Analysis and Visualization of Temporal Variations in Video

by

Michael Rubinstein

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

February 2014

Author: _____

Department of Electrical Engineering and Computer Science
November 27, 2013

Certified by: _____

Professor William T. Freeman
Thesis Supervisor

Accepted by: _____

Professor Leslie A. Kolodziejski,
Chairmain, Department Committee on Graduate Theses

To my parents,

and to my wife and daughter

Analysis and Visualization of Temporal Variations in Video

by

Michael Rubinstein

Submitted to the Department of Electrical Engineering
and Computer Science on November 27, 2013
in Partial Fulfillment of the Requirements for the Degree
of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Our world is constantly changing, and it is important for us to understand how our environment changes and evolves over time. A common method for capturing and communicating such changes is imagery – whether captured by consumer cameras, microscopes or satellites, images and videos provide an invaluable source of information about the time-varying nature of our world. Due to the great progress in digital photography, such images and videos are now widespread and easy to capture, yet computational models and tools for understanding and analyzing time-varying processes and trends in visual data are scarce and undeveloped.

In this dissertation, we propose new computational techniques to efficiently represent, analyze and visualize both short-term and long-term temporal variation in videos and image sequences. Small-amplitude changes that are difficult or impossible to see with the naked eye, such as variation in human skin color due to blood circulation and small mechanical movements, can be extracted for further analysis, or exaggerated to become visible to an observer. Our techniques can also attenuate motions and changes to remove variation that distracts from the main temporal events of interest.

The main contribution of this thesis is in advancing our knowledge on how to process spatiotemporal imagery and extract information that may not be immediately seen, so as to better understand our dynamic world through images and videos.

Thesis Supervisor: William T. Freeman
Title: Professor of Electrical Engineering and Computer Science

Thesis Committee: Professor Frédo Durand, Dr. Ce Liu (Microsoft Research), Dr. Richard Szeliski (Microsoft Research)

Acknowledgments

I would like to first thank my advisor, Prof. William T. Freeman, for his support and guidance, and Prof. Frédo Durand and Dr. Ce Liu, with whom I have collaborated closely during my PhD. I would also like to thank my other collaborators in this work: Prof. John Guttag, Dr. Peter Sand, Dr. Eugene Shih, and MIT students Neal Wadhwa and Hao-Yu Wu. During two summer internships at Microsoft Research, I also had the pleasure working with Dr. Ce Liu, Dr. Johannes Kopf, and Dr. Armand Joulin, on a separate project that is not part of this thesis [43, 42].

I would like to thank Dr. Richard Szeliski for insightful and inspiring discussions, and for his helpful feedback on this dissertation. I would also like to thank Dr. Sylvain Paris, Steve Lewin-Berlin, Guha Balakrishnan, and Dr. Deqing Sun, for fruitful discussions that contributed to this work.

I thank Adam LeWinter and Matthew Kennedy from the Extreme Ice Survey for providing us with their time-lapse videos of glaciers, and Dr. Donna Brezinski, Dr. Karen McAlmon, and the Winchester Hospital staff, for helping us collect videos of newborn babies. I also thank Justin Chen from the MIT Civil Engineering Department, for his help with the controlled metal structure experiment that is discussed in Chapter 3.

I would like to acknowledge fellowships and grants that supported my research during my PhD: Mathworks Fellowhip (2010), NVIDIA Graduate Fellowship (2011), Microsoft Research PhD Fellowship (2012-13), NSF CGV-1111415 (Analyzing Images Through Time), and Quanta Computer.

Finally, I would like to thank my parents and my wife, for years of support and encouragement; and my daughter, Danielle, for helping me see things in perspective, and for constantly reminding me what is truly important in life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The only reason for time is so that everything doesn't happen at once.

–Albert Einstein

Modern photography provides us with useful tools to capture physical phenomena occurring over different *scales of time* (Figure 1.1). At one end of the spectrum, high-speed imagery now supports frame rates of 6 MHz (6 million frames per second), allowing the high quality capture of ultrafast events such as shock waves and neural activity [20]. At the other end of the spectrum, time-lapse sequences can reveal long-term processes spanning decades, such as the evolution of cities, melting of glaciers, and deforestation, and have even recently become available on a planetary scale [12]. However, methods to automatically identify and analyze physical processes or trends in visual data are still in their infancy [17]

This thesis is comprised of several projects I explored during my PhD, which focus on analyzing and manipulating temporal variations in video and image sequences, in order to facilitate the analysis of temporal phenomena captured by imagery, and reveal interesting temporal signals and processes that may not be easily visible in the original data. Here we give a high-level overview of these projects. The details of the proposed techniques and review of related literature are given in the chapters to follow. In the rest of the thesis, I will refer to changes in intensities recorded in video and images over time (caused by change in color, or motion) as "temporal variation", or simply "variation" for brevity.

Removing distracting variation. First, we explored the problem of *removing* motions and changes that distract from the main temporal signals of interest. This is especially useful for time-lapse sequences that are often used for long-period medical and scientific analysis, where dynamic scenes are captured over long periods of time. When day- or even year-long events are condensed into minutes or seconds, the pixels can be temporally inconsistent due to the

significant time aliasing. Such aliasing effects take the form of objects suddenly appearing and disappearing, or illumination changing rapidly between consecutive frames, making the long-term processes in those sequences difficult to view or further analyze. For example, we can more easily measure the growth of outdoor plants by re-synthesizing the video to suppress the leaves waving in the wind.

We designed a video processing system which treats short-term visual changes as noise, long-term changes as signal, and re-renders a video to reveal the underlying long-term events [44] (Chapter 2). We call this technique "motion denoising" in analogy to image denoising (the process of removing sensor noise added during image capture). The result is an automatic decomposition of the original video into short- and long-term motion components. We show that naive temporal filtering approaches are often incapable of achieving this task, and present a novel computational approach to denoise motion without explicit motion analysis, making it applicable to diverse videos containing highly involved dynamics common to long-term imagery.

Magnifying imperceptible variation. In other cases, one may want to *magnify* motions and changes that are too subtle to be seen by the naked eye. For example, human skin color varies slightly with blood circulation (Figure 3.1). This variation, while invisible to the naked eye, can

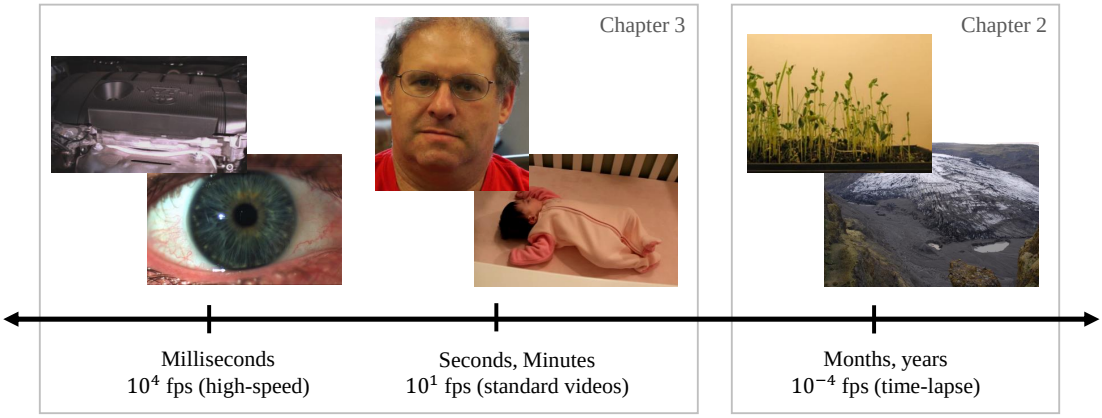


Figure 1.1: Timescales in imagery. High-speed videos (left) capture short-term, fast motions, such as vibration of engines and small eye movements. Normal-rate videos (middle) can capture physiological functions, such as heart rate and respiratory motions. Time-lapse sequences (right) depict long-term physical processes, such as the growth of plants and melting of glaciers. The listed frame rates (Frames Per Second; fps) are only representative rates and can vary considerably between videos in each category.

be exploited to extract pulse rate and reveal spatial blood flow patterns. Similarly, motion with low spatial amplitude, while hard or impossible for humans to see, can be magnified to reveal interesting mechanical behavior.

We proposed efficient methods that combine spatial and temporal processing to emphasize subtle temporal changes in videos [59, 45, 55] (Chapter 3). These methods use an *Eulerian* specification of the changes in the scene, analyzing and amplifying the variation over time at fixed locations in space (pixels). The first method we proposed, which we call the *linear method*, takes a standard video sequence as input, and applies spatial decomposition followed by temporal filtering to the frames. The resulting temporal signal is then amplified to reveal hidden information (Section 3.2). This temporal filtering approach can also reveal low-amplitude spatial motion, and we provide a mathematical analysis that explains how the temporal intensity signal interplays with spatial motion in videos, which relies on a *linear* approximation related to the brightness constancy assumption used in traditional optical flow formulations. This approximation (and thus the method) only applies to very small motions, but those are exactly the type of motions we want to amplify.

