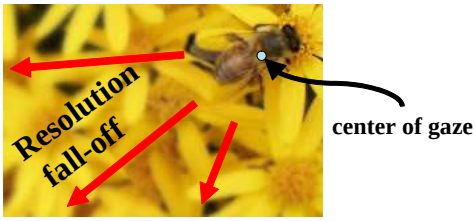


Fig. 4. A variant resolution image [12]. The center of the observer's gaze is on the bee.

This leads to a corresponding reduction of artifacts that would otherwise distract the viewer from the true and known data domain.

The foveated extrapolation algorithm suggested here performs several discrete extrapolation steps, each at a different scale (as described in Section 4). Each scale-step can be considered a black box and thus a wide range of completion algorithms can serve as this "engine." Thus, the foveated approach offers a *general framework* that can easily exploit future improvements in video completion (Fig. 3). Nevertheless, we suggest here a specific patch-based video completion algorithm that rejects peripheral distraction, to serve as this "engine" (Section 3).

The benefits of the foveated approach in reliability and runtime are demonstrated both analytically (Section 5) and on natural landscape video and images (Section 6). The viewers' response to the visual results is reported in Section 7. We start by describing relevant background (Section 2).

2 Preliminaries

2.1 Foveated Vision and Representation

The fovea occupies $\approx 1\%$ of the retina. It has high density of photoreceptors in a narrow region. This region thus corresponds to high spatial resolution and is used in high resolution tasks, which require attention. Outside this narrow region, the density of photoreceptors falls off, and the perceived signals there are spatially much coarser.

The resolution falloff of foveated human vision can be incorporated for efficient image representation and display. This effect was used for image compression (e.g., [12], [40]). There, the image display resolution falls-off gradually with the distance from the center of the viewer's gaze. See Fig. 4.

Define the effective visual resolution as the number of resolvable pixels per unit lateral (not solid) angle. Let α be the visual angle, i.e., the angle subtended by an object location, relative to the optical axis of the eye. The effective resolution then falls off monotonically, following a function $R(\alpha)$. Based on findings from psychophysical studies, e.g., [32], an approximated formula was defined in [13],

$$R(\alpha) = \frac{\epsilon}{\epsilon + \alpha} \ , \tag{1}$$

where α is given in degrees and $\epsilon = 2.3^o$. Eq. (1) is normalized such that $R(0) = 1$. We use it in our work.

Geisler and Perry [12] suggest a three-step method to gradually decrease the image resolution according to $R(\alpha)$: (a) creating a quantized scale map, based on $R(\alpha)$; (b) creating a multiscale representation of the image; (c) fusing selected multiscale image layers. We use analogous steps in our foveated video extrapolation algorithm. We now describe the steps as suggested in [12]. In Section 4 we describe how we adapt them to foveated extrapolation. Depending on the distance of the display screen, each pixel $\mathbf{x} = (x, y)$ is perceived at an angle $\alpha(\mathbf{x})$. This directly yields an effective relative *resolution map* $R^{\text{map}}(\mathbf{x}) \equiv R[\alpha(\mathbf{x})]$ over the image domain. This relative resolution map is then quantized to a few discrete values $\{f^\sigma\}_{\sigma=0}^{N_{\text{scales}}-1}$. Here, $\sigma \in [0, N_{\text{scales}} - 1]$ is a scale index, and $f < 1$ is a constant which determines the resolution quantization steps. Following this quantization, each pixel is associated with two discrete resolutions, which bound the quantization interval around $R^{\text{map}}(\mathbf{x})$:

$$\tilde{\sigma}(\mathbf{x}) : \ f^{\tilde{\sigma}(\mathbf{x})+1} \leq R^{\text{map}}(\mathbf{x}) \leq f^{\tilde{\sigma}(\mathbf{x})} \ , \ \sigma(\mathbf{x}) \in [0, N_{\text{scales}} - 1] \ . \tag{2}$$

Now, the image content is transformed to a multiscale representation. Geisler and Perry [12] used a Gaussian pyramid [6] of the input image $I^{\text{raw}}(\mathbf{x})$. Each pyramid level corresponds to a scale index σ. The image in that scale has been recursively scaled down by a factor of f^{-1}. Then, each such level is expanded to occupy the same pixel domain as that of I^{raw}. This results in a set of multiscale layers $\{G_\sigma\}_{\sigma=0}^{N_{\text{scales}}}$, where blur increases with σ.

Finally, the displayed image is created, by fusing two scale-levels at each location $\mathbf{x}$. The two scales are the bounds set by Eq. (2). Fusion is obtained by a convex superposition

$$I^{\text{out}}(\mathbf{x}) = W(\mathbf{x}) G_{\tilde{\sigma}(\mathbf{x})+1}(\mathbf{x}) + [1 - W(\mathbf{x})] G_{\tilde{\sigma}(\mathbf{x})}(\mathbf{x}) \ , \tag{3}$$

where $W(\mathbf{x})$ is a weighting function, in the range $[0, 1]$. Here, $W(\mathbf{x}) = 1$ or 0 where the continuous-valued effective relative resolution $R^{\text{map}}(\mathbf{x})$ coincides, respectively, with the upper or lower quantization bounds set by Eq. (2).

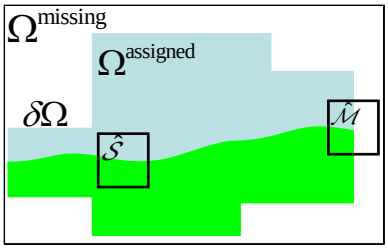


Fig. 5. A core completion engine [8]. The domain Ω^{assigned} is already completed; Ω^{missing} is the domain to be completed; $\delta\Omega$ is the boundary between them. The block $\hat{\mathcal{M}}$ has priority for being completed now, due to a strong edge crossing $\delta\Omega$ at its center point. The block $\hat{\mathcal{S}}$ is a good source for completing $\hat{\mathcal{M}}$.

2.2 Core Completion Algorithm

As illustrated in Fig. 3, our foveated video extrapolation algorithm (which we describe in Section 4) uses a full resolution video completion algorithm as a black-box "engine." We suggest a specific engine (Section 3) which follows the patch-based algorithm suggested by Criminisi et al. [8]. We found that this method required significantly less processing time than other successful patch-based algorithms. With the adjustments we detail below in Section 3, it is effective enough as the completion "engine." However, other methods (e.g., [9], [23], [42]) can be integrated into the foveated extrapolation framework as well.

In this section we briefly describe the completion method of Criminisi et al. [8]. Our description is in the context of video extrapolation. Let Ω^{raw} be the spatio-temporal (ST) domain of an input video. We wish to extrapolate the video to domain Ω^{extra} surrounding Ω^{raw}, as illustrated in Fig. 1. The algorithm is iterative. At any iteration, let Ω^{assigned} be the set of pixels to which a value was already assigned, and $\Omega^{\text{missing}} = \Omega^{\text{extra}} - \Omega^{\text{assigned}}$ the set of pixels to which an assigned value is still missing, as illustrated in Fig. 5. Initially, $\Omega^{\text{assigned}} = \Omega^{\text{raw}}$. In each iteration, Ω^{assigned} grows until $\Omega^{\text{assigned}} = \Omega^{\text{extra}}$.

In any iteration, a target ST block $\hat{\mathcal{M}}$ is selected for completion, and another ST block $\mathcal{S}$ is selected as the data source for this completion. The ST blocks[1] are of constant size of $w_x \times w_y \times w_t$. The target block is always on the boundary $\delta\Omega$ between Ω^{assigned} and Ω^{missing}. Blocks that include strong ST edges crossing $\delta\Omega$, and those that include less missing pixels, are selected as targets in an earlier iteration, for better propagation of image structure. After $\hat{\mathcal{M}}$ is selected, a source $\hat{\mathcal{S}}$ is searched throughout Ω^{raw}: a cost

$$r(\mathcal{S}) = \text{SSD\_LAB}(\mathcal{S}_{\text{assigned}}, \hat{\mathcal{M}}_{\text{assigned}}) \qquad (4)$$

is assigned to each *potential source* $\mathcal{S}$, where SSD_LAB is the sum-of-square-differences[2] (SSD) in the CIE-Lab color space[3]. Here, $\hat{\mathcal{M}}_{\text{assigned}} = \hat{\mathcal{M}} \cap \Omega^{\text{assigned}}$ and $\mathcal{S}_{\text{assigned}}$ are the corresponding pixels in $\mathcal{S}$. The selected block is the one for which

$$\hat{\mathcal{S}} = \arg \min_{\mathcal{S} \subset \Omega^{\text{raw}}} r(\mathcal{S}) \ . \qquad (5)$$

Pixels from the selected source $\hat{\mathcal{S}}$ are used for filling in the missing pixels in $\hat{\mathcal{M}}$. Eqs. (4) and (5) yield a source block having a subregion that is visually similar to the known part of the target block. Using this selection usually results in a completed target block which is similar to a real block from Ω^{raw}. Consequently, the target block also appears realistic.

As discussed above, this method (denoted *fullres-core*) is efficient relative to other patch-based completion algorithms, and yields good results when completing small holes. It nevertheless suffers from long processing time and introduces too many artifacts, when used for completing wide peripheral regions.

3 Algorithm for Peripheral Extrapolation

Here we describe an algorithm having *Peripheral Avoidance of Distractions* (PAD). This new algorithm adjusts the core algorithm in order to handle peripheral extrapolation, by reducing attention-drawing artifacts (Sec 3.1 and Section 3.2) and by reducing processing time (Section 3.3).

3.1 Rejecting Peripheral Distractions

It is desirable that the extrapolated image periphery would be visually "boring" and predictable. We do not want the viewer's attention to be distracted by salient artificial objects that spring from nowhere. Such objects may be outliers selected by Eq. (5), rather than solid representatives of potentially good sources. We decrease the probability for such artifacts by two changes to the core algorithm: First, we penalize candidate source blocks that introduce strong spatial or temporal gradients. Second, we counter outliers by seeking a source block that has high similarity to several other candidate source blocks. This is accomplished by preferring candidates that are centers of clusters of high-scoring candidates. (See resulting examples in Fig. 6.)

Penalizing edges: According to Eq. (5), the best fitting block $\hat{\mathcal{S}}$ is similar to $\hat{\mathcal{M}}$ in the part where $\hat{\mathcal{M}}$ was previously assigned ($\hat{\mathcal{M}}_{\text{assigned}}$). What about the complementary part of $\hat{\mathcal{S}}$, i.e., $\hat{\mathcal{S}}_{\text{complement}} = \hat{\mathcal{S}} / \hat{\mathcal{S}}_{\text{assigned}}$? It might contain a noticeable ST feature. Strong ST variations are not desirable in the periphery as they may draw the

1. Following [8], the blocks are boxes. However, other shaped blocks can be used, e.g. spherical, by masking.

2. Calculating $r(\mathcal{S})$ using the Fast-Fourier-Transform (FFT) [24] speeds up the process.

3. Other more robust similarity measures (e.g., the Hausdorff distance [18] or the CW-SSIM index [36]) can easily replace the SSD criterion.

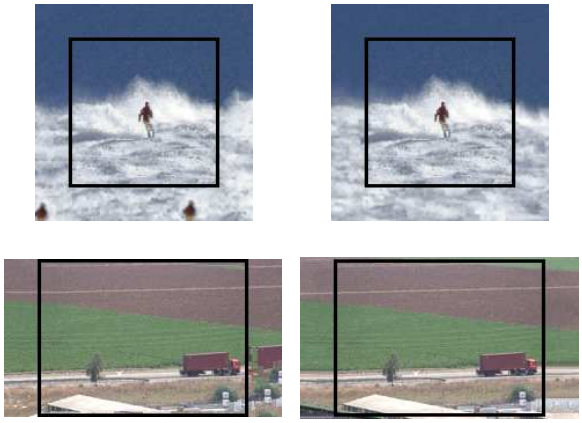


Fig. 6. Demonstrating the preference to reject peripheral distractions. [Top left]: Applying the core completion algorithm to a video clip of a skier (inside the black frame) yields copies of the skier which occasionally appear and disappear in the extrapolated region. [Top right]: Result of applying the *PAD* completion algorithm, which rejects peripheral distractions. [Bottom left]: A frame from an extrapolated video using the core completion algorithm. [Bottom right]: The corresponding frame resulting from the *PAD* completion algorithm.

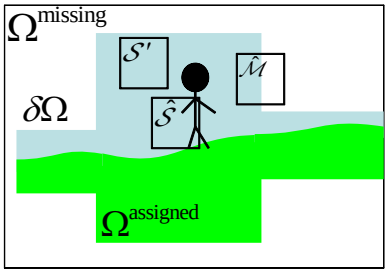


Fig. 7. Both $\hat{\mathcal{S}}$ and $\mathcal{S}'$ are good source candidates for completing $\hat{\mathcal{M}}$ according to the core algorithm. However, $\mathcal{S}'$ is a preferred source, as it does not introduce an attention-drawing object in the periphery.

viewer's attention, as illustrated in Figs. 6 and 7. On the other hand, there may be another potential source block $\mathcal{S}'$ that introduces no undesired ST edges and whose cost $r(\mathcal{S}')$ is only slightly higher than $r(\hat{\mathcal{S}})$. Such an $\mathcal{S}'$ is a more desired source candidate.

This principle is implemented as follows. Let $\mathbb{S}$ be the set of $N_{\text{candidate}}$ candidate blocks that rank best (e.g., according to Eq. 4). Now, for each $\mathcal{S} \in \mathbb{S}$, a penalty is set for blocks that introduce more prominent ST edges. This penalty is based on the ST gradient

$$g(\mathcal{S}) = \sum_{\mathbf{X} \in \mathcal{S}_{\text{complement}}} |\nabla I(\mathbf{X})| \ , \tag{6}$$

where $\mathbf{X} = (\mathbf{x}, t)$ is a video pixel defined by its spatial location $\mathbf{x}$ and the temporal index t. This penalty is considered later, when setting a final cost, as we discuss below.

Suppressing outliers: To suppress outliers, we prefer to select a candidate that is near the center of a cluster of candidates. A simple way to quantify this preference is to measure for each $\mathcal{S} \in \mathbb{S}$ the sum of 'dissimilarity' to all the other candidates $\tilde{\mathcal{S}} \in \mathbb{S}$,

$$h(\mathcal{S}) = \sum_{\tilde{\mathcal{S}} \in \mathbb{S}} \text{SSD\_LAB}(\mathcal{S}, \tilde{\mathcal{S}}) \ . \tag{7}$$

If $\mathcal{S}$ is near the center of a cluster, then the cost $h(\mathcal{S})$ is smaller than if $\mathcal{S}$ is an outlier.

Final cost: Finally, each $\mathcal{S} \in \mathbb{S}$ is assigned a cost based on $r(\mathcal{S})$, $g(\mathcal{S})$, and $h(\mathcal{S})$,

$$r'(\mathcal{S}) = \frac{r(\mathcal{S})}{\min_{\tilde{\mathcal{S}} \in \mathbb{S}} r(\tilde{\mathcal{S}})} + \frac{g(\mathcal{S})}{\min_{\tilde{\mathcal{S}} \in \mathbb{S}} g(\tilde{\mathcal{S}})} + \frac{h(\mathcal{S})}{\min_{\tilde{\mathcal{S}} \in \mathbb{S}} h(\tilde{\mathcal{S}})} \ . \tag{8}$$

The first term in Eq. (8) has the value 1 for the best fitting block, selected by Eq. (5). It has higher values in other candidate blocks. The second term in Eq. (8) has the value 1 for the candidate source $\mathcal{S}$, for which $\mathcal{S}_{\text{complement}}$ least introduces ST edges (gradients). This term has higher values in source candidates that have more edges in their complementary portion. The third term in Eq. (8) has the value 1 for the candidate source that is more likely to be the center of a cluster, according to Eq. (7). This term has higher values in other candidates.

Therefore, the lower $r'(\mathcal{S})$ is, the better $\mathcal{S}$ is as a candidate source for peripheral extrapolation. Now, instead of $r(\mathcal{S})$ defined in Eq. (4), it is possible to use $r'(\mathcal{S})$, and thus select an optimal source block. However, we take into consideration another aspect, as discussed in the next section.

3.2 Randomness

The best fit by Eq. (5) or by minimizing Eq. (8) may result in a specific ST block being selected repeatedly at consecutive iterations. This may cause synthesized regions to be perceived as repeatable and unnatural. As suggested by Efros and Leung [11], it is beneficial to introduce randomness in the selection of $\hat{\mathcal{S}}$.

Our algorithm considers the subset $\mathbb{S}' \subseteq \mathbb{S}$ of N_{top} candidates ($N_{\text{top}} \leq N_{\text{candidate}}$) that rank best in Eq. (8). Then, a block $\hat{\mathcal{S}} \in \mathbb{S}'$ is randomly selected to be *the* source block. The probability of this selection is set to be inversely proportional to the cost $r'(\mathcal{S})$, given in Eq. (8). The probability is thus

$$P(\mathcal{S} = \hat{\mathcal{S}}) = \frac{1/r'(\mathcal{S})}{\sum_{\tilde{\mathcal{S}} \in \mathbb{S}'} 1/r'(\tilde{\mathcal{S}})} \ . \tag{9}$$

3.3 Adaptive Search Domain

Patch-based completion methods demand significant processing time. As we detail in Section 6, for wide extrapolation, the core algorithm may require an intolerable amount of time. Most of the computation time

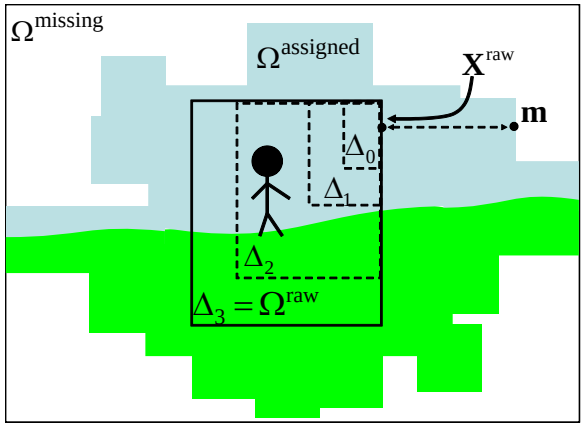


Fig. 8. Adaptive search domain. The point $\mathbf{X}^{\text{raw}}$ is the closest point in Ω^{raw} to the target point $\mathbf{m}$. The initial search domain is Δ_0. Only if a sufficient source block is not found in Δ_k does the search domain expands to Δ_{k+1}. The figure illustrates the process in 2D, for clarity.

in each iteration is spent on searching for a source block $\mathcal{S}$. The original algorithm [8], [24] searches the entire Ω^{raw}. Some methods reduce the search domain, either by filtering out irrelevant candidates using simple statistics (e.g., [9]) or by a prior semantic analysis of the scene (e.g., [20], [21], [22], [28]). We suggest a method for runtime reduction that does not rely on high level analysis. It exploits a characteristic of natural scenes: usually many good potential source blocks are found in the ST vicinity of $\mathcal{M}$, both spatially and temporally. Therefore, each search starts at a limited ST domain closest (spatially and temporally) to $\hat{\mathcal{M}}$. The search domain is then extended gradually only if a sufficiently similar block is not found. Let $\mathbf{m}$ be the center of the target block $\hat{\mathcal{M}}$, and let $\mathbf{X}^{\text{raw}}$ be the point closest to $\mathbf{m}$ (by Euclidean distance) in Ω^{raw} (see Fig. 8). Consider *search domains* $\Delta_0 \subset \Delta_1 \subset \Delta_2 ... \subset \Delta_n = \Omega^{\text{raw}}$ such that

$$\forall \mathbf{q} \in \Delta_k, \quad \|\mathbf{q} - \mathbf{X}^{\text{raw}}\|_\infty < \delta_0 \cdot 2^k, \quad k = 0, ..., n \ , \quad (10)$$

where δ_0 defines the ST length of the smallest search-domain.