The linear method to amplify motions is simple and fast, but suffers from two main drawbacks. Namely, noise gets amplified linearly with the amplification, and the approximation to the amplified motion breaks down quickly for high spatial frequencies and large motions. To counter these issues, we proposed a better approach to process small motions in videos, where we replace the linear approximation with a localized Fourier decomposition using complex-valued image pyramids [55] (Section 3.3). The phase variations of the coefficients of these pyramids over time correspond to motion, and can be temporally processed and modified to manipulate the motion. In comparison to the linear method, this *phase-based* technique has a higher computational overhead, but can support larger amplification of the motion with fewer artifacts and less noise. This further extends the regime of low-amplitude physical phenomena that can be analyzed and visualized by *Eulerian* approaches.

Visualization. We produced visualizations that suppress or highlight different patterns of temporal variation by synthesizing videos with smaller or larger changes (Chapters 2, 3). For small-amplitude variations, we also explored interactive user interfaces and visualization tools to assist the user in exploring temporal signals in videos (Section 3.4). We built a prototype application that can amplify and reveal micro changes from video streams in real-time and show phenomena occurring at temporal frequencies selected by the user. This application can run

on modern laptops and tablets, essentially turning those devices into "microscopes" for minuscule visual changes. We can also produce static image visualizations summarizing the temporal frequency content in a video.

This thesis is largely based on work that appeared in the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [44], and ACM Transactions on Graphics (Proceedings SIGGRAPH 2012 and 2013) [59, 55]. All the the accompanying materials referenced here, including software, videos and demos, are available for the research community through the thesis web page: http://people.csail.mit.edu/mrub/PhDThesis.

Chapter 2

Motion Denoising

Motions can occur over both short and long time scales. In this chapter, we introduce *motion denoising*, a video processing technique that treats short-term visual changes as noise, long-term changes as signal, and re-renders a video to reveal the underlying long-term events. We demonstrate motion denoising for time-lapse videos. One of the characteristics of traditional time-lapse imagery is stylized jerkiness, where short-term changes in the scene appear as small and annoying jitters in the video, often obfuscating the underlying (long-term) temporal events of interest. We apply motion denoising for resynthesizing time-lapse videos showing the long-term evolution of a scene with jerky short-term changes removed. We show that existing filtering approaches are often incapable of achieving this task, and present a novel computational approach to denoise motion without explicit motion analysis. We demonstrate promising experimental results on a set of challenging time-lapse sequences.

■ 2.1 Introduction

Randomness appears almost everywhere in the visual world. During the imaging process, for example, randomness occurs in capturing the brightness of the light going into a camera. As a result, we often see noise in images captured by CCD cameras. As image noise is mostly unwanted, a large number of noise removal approaches have been developed to recover the underlying signal in the presence of noise.

Randomness also appears in the form of motion. Plants sprout from soil in an unplanned order; leaves move arbitrarily under wind; clouds spread and gather; the surface of our planet changes over seasons with spotted decorations of snow, rivers, and flowers.

As much as image noise is often unwanted, motion randomness can also be undesired. For example, traditional time-lapse sequences are often characterized by small and annoying jitters

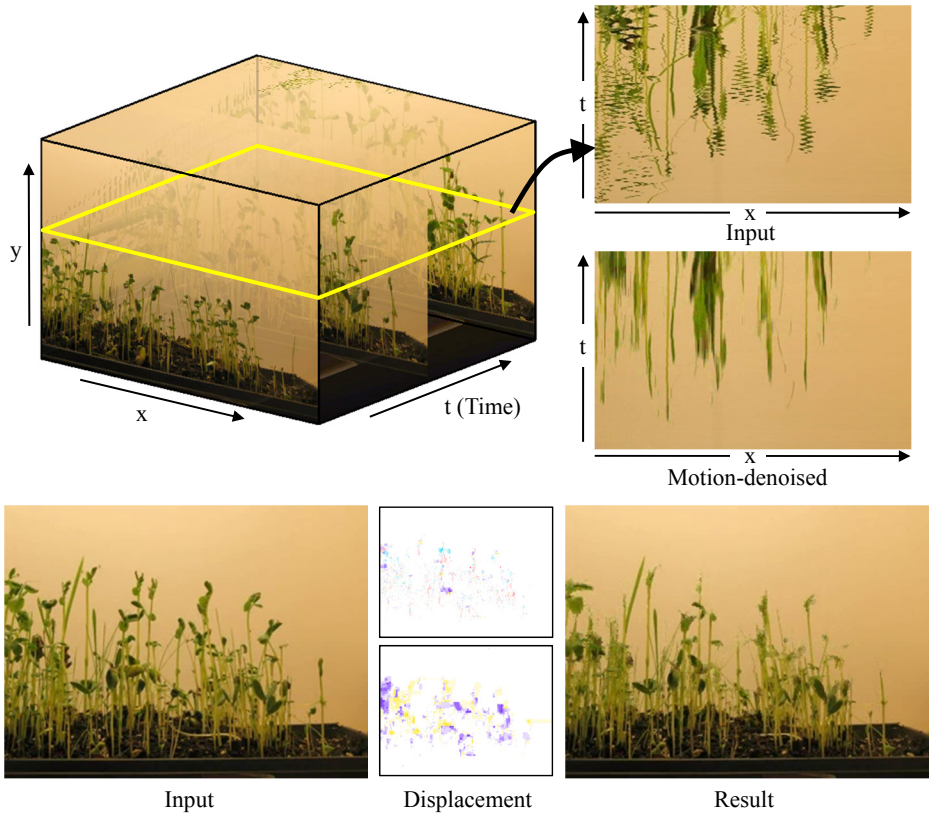


Figure 2.1: A time-lapse video of plants growing (*sprouts*). XT slices of the video volumes are shown for the input sequence and for the result of our motion denoising algorithm (top right). The motion-denoised sequence is generated by spatiotemporal rearrangement of the pixels in the input sequence (bottom center; spatial and temporal displacement on top and bottom, respectively, following the color coding in Figure 2.5). Our algorithm solves for a displacement field that maintains the long-term events in the video while removing the short-term, noisy motions. The full sequence and result are available in the accompanying material.

resulting from short-term changes in shape, lighting, viewpoint, and object position, which obfuscate the underlying long-term events depicted in a scene. Atmospheric and heat turbulence often show up as short-term, small and irregular motions in videos of far-away scenes.

It is therefore important to remove random, temporally inconsistent motion. We want to design a video processing system that removes temporal jitters and inconsistencies in an input video, and generate a temporally smooth video as if randomness never appeared. We call this technique *motion denoising* in analogy to image denoising. Since visual events are decomposed

into slow-varying and fast-changing components in motion denoising, such technique can be useful for time-lapse photography, which is widely used in the movie industry, especially for documentary movies, but has recently become prevalent among personal users as well. It can also assist long-period medical and scientific analysis. For the rest of the chapter we will refer to motion denoising and its induced motion decomposition interchangeably.

Motion denoising is by no means a trivial problem. Previous work on motion editing has focused on accurately measuring the underlying motion and carefully constructing coherent motion layers in the scene. These kind of motion-based techniques are often not suitable for analyzing time-lapse videos. The jerky nature of these sequences violates the core assumptions of motion analysis and optical flow, and prevents even the most sophisticated motion estimation algorithm from obtaining accurate enough motion for further analysis.

We propose a novel computational approach to motion denoising in videos that does not require explicit motion estimation and modeling. We formulate the problem in a Bayesian framework, where the goal is to recover a "smooth version" of the input video by reshuffling its pixels spatiotemporally. This translates to a well-defined inference problem over a 3D Markov Random Field (MRF), which we solve using a time-space optimized Loopy Belief Propagation (LBP) algorithm. We show how motion denoising can be used to eliminate short-term motion jitters in time-lapse videos, and present results for time-lapse sequences of different nature and scenes.

■ 2.2 Background and Related Work

The input to our system is an $M \times N \times T$ video sequence, $I(x, y, t)$, with RGB intensities given in range $[0, 255]$. Our goal is to produce an $M \times N \times T$ output sequence, $J(x, y, t)$, in which short-term jittery motions are removed and long-term scene changes are maintained. A number of attempts have been made to tackle similar problems from a variety of perspectives, which we will now briefly review.