Let $r_{\max}$ be a dissimilarity threshold. First, the search is only in Δ_0. If a source block $\mathcal{S}$ is found for which $r(\mathcal{S}) < r_{\max}$, the selection of the final source block $\hat{\mathcal{S}}$ is further carried out inside Δ_0, in the manner described is Secs. 3.1 and 3.2. Only if such an $\mathcal{S}$ is not found inside Δ_0, the procedure is repeated for Δ_1 (searching for $\mathcal{S} \in \Delta_1$ so that $r(\mathcal{S}) < r_{\max}$), and so forth. From our experience, this approach significantly reduces processing time: usually Δ_0 suffices. Rarely does the search expand to the whole Ω^{raw} [4].

4. Once a source block is selected, instead of simply copying the corresponding part of the selected source block to the 'unknown' part of the target block, we counter seam lines by feathering [7] (as done also by [9]).

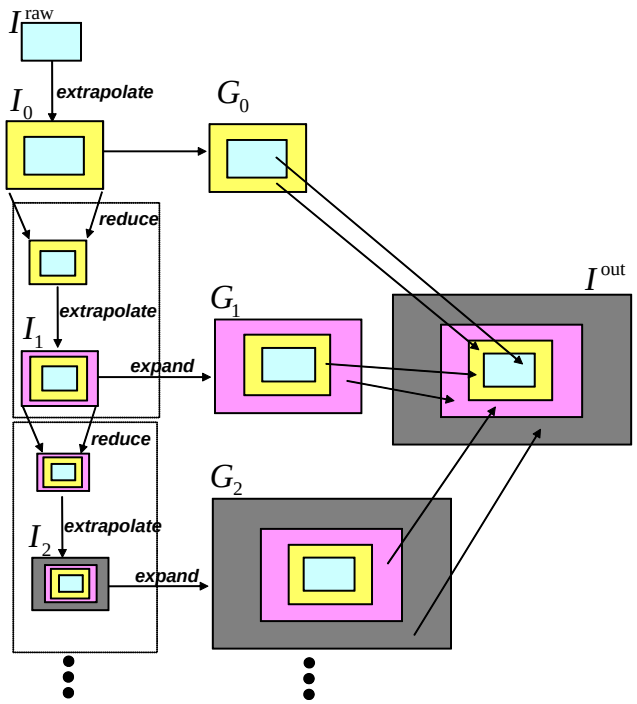


Fig. 9. A schematic description of foveated video extrapolation. The process includes a few discrete steps. At step σ, the result from the former step $I_{\sigma-1}$ is downsampled and further extrapolated in a coarser scale. The results of these steps are combined to construct I^{out}.

After applying the adjustments described in Section 3.1 and 3.2, the number of artifacts drops substantially. However, artifacts are still frequent when attempting to extrapolate very far from Ω^{raw}, at full resolution. In addition, the runtime for wide extrapolations is still intolerable even after the improvement described in Section 3.3.

For many display conditions, full resolution extrapolation is an overkill. The synthesized peripheral data usually addresses regions in the viewer's retina with a low density of photoreceptors. The foveated video extrapolation framework described in the next section exploits this characteristic, and introduces an approach that substantially reduces the runtime and further reduces the probability of noticeable artifacts.

4 Foveated Video Extrapolation

The foveated video extrapolation framework is illustrated in Fig. 9. The output video I^{out} has spatially varying resolution. In Ω^{raw}, $I^{\text{out}}(\Omega^{\text{raw}}) \equiv I^{\text{raw}}(\Omega^{\text{raw}})$, maintaining the original spatial sharpness. However, there is a continuous drop in human visual resolution away from Ω^{raw}. Aiming to exploit this, there is little benefit in performing the tedious extrapolation in the original scale. It is much more efficient to perform the extrapolation in a foveated fashion. As the extrapolation propagates outwards, the extrapolated content can be coarser. Hence, extrapolation can operate on a coarser scale version of

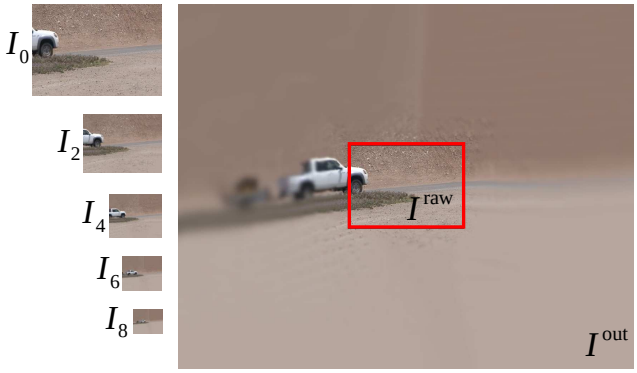


Fig. 10. Given an input video I^{raw}, the intermediate steps of the extrapolation process result in $I_0, ..., I_8$. The final result is I^{out}. Here, this is demonstrated in one frame from a movie that was extrapolated from 342×234 to 1280×1024, with $f = 1/\sqrt{2}$.

the previous step's result. Coarsening is done effectively in a few discrete steps. The results of these steps are combined to construct I^{out}, as shown in Fig. 10. In the following, we describe the principles in detail.

4.1 Extrapolation Resolution Map

In Section 2.1, a method for creating an image with spatially-varying resolution was described, based on [12]. Steps defined in [12] (and Section 2.1) are conveniently applied to our task. Specifically, the pyramid representation used in [12] is useful for us. Thus, our method has some analogous steps.

First, the relative resolution associated with each pixel $\mathbf{x}$ in the output domain Ω^{extra} is computed. In our case, we require that the raw *high* resolution be maintained across the *entire* Ω^{raw}. Thus, the *center of gaze* can be on *any* location within Ω^{raw} (not a particular point in the raw domain). The off-axis angle α (see Section 2.1) to a point $\mathbf{x}$ is thus defined relative to the boundary of Ω^{raw}.

Consider Fig. 11. Suppose the whole screen has a uniform display resolution of κ pixels per unit-length (e.g., dots-per-inch). Then, the pixel in Ω^{raw} that is closest to $\mathbf{x}$ is

$$\hat{\mathbf{x}}^{\text{raw}} = \arg \min_{\mathbf{x}^{\text{raw}} \in \Omega^{\text{raw}}} \|\mathbf{x} - \mathbf{x}^{\text{raw}}\|_2 \ . \tag{11}$$

The distance between $\mathbf{x}$ and Ω^{raw} is $d(\mathbf{x}) = \|\mathbf{x} - \hat{\mathbf{x}}^{\text{raw}}\|_2$ in units of pixels. Note that Eq. (11) automatically yields $d(\mathbf{x}) = 0$ for all $\mathbf{x} \in \Omega^{\text{raw}}$. Setting the origin of the coordinate system at the center of the screen, the distance of $\hat{\mathbf{x}}^{\text{raw}}$ from the origin is $w = \|\hat{\mathbf{x}}^{\text{raw}}\|_2$.

Let the viewer be situated opposite this center, at distance z. Then, relative to Ω^{raw}, the off-axis angle[5] is

$$\alpha(\mathbf{x}) \approx \arctan\{[w + d(\mathbf{x})]/(\kappa z)\} - \arctan[w/(\kappa z)] \ . \tag{12}$$

5. This is an approximated expression, since $\mathbf{x}$ and $\mathbf{x}^{\text{raw}}$ are generally not co-linear, with respect to the origin. The error in this approximation does not have significant implications in our task.

The resulting $\alpha(\mathbf{x})$ is used by us in Eq. (1), to yield a continuous valued *extrapolation resolution* map $R^{\text{map}}(\mathbf{x})$. Fig. 12 illustrates such a map.

Similarly to Eq. (2) in Section 2.1, quantization levels are set for $R^{\text{map}}(\mathbf{x})$. These levels define *extrapolation domains* $\Omega_0, \Omega_1, ..., \Omega_{(\text{N}_{\text{scales}}-1)}$, where

$$\Omega_\sigma = \{\mathbf{x} \in \Omega^{\text{extra}} : \quad f^{\sigma+1} < R^{\text{map}}(\mathbf{x}) \leq f^\sigma\} \ , \tag{13}$$

as illustrated in Fig. 12.

4.2 Extrapolation in Spatial Domains of Scale

The algorithm consists of two main parts. First, the video is extrapolated in the different domains, $\{\Omega_\sigma\}_{\sigma=0}^{\text{N}_{\text{scales}}-1}$. Then, feathering is applied to the domains, for a smooth transition between them. We now detail these parts.

The spatial domains $\{\Omega_\sigma\}_{\sigma=0}^{\text{N}_{\text{scales}}-1}$ are filled-in consecutively and outwards, starting from the inner domain Ω_0. Within each spatial domain, the operations are carried out in the corresponding spatial scale σ. The initial input is the raw video data I^{raw} over the domain Ω^{raw}. It is extrapolated into Ω_0 at full resolution. The result is a video sequence I_0, which occupies the domain $\Omega^{\text{raw}} \cup \Omega_0$, at the finest scale. This extrapolation step is expressed as

$$I_0(\mathbf{X}) = \mathcal{X}^{\text{extra}}\{I^{\text{raw}}(\mathbf{X})\} \ . \tag{14}$$

Up to this point, the method is similar to the known art of image completion.