Temporal filtering. A straightforward approach is to pass the sequence through a temporal low-pass filter

$$J(x, y, t) = f(I(x, y, \{k\}_{t-\delta_t}^{t+\delta_t}) \tag{2.1}$$

where f denotes the filtering opreator, and δ_t defines the temporal window size. For a video sequence with a static viewpoint, f is often taken as the median operator, which is useful for tasks such as background-foreground segmentation and noise reduction. This approach, albeit

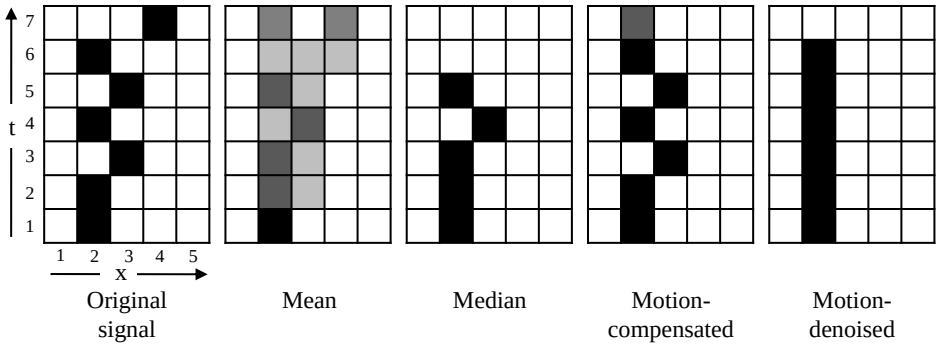


Figure 2.2: The responses of different temporal filters on a canonical, 1D signal. The mean and median temporal filters operate by sliding a window temporally at each spatial location, and setting the intensity at the center pixel in the window to the mean and median intensity value of the pixels inside the window, respectively. The motion-compensated filter computes the mean intensity value along the estimated motion trajectory. In this example, the temporal filters are of size 3, centered at the pixel, and the motion denoising algorithm uses a 3×3 support. For illustration, we assume the temporal trajectory estimated by the motion-compensated filter is accurate until $t = 6$. That is, the motion from $(x, t) = (2, 6)$ to $(4, 7)$ was not detected correctly.

simple and fast, has an obvious limitation – the filtering is performed independently at each pixel. In a dynamic scene with rapid motion, pixels belonging to different objects are averaged, resulting in a blurred or discontinuous result. Figure 2.2 demonstrates this on a canonical signal, and figure 2.3 further illustrates these effects on a natural time-lapse video.

To address this issue, motion-compensated filtering was introduced, filtering the sequence along motion trajectories (*e.g.*, [34]). Such techniques are commonly employed for video compression, predictive coding, and noise removal. Although this approach is able to deal with some of the blur and discontinuity artifacts, as pixels are only integrated along the estimated motion path, it does not filter the actual motion (i.e. making the motion trajectory smoother), but rather takes the motion into account for filtering the sequence. In addition, errors in the motion estimation may result in unwanted artifacts at object boundaries.

Motion editing. One work that took a direct approach to motion editing in videos is "Motion Magnification" [23]. A layer segmentation system was proposed for exaggerating motions that might otherwise be difficult or even impossible to notice. Similar to our work, they also manipulate video data to resynthesize a sequence with modified motions. However, our work modifies the motion without explicit motion analysis or layer modeling that are required by

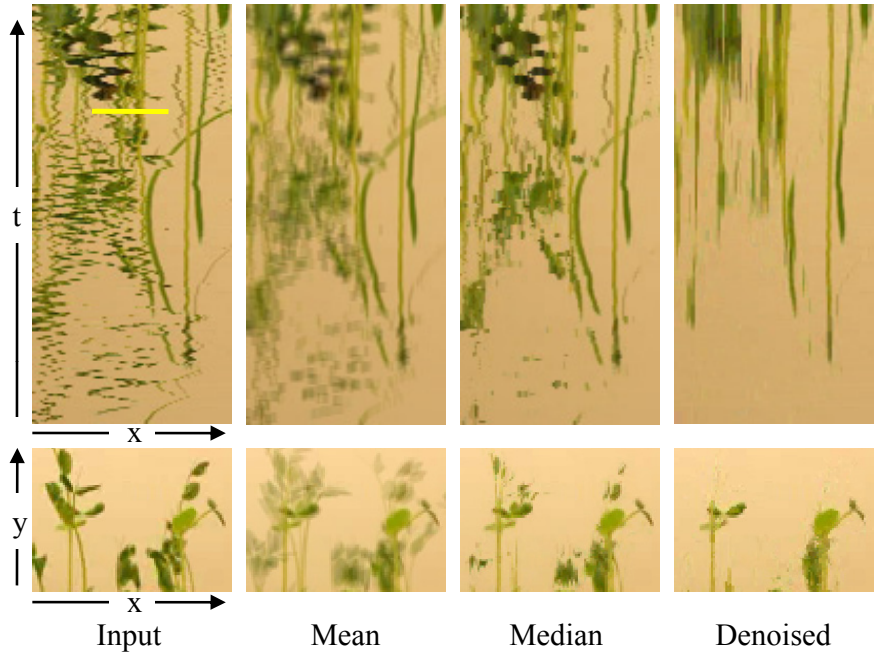


Figure 2.3: Comparison of motion denoising with simple temporal filtering on the plants sequence of Figure 2.1. At the top, zoom-ins on the left part of the XT slice from Figure 2.1 are shown for the original sequence, the (temporally) mean-filtered sequence, the median-filtered sequence, and the motion-denoised sequence. At the bottom, a representative spatial patch from each sequence is shown, taken from a region roughly marked by the yellow bar on the input video slice.

their method. In fact, layer estimation can be challenging for the sequences we are interested in; these sequences may contain too many layers (*e.g.*, leaves) for which even state-of-the-art layer segmentation algorithms would have difficulties producing reliable estimates.

Motion denoising has been addressed before in a *global* manner, known as video stabilization (*e.g.*, [31, 28]). Camera jitter (mostly from hand-held devices) is modeled using image-level transforms between consecutive frames, which are used to synthesize a sequence with smoother camera motion. Image and video inpainting techniques are often utilized to fill-in missing content in the stabilized sequence. In contrast, we focus on stabilization at the *object level*, supporting spatially-varying, pixel-wise displacement within a single frame. Inpainting is built-in naturally into our formulation.

Time-lapse videos. Time-lapse videos are valuable for portraying events occurring throughout long time periods. Typically, a frame is captured once every few minutes or hours over a

long time period (*e.g.*, weeks, months), and the captured frames are then stacked in time to create a video depicting some long-term phenomena [10]. Captured sequences span a large variety of scenes, from building construction, through changes in the human body (*e.g.*, pregnancy, aging), to natural phenomena such as celestial motion and season change. Capturing time-lapse sequences no longer requires an expert photographer. In fact, it is supported as built-in functionality in many modern consumer digital cameras.

Although sequences captured by time-lapse photography are very different in nature, they all typically share a common artifact—*stylized jerkiness*—caused by temporal aliasing that is inherent to the time-lapse capture process. Rapidly moving objects, as well as lighting changes, and even small camera movements (common if the camera is positioned outdoors) then appear as distracting, non-physical jumps in the video. These effects might sometimes be desirable, however in often cases they simply clutter the main temporal events the photographer wishes to portray.

Previous academic work involving time-lapse sequences use them as an efficient representation for video summarization. Work such as [4, 39] take a video-rate footage as input, and output a sequence of frames that succinctly depict temporal events in the video. Work such as [39, 40] can take an input time-lapse video, and produce a single-frame representation of that sequence, utilizing information from throughout the time span. Time-lapse sequences have also been used for other applications. For example, Weiss [57] uses a sequence of images of a scene under varying illumination to estimate intrinsic images. These techniques and applications are significantly different from ours. Both input and output of our system are time-lapse videos, and our goal is to improve the input sequence quality by suppressing short-term distracting events and maintaining the underlying long-term flux of the scene.

Direct editing of time-lapse imagery was proposed in [49] by factorizing each pixel in an input time-lapse video into shadow, illumination and reflectance components, which can be used for relighting or editing the sequence, and for recovering the scene geometry. In our work, we are interested in a different type of decomposition: separating a time-lapse sequence into shorter- and longer-term events. These two types of decompositions can be complimentary to each other.

Geometric rearrangement. In the core of our method is a statistical (data-driven) algorithm for inferring smooth motion from noisy motion by rearranging the input video. Content rearrangement in images and video has been used in the past for various editing applications. Image

reshuffling is discussed in [9]. They work in patch space, and the patch moves are constrained to an underlying coarse grid which does support relatively small motions common to time-lapse videos. We, on the other hand, work in pixel resolution, supporting both small and irregular displacements. [38] perform geometric image rearrangement for various image editing tasks using a MRF formulation. Our work concentrates on a different problem and requires different formulation. Inference in videos is much more challenging than in images, and we use different inference tools from the ones used in their work.