Filling in of the outer domains is more interesting. For instance, the domain Ω_1 is outside the gaze of a viewer who looks at Ω^{raw}. Hence, Ω_1 can be filled in with blocks having effectively a coarser spatial extent. As shown later in Section 5, using coarser-scale blocks yields benefits in terms of artifacts and speed. To operate efficiently in a coarser scale, the video I_0 is spatially demagnified, in analogy to the REDUCE operation which was defined by Burt and Adelson [6] in the context of the Gaussian pyramid. Denote the demagnification operator as $\mathcal{R}$. Its result in this case is

$$I_0^{\text{reduced}} = \mathcal{R}\{I_0\} \ . \tag{15}$$

This demagnification is done with a lateral scale factor of f. Hence, the number of pixels in I_0^{reduced} relative to the number of pixels in I_0 is reduced by f^2. This is illustrated in Fig. 9. The demagnified video I_0^{reduced} still resides in $\{\Omega^{\text{raw}} \cup \Omega_0\}$, but now its coarser scale matches the scale assigned to the next, outer domain, Ω_1. Now, Ω_1 is filled-in, using the same extrapolation engine, but now the input is *not* the high-resolution video I_0, but the coarser (lower resolution) I_0^{reduced}, as illustrated in Fig. 9. This way, Ω_1 is filled in fast, with data corresponding to coarser spatial features.

In general, the result of such a step is

$$I_{\sigma+1} = \mathcal{X}^{\text{extra}}\{I_\sigma^{\text{reduced}}\} \qquad \text{for} \quad \sigma \in [0, 1, ...(N_{\text{scales}} - 2)], \tag{16}$$

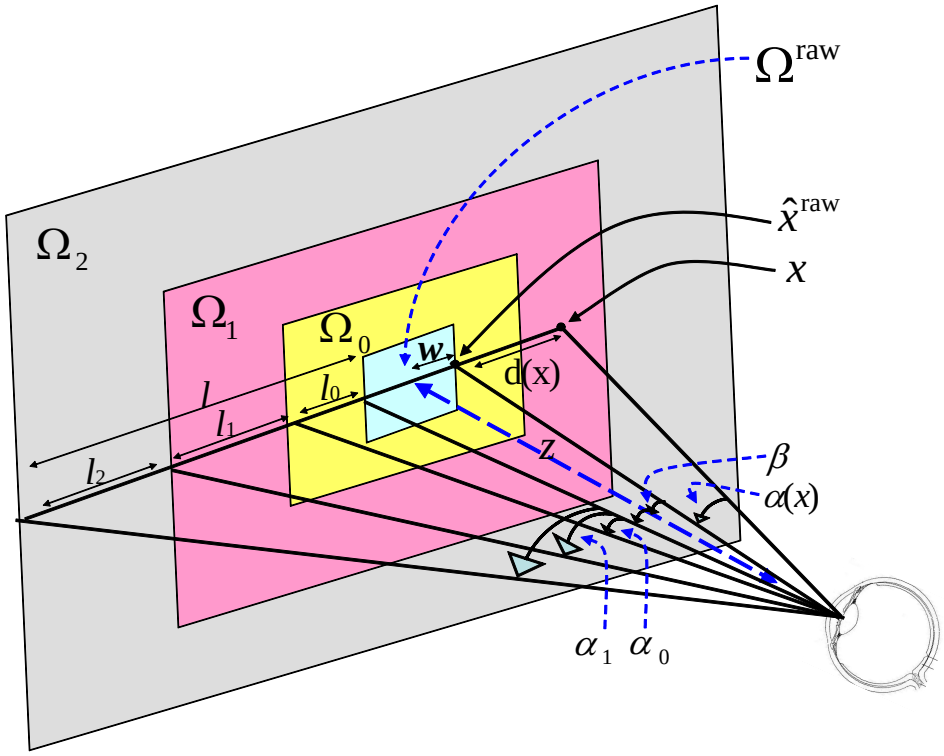


Fig. 11. Foveated extrapolation domains. The domain of the input data is Ω^{raw} having horizontal spatial width of $2w$ pixels. The input is extended by l pixels to each side. The approach synthesizes each of the domains $\Omega_0, ...\Omega_{N_{\text{scales}}-1}$ in a different scale. The viewer observes the screen from distance z, fixating at Ω^{raw}.

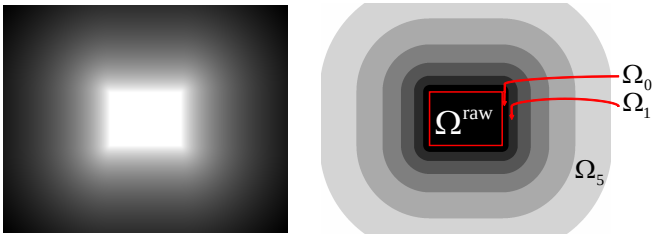


Fig. 12. [left]: The log of the *extrapolation resolution map* R^{map} when extrapolating from 320×240 to 1280×1024, for a 86 DPI screen and at distance of 0.5m. [Right]: The corresponding domains $\Omega_0, \Omega_1, ..., \Omega_{(\text{N}_{\text{scales}}-1)}$ defining the extrapolation steps. Here $f = 1/\sqrt{2}$.

where

$$I_\sigma^{\text{reduced}} = \mathcal{R}\{I_\sigma\} \ . \tag{17}$$

Note that the operator $\mathcal{X}^{\text{extra}}$ in Eq. (16) is exactly the same as the one used in Eq. (14). It uses blocks of size $w_x \times w_y \times w_t$, whatever its input is. So, the computational demands of matching a single block and merging it do not increase with the scale σ. Similarly, the operator $\mathcal{R}$ in Eq. (17) is exactly the same as the one used in Eq. (15). It demagnifies its input by a factor of f^{-1} in each spatial axis. Because of these properties, the video $I_\sigma^{\text{reduced}}$ always matches the scale assigned to the consecutive outer domain $\Omega_{\sigma+1}$, ensuring a proper transition for extrapolation in a coarser scale. Eqs. (16,17) are iterated, until the entire Ω^{extra} is filled.

The set of videos $\{I_\sigma\}_{\sigma=0}^{\text{N}_{\text{scales}}-1}$ should be brought back to a raw resolution representation, to enable their display jointly with I^{raw}. Thus each of the videos undergoes spatial magnification, using an expansion operator (see for example [6]) $\mathcal{E}_\sigma$, yielding

$$G_\sigma = \mathcal{E}_\sigma\{I_\sigma\} \ . \tag{18}$$

The operator $\mathcal{E}_\sigma$ magnifies its input spatially by a lateral scale factor of $f^{-\sigma}$. Hence, $I_0 \equiv G_0$, while the other videos are magnified such that the size of their output pixels corresponds to the raw pixel size.

Each video G_σ occupies a domain $\{\Omega^{\text{raw}} \cup (\bigcup_{b=0}^{\sigma} \Omega_b)\}$. Each such domain now has a physical size on the display screen, which is the same as the one that existed for these domains before demagnification had taken place. These videos should be fused. This is the second (and rather minor) part of algorithm. Based on Eqs. (2,13), a scale index $\tilde{\sigma}(\mathbf{x})$ is defined per spatial location $\mathbf{x}$. This index corresponds to the domain $\Omega_{\tilde{\sigma}}$ in which $\mathbf{x}$ resides. Thus, for each frame in the videos $\{G_\sigma\}_{\sigma=0}^{\text{N}_{\text{scales}}-1}$, Eq. (3) is applied. Specifically, the feathering weight we used in Eq. (3) is

$$W(\mathbf{x}) = \frac{f^{\tilde{\sigma}(\mathbf{x})} - R^{\text{map}}(\mathbf{x})}{f^{\tilde{\sigma}(\mathbf{x})} - f^{\tilde{\sigma}(\mathbf{x})+1}} \ . \tag{19}$$

It results in a video for which the spatial transition between domains is seamless. Then, I^{out} appears as

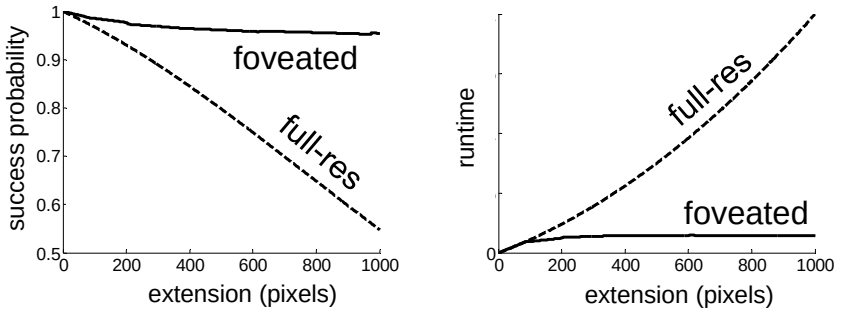


Fig. 13. Theoretical performance of full resolution vs. foveated extrapolation. [Left]: The probability for successful extrapolation without distracting artifacts, as a function of the extrapolation extent. [Right]: Relative runtime as a function of the extrapolation extent. Here, the raw data consists of 50 frames of size 1000×1000 pixels, $z = 10w$, $p = 10^{-6}$, $f = 1/\sqrt{2}$, and $w_x = w_y = w_t = 11$.

having gradual continuous coarsening.

Note that the above algorithm offers a general framework. The extrapolation at each scale-step can be considered as a black box, and thus a wide range of completion algorithms can serve as this "engine." In our implementation we used the patch based method described in Section 3. The next section analyzes the benefits of the foveated approach, comparing it to an approach that uses a patch-based full resolusion (single scale) implementation across the whole extrapolated domain.

5 Theoretical Performance Analysis

This section presents a theoretical analysis of the benefits of foveated extrapolation, in reduction of both artifacts and processing time. We start with a short discussion on the general problem of patch size selection in completion algorithms (Section 5.1). Based on arguments discussed there, we analyze the reduction in artifacts (Section 5.2). Finally, section 5.3 analyzes the reduction in runtime.

5.1 Patch Size Discussion

Patch-based completion methods must decide which patch size to use. It is easier to find matches for small patches. However, accumulating a large number of small patches into a single coherent image region is difficult and prone to artifacts. On the other hand, it is more difficult to find good matches for large ST blocks within the raw data.

There are several ways to tackle this tradeoff. One way is to adapt the patch size to the content. For instance, in regions having predictable motion (e.g., stationary or periodic), larger ST blocks can be used. For highly non-stationary motion and background (e.g., a person walking in front of a painting), smaller blocks should be used. Another way is to use a large database of video/images to extract source patches from. Then, larger patches can be used, since the probability of finding a matching block increases thanks to the database size [16]. Large blocks are also effectively used for periphery completion in the foveated extrapolation algorithm. Contrary to full resolution completion, it is easier to find matching blocks in the foveated approach. The reason is that in foveated extrapolation, the data is blurred, and hence less sensitive to discrepancies stemming from small details.

5.2 Reduction of Artifacts

In this section we analytically show that the probability for distracting artifacts decreases significantly using the foveated extrapolation approach. A precise analysis is difficult. In order to obtain a general impression of the approach's benefits, we make assumptions that are motivated by arguments raised in Section 5.1. Our analysis is based on the assumption that the probability for an artifact-free extrapolation depends on the *number of blocks* synthesized in the extrapolation process. A patch itself does not contain an artifact, as it stems from real data. An artifact is only caused by selecting and merging of blocks.

Let p be the probability that a new synthesized block is successfully added, i.e., it does not introduce a seed of a highly distracting artifact. Then, the probability for a 'successful extrapolation' is

$$P(\text{success}) = p^{N^{\text{blocks}}} \quad , \tag{20}$$

where N^{blocks} is the number of ST blocks that form the extrapolation. Each new block that is searched and merged carries with it the potential of creating a seed for an artifact, which then propagates to a larger artifact as the extrapolation evolves. Hence, it is beneficial to use an algorithm having a smaller N^{blocks}.

The known domain Ω^{raw} is extrapolated by $|\Omega^{\text{extra}}|$ ST pixels. If the entire extrapolation is performed in full spatial resolution, then[6]

$$N^{\text{blocks}}_{\text{fullres}} \approx 2|\Omega^{\text{extra}}|/(w_x w_y w_t) \quad . \tag{21}$$

Thus, the probability for an artifact-free extrapolation is

6. The factor 2 in Eq. 21 is since in each iteration of the core algorithm (Section 2.2), data for approximately half the block is added.

$$P_{\text{fullres}}(\text{success}) \approx p^{2|\Omega^{\text{extra}}|/(w_x w_y w_t)} \ . \quad (22)$$

If a foveated extrapolation approach is taken, the number of blocks is greatly reduced. In the following, we assess $N^{\text{blocks}}_{\text{foveated}}$. Refer to Fig. 11. The domain to be completed, Ω^{extra}, is divided into a few segments $\Omega_0, \Omega_1, ..., \Omega_{(N_{\text{scales}}-1)}$, as described in Section 4.1. At full resolution, the segment Ω_σ requires $\approx 2|\Omega_\sigma|/(w_x w_y w_t)$ blocks to be completed. However, in the foveated approach, the segment is effectively demagnified prior to being filled-in (see Eqs. 16,17). Demagnification is by a lateral factor of $f^{-\sigma}$ along each spatial axis, when accounting for all the recursive reductions. Thus, only

$$N^{\text{blocks}}_{\sigma} \approx 2|\Omega_\sigma| f^{2\sigma}/(w_x w_y w_t) \quad (23)$$

blocks are needed by the foveated approach, per domain. Over the entire extrapolation domain,

$$N^{\text{blocks}}_{\text{foveated}} \approx 2(w_x w_y w_t)^{-1} \sum_{\sigma=0}^{N_{\text{scales}}-1} |\Omega_\sigma| f^{2\sigma} \ . \quad (24)$$

To calculate Eq. (24), we now derive N_{scales} and $|\Omega_\sigma|$. For simplicity, let the video be spatially square, i.e., Ω^{raw} is of spatial size $2w \times 2w$, over a viewing angle

$$\beta = \arctan(w/\kappa z) \ , \quad (25)$$

as shown in Fig. 11. The spatial domain is extrapolated by l pixels in each side. The angles $\alpha_0, \alpha_1, ..., \alpha_{N_{\text{scales}}-1}$ correspond to the boundaries between the scale domains, i.e., based on Eq. (1),

$$\alpha_\sigma = R^{-1}(f^{\sigma+1}) = \frac{\epsilon}{f^{\sigma+1}} - \epsilon \ . \quad (26)$$

The domain Ω_σ is of spatial width (in pixels)

$$l_\sigma = \kappa z[\tan(\alpha_\sigma + \beta) - \tan(\alpha_{\sigma-1} + \beta)] \ . \quad (27)$$

Then, the number of ST pixels in Ω_σ is

$$\begin{aligned} |\Omega_\sigma| &= |\bigcup_{b=0}^{\sigma} \Omega_b| - |\bigcup_{b=0}^{\sigma-1} \Omega_b| \\ &= 4N_{\text{frames}} \left[(w + \sum_{b=0}^{\sigma} l_b)^2 - (w + \sum_{b=0}^{\sigma-1} l_b)^2 \right] \end{aligned} \quad (28)$$

where N_{frames} is the number of video frames. To find N_{scales} we solve $l = \kappa z \tan(\alpha_{N_{\text{scales}-1}} + \beta) - w$ for N_{scales}, using Eq. (26):

$$N_{\text{scales}} = \left\lceil \log_f \left\{ R \left[\arctan\left(\frac{l+w}{\kappa z} \right) - \beta \right] \right\} \right\rceil \ . \quad (29)$$

Now, Eqs. (1,24,27,28), and (29) yield the value

$$P_{\text{foveated}}(\text{success}) \approx p^{N^{\text{blocks}}_{\text{foveated}}} \ . \quad (30)$$

The left graph in Fig. 13 plots $P_{\text{fullres}}(\text{success})$ and $P_{\text{foveated}}(\text{success})$ as a function of the extension width, l. The probability for an artifact-free output is significantly higher when uing the foveated approach.

5.3 Runtime Reduction

Here we analyze the reduction in the runtime complexity. If the extrapolation is performed in full resolution, at each iteration a block of size $w_x w_y w_t$ is completed. There are $2|\Omega^{\text{extra}}|/w_x w_y w_t$ full resolution blocks in Ω^{extra} to complete. Before each block completion, a similar block is searched for in Ω^{raw}. Each comparison requires $\approx w_x w_y w_t/2$ operations. Therefore, the runtime is

$$T^{\text{fullres}} = \mathcal{O}(|\Omega^{\text{raw}}| \cdot |\Omega^{\text{extra}}|) \ . \quad (31)$$

In the foveated approach, each extrapolation spatial domain Ω_σ is processed in a different effective scale. When completing Ω_σ, each block-completion covers a ST region of size $w_x w_y w_t/(2f^{2\sigma})$. Therefore, $N^{\text{blocks}}_{\sigma}$ (Eq. 23) iterations fill Ω_σ. In each iteration, a good fit to a block of size $w_x w_y w_t$ is searched for in a demagnified version of Ω^{raw} of size $|\Omega^{\text{raw}}| f^{2\sigma}$ ST pixels. Again, each comparison requires $w_x w_y w_t/2$ operations. The computation cost for completing Ω_σ is therefore

$$T^{\text{foveated}}_{\sigma} = \mathcal{O}(|\Omega^{\text{raw}}||\Omega^{\sigma}| f^{4\sigma}) \ . \quad (32)$$

Hence, in a domain Ω_σ, the operations are of factor $f^{-4\sigma}$ less than full resolution extrapolation in the same domain. The right graph of Fig. 13 compares between T^{fullres} (Eq. 31) and

$$T^{\text{foveated}} = \sum_{\sigma=0}^{N_{\text{scales}}-1} T^{\text{foveated}}_{\sigma} \ , \quad (33)$$

demonstrating the significant reduction in processing time offered by the foveated approach.

6 Experimental Results

The theoretical results above are indeed evident in practice. The runtime and the chance for artifacts are reduced significantly with foveated extrapolation, compared to the full resolution processes. See Fig. 14 for resulting video frames and for the listed processing time of the *fullres-core* completion algorithm (described in Section 2.2), the *fullres-PAD* algorithm (described in Section 3), and the *foveated* algorithm (described in Section 4). See the supplementary material (or [1]) for videos. Fig. 15 shows additional results for still photographs. We used the *same* (no tuning) parameters in all the video experiments: $w_x = w_y = 11$, $w_t = 5$, $N_{\text{candidate}} = 100$, $N_{\text{top}} = 10$, $\delta_0 = 50$, and $r_{\max} = 20$. For still images $N_{\text{candidate}} = 20$ and of course $w_t = 1$. Given that the screen resolution is set to the size of the output, it is assumed that the viewer sits at a distance equal to about half an extrapolated frame width.