For videos, [46] consider the sequence appearance and dynamics for shifting entire frames to create a modified playback. [39] generate short summaries for browsing and indexing surveillance data by shifting individual pixels temporally while keeping their spatial locations intact. [52] align two videos using affine spatial and temporal warps. Our problem is again very different from all these work. Temporal shifts alone are insufficient to denoise noisy motion, and spatial offsets must be defined in higher granularity than the global frame or video. This makes the problem much more challenging to solve.

■ 2.3 Formulation

We model the world as evolving slowly through time. That is, within any relatively small time span, we assume objects in the scene attain some stable configuration, and changes to these configurations occur in low time rates. Moreover, we wish to make use of the large redundancy in images and videos to infer those stable configurations. This leads to the following formulation.

Given an input video I, we seek an output video J that minimizes the energy $E(J)$, defined as

$$E(J) = \sum_{x,y,t} \left|J(x,y,t) - I(x,y,t)\right| + \alpha \sum_{x,y,t} \left|J(x,y,t) - J(x,y,t+1)\right|, \tag{2.2}$$

subject to

$$J(x,y,t) = I(x + w_x(x,y,t), y + w_y(x,y,t), t + w_t(x,y,t)) \tag{2.3}$$

for spatiotemporal displacement field

$$w(x,y,t) \in \{(\delta_x, \delta_y, \delta_t) : |\delta_x| \leq \Delta_s, |\delta_y| \leq \Delta_s, |\delta_t| \leq \Delta_t\}, \tag{2.4}$$

where (Δ_s, Δ_t) are parameters defining the support (search) region.

In this objective function, the first term is a *fidelity term*, enforcing the output sequence to resemble the input sequence at each location and time. The second term is a *temporal coherence* term, which requires the solution to be temporally smooth. The tension between those two terms creates a solution which maintains the general appearance of the input sequence, and is temporally smooth. This tradeoff between appearance and temporal coherence is controlled via the parameter α.

As J is uniquely defined by the spatiotemporal displacements, we can equivalently rewrite Equation 2.2 as an optimization on the displacement field w. Further parameterizing $p = (x, y, t)$, and plugging constraint 2.3 into Equation 2.2, we get

$$\begin{aligned} E(w) = &\sum_{p} \big|I(p + w(p)) - I(p)\big| + \\ &\alpha \sum_{p,r \in \mathcal{N}_t(p)} \big|\big|I(p + w(p)) - I(r + w(r))\big|\big|^2 + \\ &\gamma \sum_{p,q \in \mathcal{N}(p)} \lambda_{pq} \big|w(p) - w(q)\big|, \end{aligned} \tag{2.5}$$

where we added an additional term for regularizing the displacement field w, with weight $\lambda_{pq} = \exp\{-\beta||I(p) - I(q)||^2\}$. β is learnt as described in [50]. λ_{pq} assigns varying weight to discontinuities in the displacement map, as function of the similarity between neighboring pixels in the original video. $\mathcal{N}(p)$ denotes the spatiotemporal neighborhood of pixel p, and $\mathcal{N}_s(p), \mathcal{N}_t(p) \subseteq \mathcal{N}(p)$ denote the spatial and temporal neighbors of p, respectively. We use the six spatiotemporal pixels directly connected to p as the neighborhood system.

α and γ weight the temporal coherence and regularization terms, respectively. The L_2 norm is used for temporal coherence to discourage motion discontinuities, while L_1 is used in the fidelity and regularization terms to remove noise from the input sequence, and to account for discontinuities in the displacement field, respectively.

■ 2.3.1 Optimization

We optimize Equation 2.5 discretely on a 3D MRF corresponding to the three-dimensional video volume, where each node p corresponds to a pixel in the video sequence and represents the latent variables $w(p)$. The state space in our model is the set of possible three-dimensional displacements within a predefined search region (Equation 2.4). The potential functions are

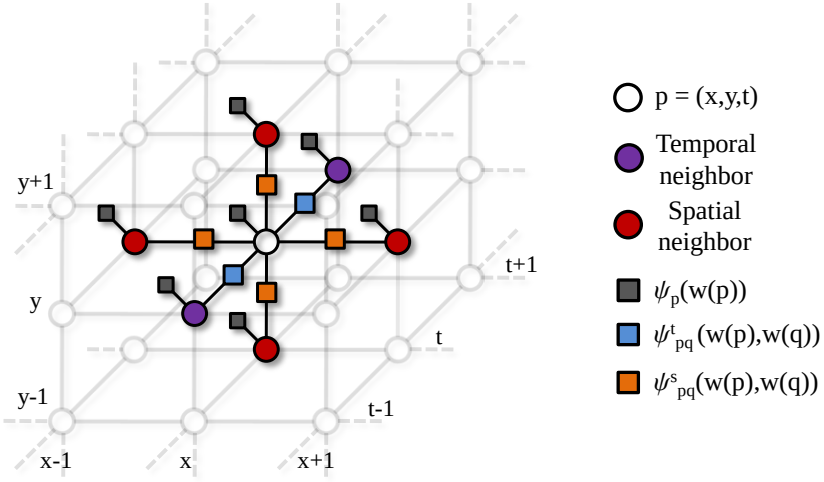


Figure 2.4: An illustration of the graphical model corresponding to Equation 2.5. Note that each node contains the three (unknown) components of the spatiotemporal displacement at that location.

given by

$$\psi_p(w(p)) = \big|I(p + w(p)) - I(p)\big|, \tag{2.6}$$

$$\psi^t_{pr}(w(p), w(r)) = \alpha\big|\big|I(p + w(p)) - I(r + w(r))\big|\big|^2 + \gamma\lambda_{pr}\big|w(p) - w(r)\big|, \tag{2.7}$$

$$\psi^s_{pq}(w(p), w(q)) = \gamma\lambda_{pq}\big|w(p) - w(q)\big|, \tag{2.8}$$

where ψ_p is the unary potential at each node, and ψ^t_{pr}, ψ^s_{pq} denote the temporal and spatial pairwise potentials, respectively. Figure 2.4 depicts the structure of this graphical model.

We have experimented with several optimization techniques for solving Equation 2.5, namely Iterated Conditional Modes (ICM), α-expansion (GCUT) [6] and Loopy Belief Propagation (LBP) [60]. Our temporal pairwise potentials (Equation 2.7) are neither a metric nor a semi-metric, which makes the graph-cut based algorithms theoretically inapplicable to this optimization. Although previous work use those algorithms ignoring the metric constraints and still report good results (*e.g.*, [38]), our experiments consistently showed that LBP manages to produce more visually appealing sequences, and in most cases also achieves lower energy solutions compared to the other solvers. We therefore choose LBP as our inference engine.

Figure 2.5 compares the results of the three optimizations. The complete sequences are available in the supplementary material. The ICM results suffer, as expected, from noticeable

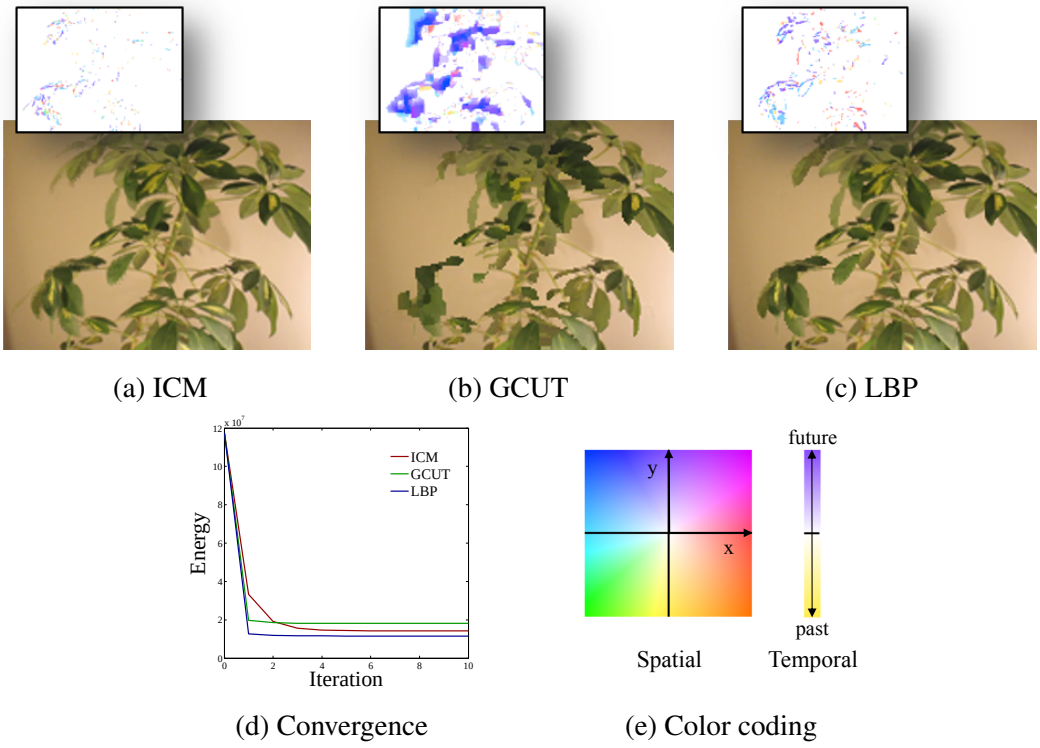


Figure 2.5: Comparison between different optimization techniques for solving Equation 2.5, demonstrated on the *plant* time-lapse sequence (Figure 2.7). (a-c) Representative frames from each result. The spatial components of the displacement fields (overlaid) illustrate that different local minima are attained by the different optimization methods (the full sequences are available in the accompanying material). (d) The energy convergence pattern over 10 iterations of the algorithms. (e) The color coding used for visualizing the displacement fields, borrowed from [3].

discontinuities. There are also noticeable artifacts in the graph-cut solution. As part of our pairwise potentials are highly non-metric, it is probable that α-expansion will make incorrect moves that will adversely affect the results. All three methods tend to converge quickly within $3-5$ iterations, which agrees with the related literature [50]. Although the solution energies tend to be within the same ballpark, we noticed they usually correspond to different local minima.