We suggest that video results be viewed on the largest available screen, from approximately this assumed distance, fixating inside the red frame (the raw video). If

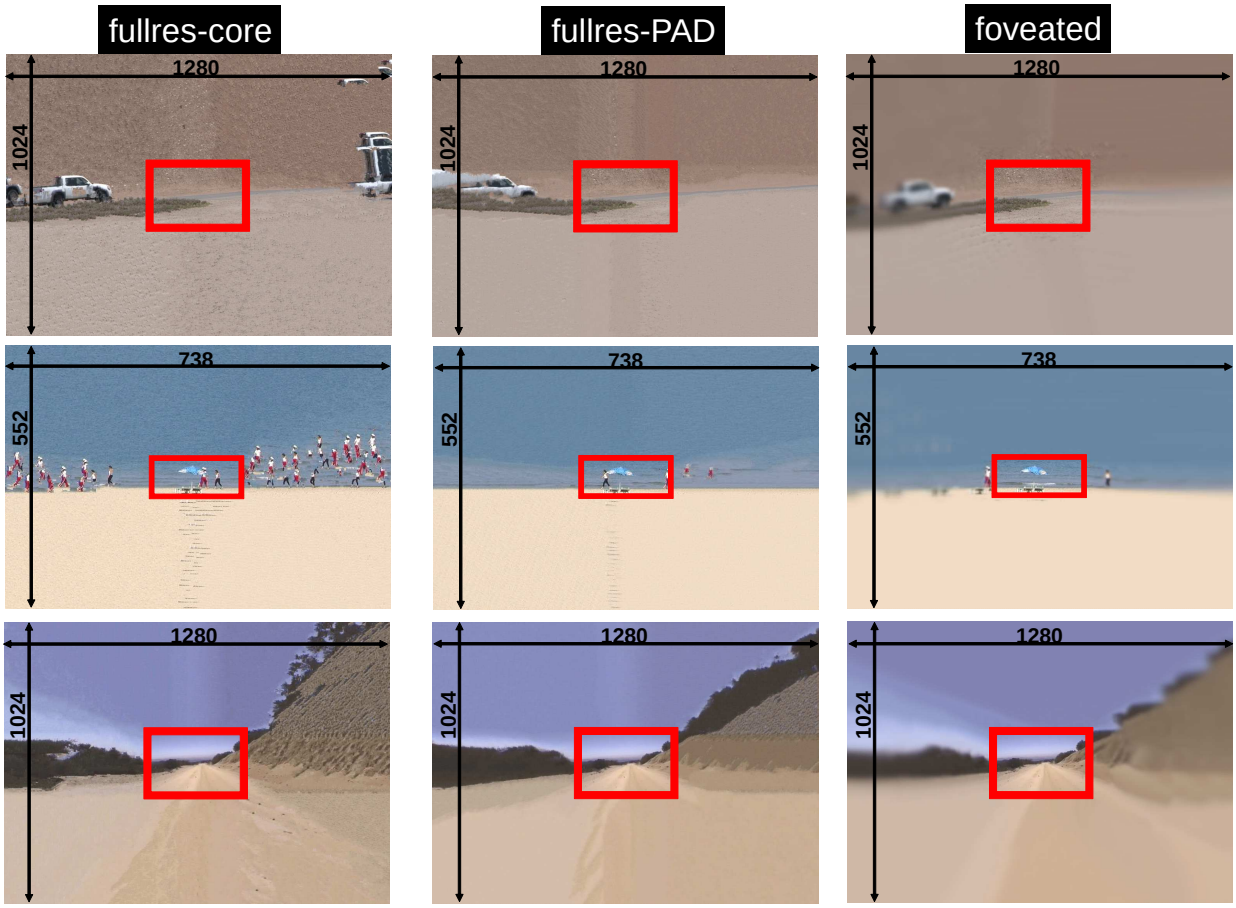


Fig. 14. Sample output video frames, resulting from different algorithms. The videos are available in the supplementary material (or [1]). The input domain is marked by the red frames. [Left]: Fullres-core completion (Sec 2.2). [Center]: Our *PAD* full resolution algorithm rejecting peripheral distraction (Section 3). [Right]: Foveated extrapolation (Section 4). Note the significant reduction in distracting peripheral artifacts. The processing time was reduced by a factor of 30-50. The video corresponding to the lower row demonstrates a failure case, where the periphery exhibits a distracting wavy motion. In this case, a completion engine that incorporates camera motion analysis may help. The raw video used in the 2^{nd} row is from [42].

you fixate on the coarsened peripheral video, peripheral blur is apparent and soft artifacts are sometimes found. However, when watching the running video at close range and fixating on the region of the original data, a highly enhanced visual experience is obtained (relative to raw display) without soft artifacts and peripheral blur being noticed.

The videos and images clearly show that distracting artifacts are significantly less apparent using the foveated approach, particularly compared to full resolution results of the core algorithm. Moreover, processing time is dramatically reduced (by a factor of 30-50) using the foveated approach.

The algorithms were applied to different types of scenes. Specifically, they were applied to images from the LabelMe database used in [27], as demonstrated in Fig. 15. This database includes outdoor images divided into eight categories. Some of the categories (e.g., 'open country', 'coast', and 'mountains') include natural scenery. Other categories (e.g., 'inside city' and 'tall buildings') include urban scenes. Both the fullres-core algorithm and the foveated algorithm suggest better results for natural scenery and less acceptable results for urban scenes. Nevertheless, the foveated results almost always appear more realistic and include fewer artifacts than results of the fullres-core algorithm. See Section 7.1 for an empirical study of the density of artifacts in these images.

The video corresponding to the lower row in Fig. 14 demonstrates a failure case. There, the periphery exhibits a distracting wave-like motion in all three methods. In this case using a completion engine that incorporate motion analysis may help. See Section 8 for additional discussion on failure cases and possible improvements.

7 User Studies of Visual Quality

While runtime is objectively measured, quality is subjective and hard to quantify. The following user studies consider only the *quality* and discard the processing-time

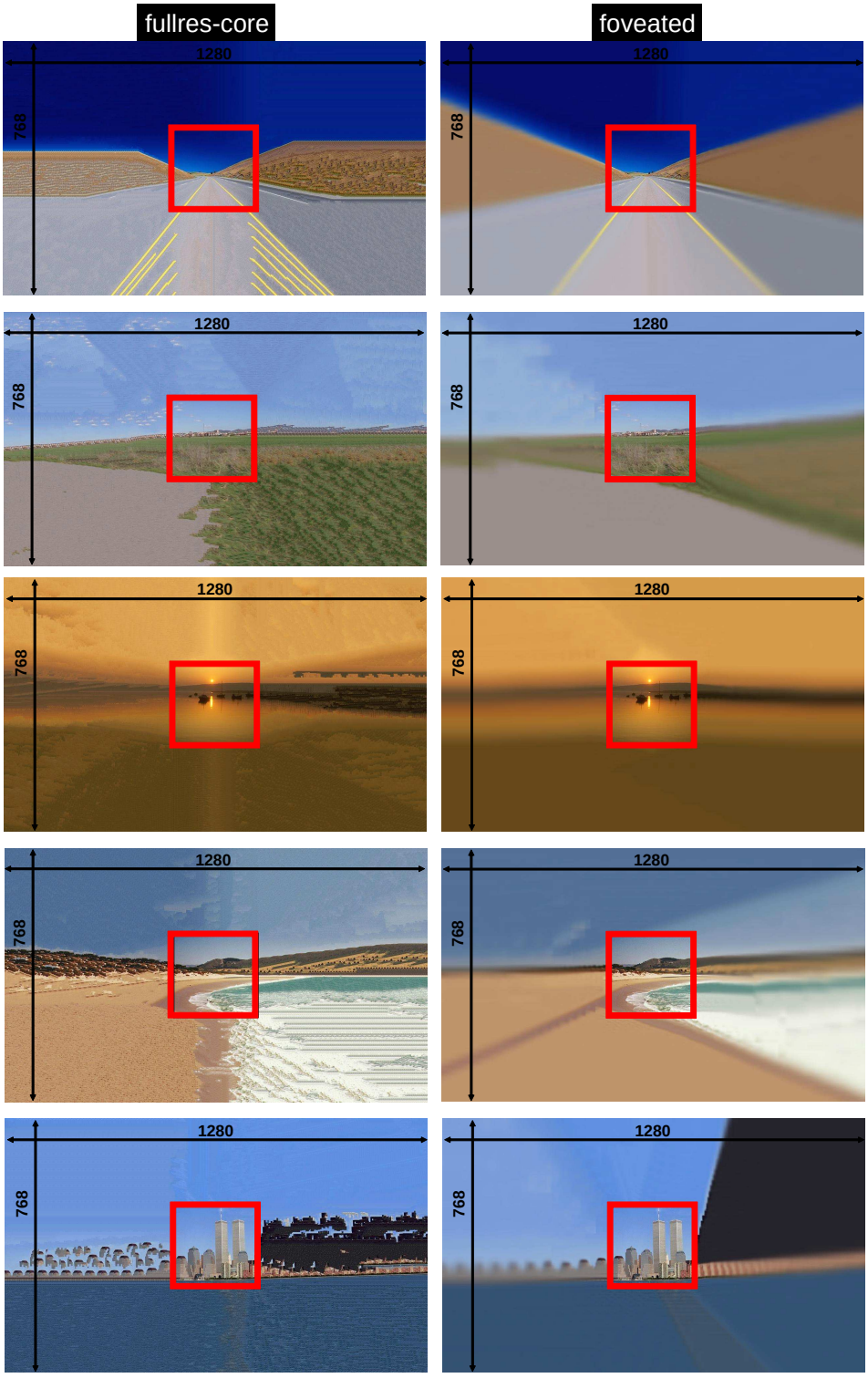


Fig. 15. Results on images from the LabelMe database [35]. The input domain is marked by the red frames. [Left]: Fullres-core completion (Sec 2.2). [Right]: Foveated extrapolation (Section 4). The processing time was reduced by a factor of 10.

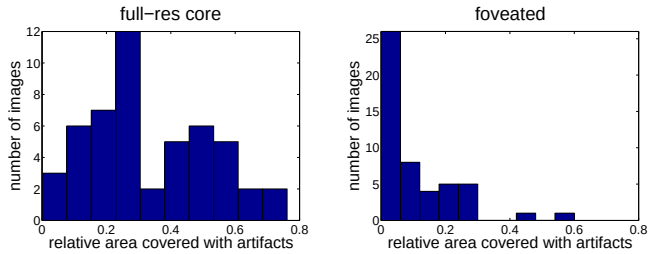


Fig. 16. Image extrapolation quality. Fifty images were extrapolated from 256×256 to 1280×768. The histograms above summarize the relative area of extrapolation marked subjectively as artifacts. [Left]: The histogram corresponding to the results of the fullres-core algorithm. [Right]: The histogram corresponding to the results of the foveated algorithm.