■ 2.3.2 Implementation Details

Our underlying graphical model is a massive 3D grid, which imposes computational difficulties in both time and space. For tractable runtime, we extend the message update schedule by

Tappen and Freeman [51] to 3D, by first sending messages (forth and back) along rows in all frames, then along columns, and finally along time. This sequential schedule allows information to propagate quickly through the grid, and helps the algorithm converge faster. To search larger ranges, we apply LBP to a spatiotemporal video pyramid. Since time-lapse sequences are temporally aliased, we apply smoothing to the spatial domain only, and sample in the temporal domain. At the coarser level, the same search volume effectively covers twice the volume used in the finer level, allowing the algorithm to consider larger spatial and temporal ranges. To propagate the displacements to the finer level, we bilinear-interpolate and scale (multiply by 2) the shifts, and use them as centers of the search volume at each pixel in the finer level.

The complexity of LBP is linear in the graph size, but quadratic in the state space. In our model, we have a K^3 search volume (for $\Delta_s = \Delta_t = K$), which requires K^6 computations per message update and may quickly become intractable even for relatively small search volumes. Nevertheless, we can get significant speedup in the computation of the spatial messages using distance transform, as the 3D displacement components are decoupled in the L_1-norm distance. Felzenszwalb and Huttenlocher have shown that computing a distance transform on such 3D label grids can be reduced to consecutive efficient computations of 1D distance transforms [13]. The overall complexity of this computation is $O(3K^3)$, which is linear in the search range. For our multiscale computation, Liu *et al.* [27] already showed how the distance transform can be extended to handle offsets (corresponding to the centers of the search volumes obtained from the coarser level of the video pyramid) in 2D. Following the reduction in [13] therefore shows that we can trivially handle offsets in the 3D case as well. We note that the distance transform computation is not an approximation, and results in the exact message updates.

We briefly demonstrate the computation of the spatial messages in our formulation using distance transform, and refer the interested reader to [13] for more details. Our min-sum spatial message update equation can be written as

$$m_{p \to q}(w_q) = \min_{w_p} \Big(|w_p - w_q| + h_p(w_p) \Big), \tag{2.9}$$

where h_p is the distance transform function, given by $h_p(w_p) = \psi_p(w_p) + \sum_{r \in \mathcal{N}_s(p) \setminus q} m_{r \to p}(w_p)$.

We can expand Equation 2.9 as

$$\begin{aligned} & m_{p\to q}(w_q^x, w_q^y, w_q^t) \\ & = \min_{w_p^x, w_p^y, w_p^t} \Big(|w_p^x - w_q^x| + |w_p^y - w_q^y| + |w_p^t - w_q^t| + h_p(w_p) \Big) \\ & = \min_{w_p^t} |w_p^t - w_q^t| + \Big(\min_{w_p^y} |w_p^y - w_q^y| + \Big(\min_{w_p^x} |w_p^x - w_q^x| + h_p(w_p) \Big) \Big) \Big). \end{aligned} \tag{2.10}$$

Written in this form, it is evident that the the 3D distance transform can be computed by consecutively computing the distance transform of each component in place. Note that the order of the components above was chosen arbitrarily, and does not affect the decomposition.

This yields a significant improvement in running time, since $2/3$ of the messages propagating through the graph can be computed in linear time. Unfortunately, we cannot follow a similar approach for the temporal messages, as the temporal pairwise potential function (Equation 2.7) is non-convex.

This massive inference problem imposes computational difficulties in terms of space as well. For example, a 500^3 video sequence with a 10^3 search region requires memory, for the messages only, of size at least $500^3 \times 10^3 \times 6 \times 4 \simeq 3$ terabytes (!). Far beyond current available RAM, and probably beyond the average available disk space. We therefore restrict our attention to smaller sequences and search volumes. As the messages structure cannot fit entirely in memory, we store it on disk, and read and write the necessary message chunks on need. For our message update schedule, it suffices to maintain in memory the complete message structure for one frame for passing messages spatially, and two frames for passing messages temporally. This imposes no memory difficulty even for larger sequences, but comes at the cost of lower performance as disk I/O is far more expensive than memory access. Section 2.4 details the algorithm's space and time requirements for the videos and parameters we used.

Finally, once LBP converges or message passing is complete, the MAP label assignment is traditionally computed independently at each node [14]:

$$\widehat{w}_p = \arg\min_{w_p} \Big(\psi_p(w_p) + \sum_{q \in \mathcal{N}(p)} m_{q\to p}(w_p) \Big). \tag{2.11}$$

For our problem, we observe that the local conditional densities are often multi-modal (Figure 2.6), indicating multiple possible solutions that are equivalent, or close to equivalent, with respect to the objective function. The label (displacement) assigned to each pixel using Equation 2.11 therefore depends on the order of traversing the labels, which is somewhat arbitrary.

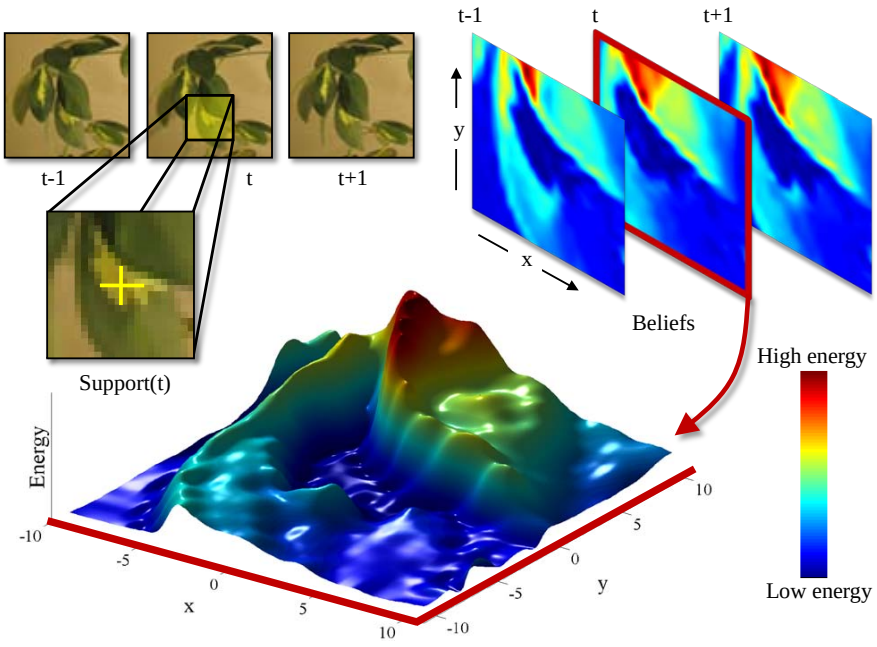


Figure 2.6: A visualization of the beliefs computed by LBP for a single pixel in the *plant* sequence, using a $31 \times 31 \times 3$ support. The support frame containing the pixel is zoomed-in on the upper left. The beliefs over the support are shown on the upper right, colored from blue (low energy) to red (high energy). We seek the displacement that has the minimal energy within the support region. At the bottom, the belief surface is shown for the middle frame of the support region, clearly showing multiple equivalent (or nearly equivalent) solutions.

As a result, the selected displacements at neighboring pixels need not be coherent with respect to the pairwise potentials.

We propose a different procedure for assigning the MAP label at each node. Following our message update schedule, we start with the pixel at position $(0, 0, 0)$ and assign it the label satisfying Equation 2.11. Then, traversing the grid from back to front, top to bottom and left to right, we assign each node the label according to

$$\widehat{w}_p^* = \arg\min_{w_p} \Big(\psi_p(w_p) + \sum_{q \in \mathcal{P}(p)} \psi_{pq}(w_p, \widehat{w}_q^*) + \sum_{q \in \mathcal{N}(p) \setminus \mathcal{P}(p)} m_{q \to p}(w_p) \Big), \tag{2.12}$$

where $\mathcal{P}(p)$ denotes the left, top, and backward neighbors of node p. Notice that these nodes were already assigned labels by the time node p is reached in this scan pattern, and so their assignments are used while determining the assignment for p. Overall, this process produces a MAP assignment that is locally more coherent with respect to the objective function. In

practice, we found that the solutions produced with this approach have energies $2-3\%$ lower on average comparing to independent assignment of states to pixels, and the decrease in energy is obviously larger when heavier regularization is sought.

■ 2.4 Results

Our main application of interest is time-lapse processing, and so we ran the motion denoising algorithm on several time-lapse sequences of different nature. We fixed the parameters in all the experiments to $\alpha = 2$, $\gamma = 10$, $\Delta_s = 7$, $\Delta_t = 5$, and used a 2-level video pyramid. We ran LBP for $5-10$ iterations, during which the algorithm converged to a stable minima. Representative frames for each experiment are shown in Figures 2.7, 2.8 and 2.10, and the full sequences and results are available in the accompanying material.

Recall that our basic assumption is that the input sequence contains events of different time scales. We first produced sequences which demonstrate this effect in a controlled environment. First, we set up a Canon PowerShot series camera shooting a plant indoors ((*plant*, Figure 2.7 top). We used a fan for simulating wind, and a slow-moving light source for emulating a low-variation change in the scene. We then sampled the captured video at a low frame rate to introduce aliasing in time. A sampling rate of 2 frames/sec allowed sufficient time for the leaves to move and create the typical motion jitter effect. As can be seen in the result, our algorithm manages to find a stable and faithful configuration for the plant, while perfectly maintaining the (longer-term) lighting change in the scene.

sprouts (Figure 2.7 bottom) illustrates the process of plants growing in a similar indoor setup. In this experiment, we set the camera to capture a still image every 15 minutes, and the sprouts were placed in an enclosure so that motions are created solely by the plants. The motion-denoised result appears smoother than the original sequence, and captures the growth process of the plants with the short-term noisy motions removed. It can be seen that for some parts of the plants the motions are stabilized (as also shown in the spatiotemporal slices of the video (Figures 2.1, 2.7), while other parts, namely the sprouts' tops, are partly removed in the result. The motion at the top of the stems is the largest, and the algorithm is sometimes unable to infer the underlying stationary configuration using the search volumes we used. In such cases, the algorithm gracefully removes these objects and fills-in their place with the background or other objects they occlude, so as to generate a smooth looking sequence. Enlarging the support region allows the algorithm to denoise larger motions (Figure 2.11), but has a large impact on

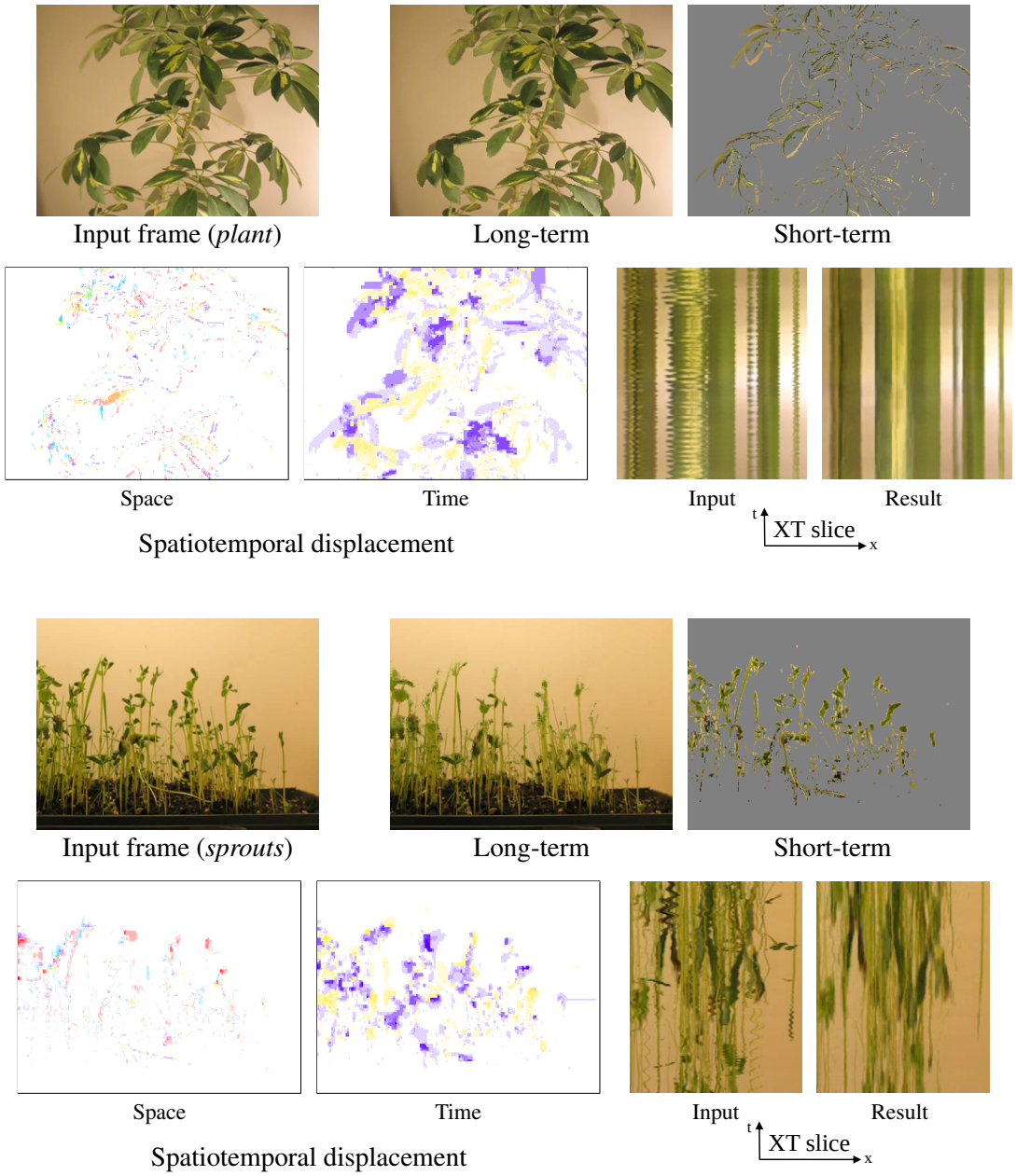


Figure 2.7: Motion denoising results on the time-lapse sequences *plant* and *sprouts*. For each sequence, we show a representative source frame (top left), representative frames of the long-term (motion-denoised) and short-term changes (top right), the computed displacement field (bottom left; spatial displacement on the left, temporal displacement on the right), and a part of an XT slice of the video volumes for the input and motion-denoised result (bottom right). The short-term result is computed by thresholding the color difference between the input and motion-denoised frames and copying pixels from the input. The vertical position of the spatiotemporal slice is chosen such that salient sequence dynamics are portrayed.

its performance as will be discussed shortly.

Time-lapse videos span a large variety of scenes and styles. To test our technique on a different type of time-lapse sequences, we experimented with a sequence of a melting glacier (Figure 2.10) taken by the Extreme Ice Survey (EIS) [1]. EIS documents the changes to the Earth's glacial ice via time-lapse photography. We received their raw time-lapse capture of the Sólheimajökull glacier in Iceland, taken between April 2007 and May 2010 at a rate of 1 frame per hour. While the video appears cluttered due to changes in lighting and weather conditions in this extreme environment, the characteristics of this sequence and motions within it are quite different than the ones we addressed before. Specifically, the decomposition into short-term and long-term changes is not as evident.

To reduce the sequence size and to prevent the large scene variations from contaminating the result, we first sampled this sequence using a non-uniform sampling technique similar to the one proposed by [4], and then ran our motion denoising algorithm on the selected frames (Figure 2.9). We found the gist descriptor [33] useful as a frame distance measure for selecting the frames, producing a more temporally-coherent sampling than the one produced with L_1 or L_2 distance in color space as used in [4].

Our algorithm's result on this sequence resembles the response of a temporal filter. Without apparent motions jitters, the best solution is to smooth the sequence temporally. Indeed, the computed displacements are dominated by the temporal component (Figure 2.10), meaning that the algorithm reduces to temporal rearrangement. This, however, happens naturally without user intervention or parameter tuning.