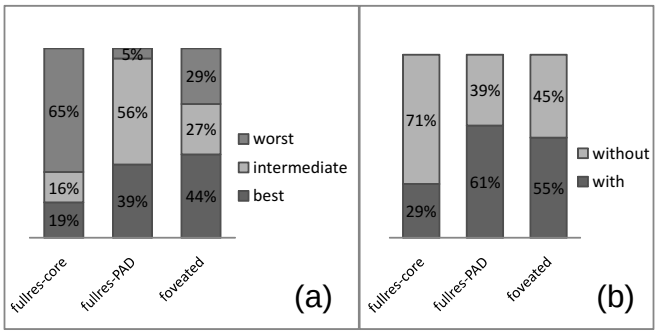


Fig. 17. A user study for video display options. (a) Percent of each rating ('best', 'intermediate', 'worst') for each method. (b) For each result, the participants had to state whether they preferred viewing the video clip either 'with' extrapolation, or using a standard black periphery ('without').

aspect. We performed two subjective examinations. The first dealt with density of artifacts in extrapolated images and was performed by two people. The second and more intensive study dealt with overall reaction to different video display options. The latter study was based on 25 people. In both studies, the subjects were unfamiliar with the algorithms being tested or had no idea which results correspond to the algorithms suggested in this paper.

7.1 Density of Artifacts in Extrapolated Images

This experiment used 50 images from the LabelMe dataset mentioned above, covering all 8 scene categories. Each of those images was extrapolated both by the fullres-core algorithm and the foveated algorithm. Two subjects, aged 10 and 37, were asked to mark regions in the extrapolated images that they found to be unnatural and that distracted their attention from the raw frame. Each subject marked 25 pairs, in random order. Fig. 16 shows histograms summarizing the relative area marked as artifacts, for each of the two algorithms. According to this marking, the foveated algorithm introduced significantly fewer artifacts compared to the fullres-core algorithm.

7.2 A User Study for Video Display Options

Twenty-five people aged 6 to 67 with normal or corrected-to-normal vision were asked to view the video clips corresponding to the nine results described in Fig.14. Each participant sat in front of a 26″ display, at a distance of $\approx$ 13″. The participants were asked to rank each set of results (corresponding to a row in Fig.14) from 'best' to 'worst.' The results inside such a set and between the sets were randomly permuted so that each participant saw the results in a different order.

Overall there were 75 experiments (3 input scenes $\times$ 25 participants). Fig. 17a summarizes them. The foveated results were ranked as 'best' in far more experiments than the fullres-core results, and slightly more than the fullres-PAD results. The fullres-core results were ranked as 'worst' in the highest number of experiments.

Data analysis shows that the ratings given by any participant were self-consistent: one method was preferred by a participant, for all scenes.[7] However, per specific video scene, a similar pattern to that in Fig. 17a is observed. This consistency implies that preferences are viewer-dependent and *not* video dependent.

The participants who preferred the fullres-core results reported that they tolerated the artifacts, as they resemble interrupts they are familiar with. We hypothesize that they refer to artifacts commonly caused by poor communication or brute compression.

Afterwards, the participants were asked to state for each video scene, whether they preferred watching it with or without the additional extrapolation, as plotted in Fig. 17b. Here too, our approach demonstrates significant superiority (Student's t-test, p-value< 0.01). In 71% of the experiments, the participants preferred a black background rather than watching the results of the fullres-core extrapolation. The trend was reversed by our algorithms (fullres-PAD and foveated). Then, the extrapolation was preferred over a black background in $\approx 60\%$ of the experiments. These preferences also tended to be consistent per user, across scenes.

Consequently, such extrapolation can be an optional feature of a display instrument or application. This display feature can be turned on/off by the user. This user-study implies that the number of viewers who may choose to turn this feature *on*, increases notably, thanks to the algorithms suggested in this paper.

8 Discussion

8.1 Potential Display Designs

The sections above indicate that displaying very wide, yet blurred video extensions can be beneficial for a significant percentage of viewers, while facilitating

7. This preference was expressed without the participants being aware of this consistency (not knowing which choice corresponds to any method).

faster processing. One question which now arises is, which display hardware might be needed to show such extrapolated videos efficiently? A wide array of possible configurations can deliver the required display. Here, we describe three options, in the context of applications.

Standard displays. Some applications use standard display hardware without exploiting the high definition of the screen. Examples for such applications are viewing of Internet streaming video and viewing of short amateurish videos, taken e.g., by digital cameras integrated in cellular phones. Such videos are usually of much lower effective resolution than the screen. Thus, using a video player window of a size that corresponds to the video leaves much of the screen area unused, filled with unrelated icons, etc. This area could be better used for extrapolating the scene. Rather than cropping or stretching videos, the native video is shown, while the complementary screen regions are used for extrapolation.

Distorted projection displays. We believe that creating displays having large, yet blurred peripheries should be easier using a projector. Consider projection through wide-angle optics, such as a fisheye lens or a curved mirror, as used in omnidirectional cameras [44]. Such optical systems distort the images, particularly at the periphery: the magnification at the center is not the same as the peripheral magnification. Moreover, the distortion is known. Thus it can be compensated for and actually exploited using the digital algorithm that creates the peripheral content.

The implementation can thus be based on tailored distortion optics. It may actually ease the digital implementation: the need for digital magnification expressed in Eq. (18) is alleviated, since this fixed spatially-varying magnification is done optically using the projection optics. Thanks to the optical magnification, the projector LCD or DLP chip need not be significantly larger than the chip used for the raw FOV: the peripheral LCD pixels are converted to larger projected areas, as illustrated in Fig. 18. Furthermore, optical aberrations tend to increase with off-axis angles. Aberrations create image blur. Thus, optical engineers typically take pains to counter the tendency of peripheral blur. However, for foveated extrapolation, the natural tendency of projected peripheral blur can be a blessing, rather than a curse. Such hardware can be used for projecting on flat screens or on immersive hemispherical displays.

Microdisplay goggles. Foveated extrapolation should fit nicely into microdisplay goggles, which write the image directly on the viewer's retina [33], using laser raster scan. There, the sampling density and spot size of the scanning laser correspond to the high-resolution of the fovea. To display the extrapolation, the peripheral regions of the retina should be illuminated as well. This peripheral illumination can have coarser resolution. This can be achieved, for instance by distorting the scanning beam using proper optics, as in the case of projectors.

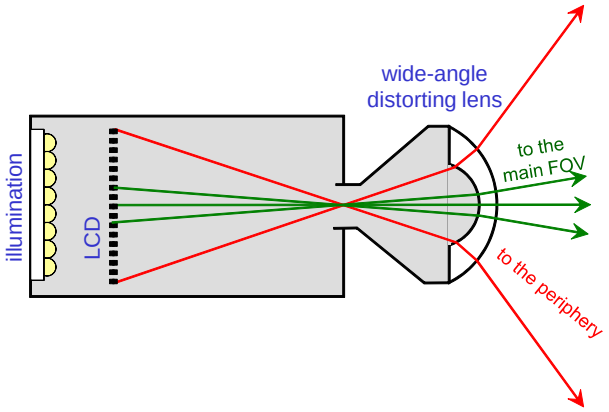


Fig. 18. A scheme of a distorted projection display. Given an extrapolated video, the green rays demonstrate the projection of the raw FOV, while the red rays demonstrate the projection of the peripheral extrapolation results. Due to the wide-angle distorting lens, the peripheral LCD region is much more strongly magnified optically than the central FOV (replacing the digital magnification in Eq.18). The projector LCD or DLP chip need not be significantly larger than the chip used for the raw FOV.

8.2 General Discussion

Video extrapolation may create an enhanced experience when viewing both HD movies and low resolution video, using standard displays, microdisplays, or tailored display designs such as distorted projectors. To the best of our knowledge, this paper reports the first work addressing the ambitious task of very *wide* extrapolation of video, and the first to provide a feasible solution tailored to this task. The foveated method allows both significant runtime reduction (one or two orders of magnitude) and significantly fewer potential seeds for distracting artifacts. These two advantages are demonstrated both theoretically and in practice. We stress that the foveated algorithm introduced in Section 4 is a general framework that uses a completion "engine" as a block that can be replaced with a wide range of video completion algorithms.

There exist many possibilities for further improvement of results and processing time. Although the results of the user study described in section 7 are encouraging, we believe the percent of "pro-extrapolation" viewers is likely to increase significantly with future improvements. One shortcoming of the current implementation is that temporal inconsistencies sometimes result (see the video corresponding to the bottom row in Fig. 14). To improve the method, it is worth considering incorporating semantic analysis (e.g., tracking, segmentation, mosaicing), as suggested in some recent video completion methods (e.g., [20], [21], [22], [28]), or to incorporate camera and

object motion estimation methods (e.g., [4], [17], [29], [39], [45]) in the current engine. The Matlab runtime, although significantly reduced, is still a burden (In the current Matlab implementation, running on a Core 2 Duo E8400, 3.00 GHz Intel CPU with 4.00 GB RAM, wide extrapolations such as those described in the experiments section may take up to several minutes per frame). Incorporating motion analysis methods will surely lead to more efficient algorithms. In addition, we consider a parallel computing GPGPU solution.

The current work assumes that the viewer fixates always on the raw data. However, it is worth using computerized visual attention models (e.g., [3], [19], [30]), which will estimate a few possible eye fixation locations. Then, the resolution map guiding the foveated extrapolation process can be adaptive. Consider cases where the eyes are likely to leave the raw data (as in the case of the vehicle driving out of the frame). The attention-guided extrapolation may provide sharp results in that area, although it is in the peripheral part of the screen.