More results are shown in Figure 2.8. *pool* and *pond* nicely illustrate how the algorithm manages to maintain temporal events in the scene while eliminating short term distracting motions. Notice how in the *pool* sequence, the shadows are maintained on the porch and near the swing chair, while the rapid motions of the tree in the back and flowers in the front are stabilized. Worthy of note is that the input sequences were downloaded from the web in moderate resolution, showing robustness of the algorithm to video quality and compression artifacts.

In *pond*, the algorithm manages to denoise the jittery motions in the water and vegetation, while maintaining the motion of the sun and its reflection on the water. Some artifacts are apparent, however, in the sky region. This is because the clouds' dynamics does not fully fit our motion decomposition model – they neither exhibit jittery motion per se, nor evolve slowly as other objects (*e.g.*, the sun) in the scene.

Despite our optimizations, the running time of the algorithm is still substantial. Solving for

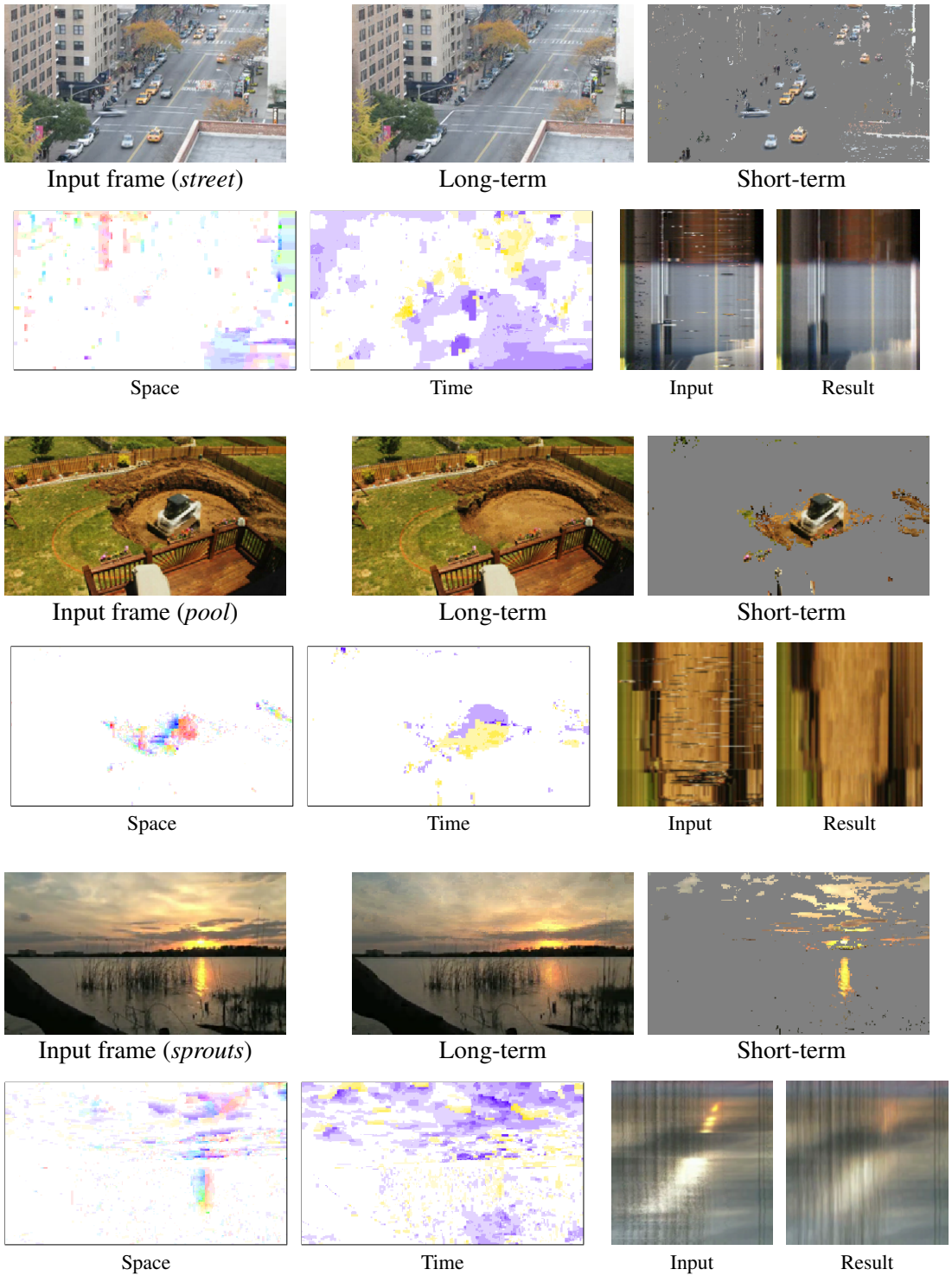


Figure 2.8: Additional results on the time-lapse sequences *street*, *pool*, and *pond*, shown in the same layout as in Figure 2.7.

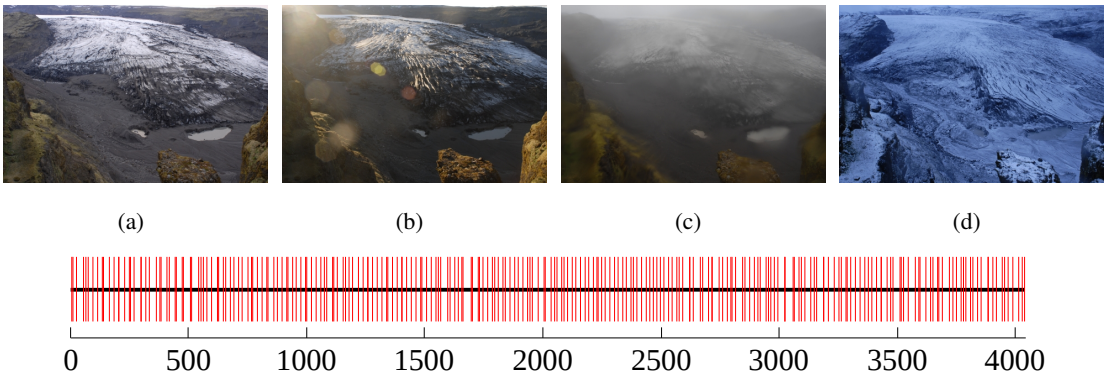


Figure 2.9: Four frames from the glacier time-lapse (top), taken within the same week of May 18, 2007, demonstrate the large variability in lighting and weather conditions, typical to an outdoor time-lapse footage. For this sequence, we first apply a non-uniform sampling procedure (bottom; see text) to prevent noisy frames from affecting the synthesis. The x-axis is the frame number, and the vertical lines represent the chosen frames.

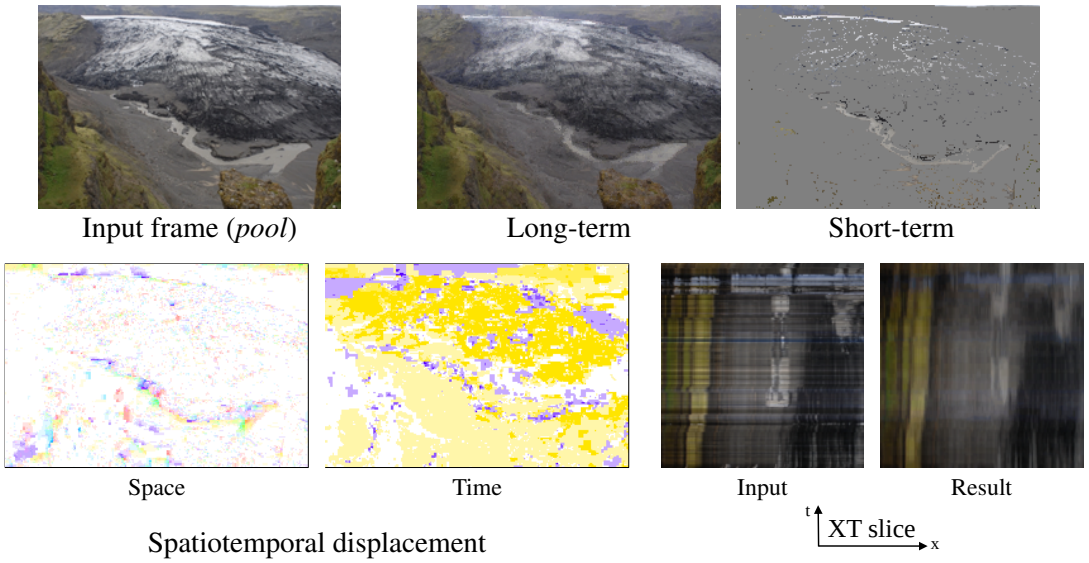


Figure 2.10: Result on the time-lapse sequence *glacier*, shown in the same layout as in Figure 2.7.