Following the majority of completion algorithms, the method suggested here uses only the input image/video as the completion source. As a result, it may fail when the appropriate data for completing the missing parts is not available in this source. For example, extrapolating a close-up on part of a face cannot introduce the other missing parts using our current formulation. Following the work of [16], we believe that this may be solved using a database of movies, from which source blocks can be selected. The foveated extrapolation approach makes this easier, as only coarse, iconic versions of the videos may suffice to complete the distant periphery.

Acknowledgments

This research was supported by Philips Consumer Lifestyle, Eindhoven, The Netherlands, and by the Israeli Science Foundation (grant 1031/08). We thank Ad de Vann and Guido Volleberg for useful discussions. Yoav Schechner is a Landau Fellow - supported by the Taub Foundation, and an Alon Fellow. This work was conducted in the Ollendorff Minerva Center. Minerva is funded through the BMBF. We thank Ben Herzberg and Roee Sulimarski for their help in preparing the supplementary video.

References

[1] http://www.cs.technion.ac.il/~tammya/extrapolation.html.

[2] http://www.televisionpoint.com/news/newsfullstory.php?id=1124298851, 2005.

[3] T. Avraham and M. Lindenbaum. Esaliency (extended saliency): Meaningful attention based on stochastic image modeling. *IEEE Tran. PAMI*, 32(4):693–708, 2010.

[4] A. Beric, B. van der Waal, R. Sethuraman, and G. de Haan. Low-bandwidth dynamic aspect ratio region-based motion estimation. In *Proc. IEEE Workshop on Signal Processig System*, pages 393–398, 2005.

[5] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. *ACM Tran. Graphics, SIGGRAPH Proceedings*, pages 417–424, 2000.

[6] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Tran. Comm.*, 31:532–540, 1983.

[7] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Tran. Graphics*, 2:217–236, 1983.

[8] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based inpainting. *IEEE Tran. IP*, 13:1200–1212, 2004.

[9] I. Drori, D. Cohen-Or, and H. Yeshurun. Fragment-based image completion. *ACM Tran. Graphics*, 22:303 – 312, 2003.

[10] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. *ACM Tran. Graphics, SIGGRAPH Proceedings*, pages 341–346, 2001.

[11] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. *Proc. IEEE ICCV*, pages 1033–1038, 1999.

[12] W. S. Geisler and J. S. Perry. A real-time foveated multi-resolution system for low-bandwidth video communication. *Proc. SPIE Human Vision and Electronic Imaging*, pages 294–305, 1998.

[13] W. S. Geisler, J. S. Perry, and J. Najemnik. Visual search: The role of peripheral information measured using gaze-contingent displays. *Journal of Vision*, 6:858–873, 2006.

[14] A. Ghosh, M. Trentacoste, H. Seetzen, and W. Heidrich. Real illumination from virtual environments. *Proc. Eurographics Symposium on Rendering*, pages 243–252, 2005.

[15] C. Han, E. Risser, R. Ramamoorthi, and E. Grinspun. Multiscale texture synthesis. *ACM Tran. Graphics*, 27:51–58, 2008.

[16] J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Tran. Graphics*, 26:87–94, 2007.

[17] W. Hong, Y. Kim, T. Oh, and S. Ko. A partial norm based early rejection algorithm for fast motion estimation. *IEICE Tran.*, 88-A(3):626–632, 2005.

[18] D. P. Huttenlocher, G. A. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Tran. PAMI*, 15(9):850–863, 1993.

[19] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Tran. PAMI*, 20(11):1254–1259, 1998.

[20] J. Jia, T. P. Wu, Y. W. Tai, and C.K. Tang. Video repairing: Inference of foreground and background under severe occlusion. *Proc. IEEE CVPR*, pages 364–371, 2004.

[21] Y. Jia, S. Hu, and R. R. Martin. Video completion using tracking and fragment merging. *The Visual Computer*, 21:601–610, 2005.

[22] A. C. Kokaram, B. Collis, and S. Robinson. Automated rig removal with bayesian motion interpolation. *IEE J. on Vision, Image and Signal Proccesing*, 152:407– 414, 2005.

[23] N. Komodakis and G. Tziritas. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Tran. IP*, 16(11):2649–2661, 2007.

[24] S. Kumar, M. Biswas, J. Belongie, and T. Q. Nguyen. Spatio-temporal texture synthesis and image inpainting for video applications. *Proc. IEEE ICIP*, 2:85–88, 2005.

[25] S. Masnou and J. M. Morel. Level lines based disocclusion. *Proc. IEEE ICIP*, 3:259–263, 1998.

[26] Y. Matsushita, E. Ofek, W. G. X. Tang, and H. Shum. Full-frame video stabilization with motion inpainting. *IEEE Tran. PAMI*, 28:1150–63, 2006.

[27] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[28] K. A. Patwardhan, G. Sapiro, and M. Bertalmio. Video inpainting of occluding and occluded objects. *Proc. IEEE ICIP*, 2:69–72, 2005.

[29] W. Qu and D. Schonfeld. Low-bandwidth dynamic aspect ratio region-based motion estimation. In *IEEE Tran. IP*, volume 17, pages 1721–1726, 2008.

[30] U. Rajashekar, I. van der Linde, A. C. Bovik, and L. K. Cormack. Gaffe: A gaze-attentive fixation finding engine. *IEEE Tran. IP*, 17(4):564–573, 2008.

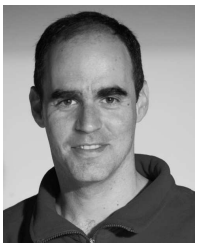
[31] A. G. Rempel, W. Heidrich, H. Li, and R. Mantiuk. Video viewing preferences for HDR displays under varying ambient illumination. *ACM Proc. 6th Sympos. Applied Perception in Graphics and Visualization*, pages 45–52, 2009.

[32] J. G. Robson and N. Graham. Probability summation and regional variation in contrast sensitivity across the visual field. *Vision Research*, 21:409–418, 1981.

[33] M. Rose. Microdisplays: Coming soon to an eye near you? *Photonics Spectra*, pages 68–69, Sep 2008.

[34] M. Rubinstein, A. Shamir, and S. Avidan. Improved seam carving for video retargeting. *ACM Tran. Graphics*, 27(3):1–9, 2008.
[35] B. C. Russell and A. Torralba. Labelme: a database and web-based tool for image annotation. *International Journal of Computer vision*, 77:157–173, 2008.
[36] M. P. Sampat, Z. Wang, S. Gupta, A. Conrad Bovik, and M. K. Markey. Complex wavelet structural similarity: A new image similarity index. *IEEE Tran. IP*, 18(11):2385–2401, 2009.
[37] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani. Summarizing visual data using bidirectional similarity. In *Proc. IEEE CVPR*, 2008.
[38] J. Sun, L. Yuan, J. Jia, and H. Shum. Image completion with structure propagation. *ACM Tran. Graphics*, 24:861 – 868, 2005.
[39] S. Treetasanatavorn, J. Heuer, U. Rauschenbach, K. Illgner, and A. Kaup. Temporal video segmentation using global motion estimation and discrete curve evolution. *Proc. IEEE ICIP*, pages 385–388, 2004.
[40] Z. Wang and A. C. Bovik. Embedded foveation image coding. *IEEE Tran. IP*, 10:1397–1410, 2001.
[41] L. Wei. Texture synthesis from multiple sources. *ACM Tran. Graphics*, 22:1–1, 2003.
[42] Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *IEEE Tran. PAMI*, 29:463–476, 2007.
[43] M. Wilczkowiak, G. J. Brostow, B. Tordoff, and R. Cipolla. Hole filling through photomontage. *Proc. BMVC*, pages 492–501, 2005.
[44] Y. Yagi. Omnidirectional sensing and its applications. *IEICE Tran. Inform. Systems*, E82-D:568–579, 1999.
[45] H. Zhang, L. Shao, and K. K. Wong. Motion recovery for uncalibrated turntable sequences using silhouettes and a single point. *Proc. Inter. Conf. Advanced Concepts for Intelligent Vision Systems*, pages 796–807, 2008.

Tamar Avraham received her B.Sc. (summa cum laude, 1996) and her Ph.D. (2008) in the computer science department at the Technion–Israel Institute of Technology. She worked for several years in the industry (Fibronics Ltd. and CMT Medical Technologies Ltd.) as a software engineer, a project manager, and a product manager. Currently, Tamar is a research fellow at the Technion Research & Development Foundation. Her main research interest is in computer vision, with a focus on scene analysis, statistics of natural images, visual attention, and computer graphics applications that incorporate image and video analysis.



Yoav Y. Schechner Yoav Y. Schechner received his BA and MSc degrees in physics and PhD in electrical engineering from the Technion - Israel Institute of Technology in 1990,1996, and 1999, respectively. During the years 2000 to 2002 Yoav was a research scientist at the computer science department in Columbia University. Since 2002, he is a faculty member at the department of Electrical Engineering of the Technion, where he heads the Hybrid Imaging Lab. His research is focused on computer vision, the use of optics and physics in imaging and computer vision, and on multi-modal sensing. He was the recipient of the Wolf Foundation Award for Graduate Students in 1994, the Gutwirth Special Distinction Fellowship in 1995, the Israeli Ministry of Science (Eshkol) Distinction Fellowship and the Ollendorff Award in 1998, the Schwartz Foundation Award in 1999 and the Morin Fellowship in 2000-2002. He is now a Landau Fellow - supported by the Taub Foundation, and an Alon Fellow. He received the Klein Research Award in 2006, the Henry Taub Prize for Academic Excellence in 2008 and Outstanding Reviewer Awards in IEEE CVPR 2007, IEEE ICCV 2007, ECCV 2008 and IEEE ICCV 2009.