a sequence of size 300^3 with a 7^3 search volume takes approximately 50 hours for our CPU-based C++ implementation of LBP using sequential message passing, on a quad-core Intel Xeon 2.66GHZ desktop with 28GB RAM. For such dimensions, the algorithm makes use of 1GB of RAM, and up to 50GB scratch space on disk. We used an Intel X-25M solid state drive,

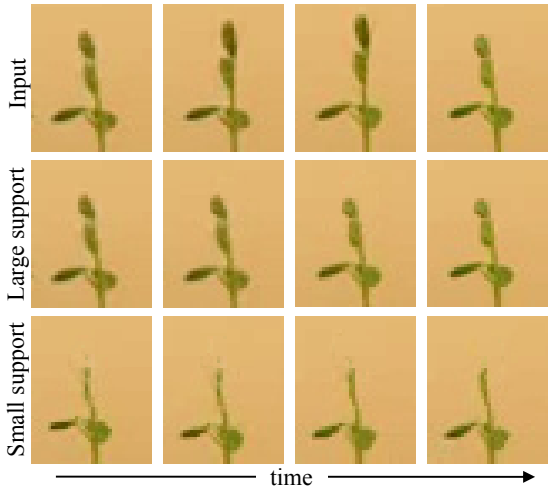


Figure 2.11: Zoom-in on the rightmost plant in the *sprouts* sequence in four consecutive frames shows that enlarging the search volume used by the algorithm can greatly improve the results. "Large support" corresponds to a $31 \times 31 \times 5$ search volume, while "small support" is the $7 \times 7 \times 5$ volume we used in our experiments.

which gave an approximate x2 speedup in disk I/O, but the running time was dominated by the computation of the temporal messages. As this computation is quadratic in the size of the search volume, using larger volumes means even larger increase in the run time. We therefore resorted to using relatively small search volumes in our experiments.

Even with multi-scale processing, some objects might exhibit motions larger than the range scanned by the algorithm. In such cases they will be gracefully removed from the sequence, which might not necessarily be the desired result. Figure 2.11 demonstrates that this limitation is computational, rather than theoretical.

Chapter 3

Motion and Color Magnification

It is only with the heart that one can see rightly; what is essential is invisible to the eye.

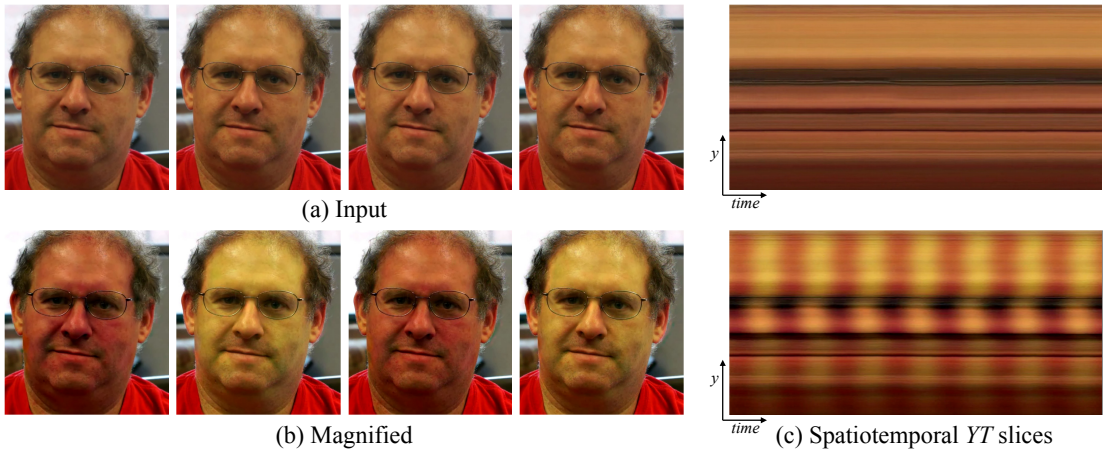
–Antoine de Saint-Exupery

In the previous chapter, we saw examples of imagery containing random, distracting variation that is important to remove. In many other cases, however, it is essential to *amplify* temporal variations that are of interest, yet may be imperceptible to us. In this chapter, we describe a computational "microscope" for such variations – new techniques we developed that can efficiently extract, from regular video, subtle motion and color changes that are difficult or impossible to see with the naked eye, and re-render a video with the desired changes amplified.

Our techniques model changes through space and time in a way that is analogous to an *Eulerian* framework for fluid flow analysis, observing and manipulating temporal variations at fixed locations in space. We call those techniques *Eulerian video processing*. We describe two Eulerian approaches to analyze and manipulate small-amplitude motion and color changes in video, which we call *linear* and *phase-based*. We discuss their tradeoffs, and also compare between our proposed Eulerian techniques and previous *Lagrangian* methods to amplify motion. We use our techniques to reveal a variety of faint yet informative visual signals related to physiological and mechanical functions.

■ 3.1 Introduction

The human visual system has limited spatio-temporal sensitivity, but many signals that fall below this capacity can be informative. For example, human skin color varies slightly with blood circulation. This variation, while invisible to the naked eye, can be exploited to extract pulse rate [53, 36, 35]. Similarly, motion with low spatial amplitude, while hard or impossible



(a) Input (b) Magnified (c) Spatiotemporal *YT* slices

Figure 3.1: An example of using our Eulerian Video Magnification framework for visualizing the human pulse. (a) Four frames from the original video sequence (*face*). (b) The same four frames with the subject's pulse signal amplified. (c) A vertical scan line from the input (top) and output (bottom) videos plotted over time shows how our method amplifies the periodic color variation. In the input sequence the signal is imperceptible, but in the magnified sequence the variation is clear. The complete sequence is available in the supplemental video.

for humans to see, can be magnified to reveal interesting mechanical behavior [26]. The success of these tools motivates the development of new techniques to reveal invisible signals in videos.

Our basic approach is to consider the time series of color values at any spatial location (pixel) and amplify variation in a given temporal frequency band of interest (Section 3.2). For example, in Figure 3.1 we automatically select, and then amplify, a band of temporal frequencies that includes plausible human heart rates. The amplification reveals the variation of redness as blood flows through the face. For this application, temporal filtering needs to be applied to lower spatial frequencies (spatial pooling) to allow such a subtle input signal to rise above the camera sensor and quantization noise.

It turns out that this temporal filtering approach not only amplifies color variation, but can also reveal low-amplitude motion. For example, in the supplemental video, we show that we can enhance the subtle motions around the chest of a breathing baby. We provide a mathematical analysis that explains how temporal filtering interplays with spatial motion in videos. Our analysis relies on a linear approximation related to the brightness constancy assumption used in optical flow formulations. We also derive the conditions under which this approximation holds. This leads to a multiscale approach to magnify motion without feature tracking or motion

estimation.

Previous attempts have been made to unveil imperceptible motions in videos. [26] analyze and amplify subtle motions and visualize deformations that would otherwise be invisible. [56] propose using the Cartoon Animation Filter to create perceptually appealing motion exaggeration. These approaches follow a *Lagrangian* perspective, in reference to fluid dynamics where the trajectory of particles is tracked over time. As such, they rely on accurate motion estimation, which is computationally expensive and difficult to make artifact-free, especially at regions of occlusion boundaries and complicated motions. Moreover, Liu et al. [26] have shown that additional techniques, including motion segmentation and image in-painting, are required to produce good quality synthesis. This increases the complexity of the algorithm further.

In contrast, we are inspired by the *Eulerian* perspective, where properties of a voxel of fluid, such as pressure and velocity, evolve over time. In our case, we study and amplify the variation of pixel values over time, in a spatially-multiscale manner. In our Eulerian approach to motion magnification, we do not explicitly estimate motion, but rather exaggerate motion by amplifying temporal color changes at fixed positions. We rely on the same differential approximations that form the basis of optical flow algorithms [30, 22].

Temporal processing has been used previously to extract invisible signals [36] and to smooth motions [18]. For example, Poh et al. [36] extract a heart rate from a video of a face based on the temporal variation of the skin color, which is normally invisible to the human eye. They focus on extracting a single number, whereas we use localized spatial pooling and bandpass filtering to extract and reveal visually the signal corresponding to the pulse[1]. This primal domain analysis allows us to amplify and visualize the pulse signal at each location on the face. This has important potential monitoring and diagnostic applications to medicine, where, for example, the asymmetry in facial blood flow can be a symptom of arterial problems.

Fuchs et al. [18] use per-pixel temporal filters to dampen temporal aliasing of motion in videos. They also discuss the high-pass filtering of motion, but mostly for non-photorealistic effects and for large motions (Figure 11 in their paper). In contrast, our method strives to make imperceptible motions visible using a multiscale approach. We analyze our method theoretically and show that it applies only for small motions.

Finally, we further extend our approach and introduce a technique to manipulate small movements in videos based on an analysis of motion in complex-valued image pyramids (Sec-

[1] See Section 3.2.3 for the algorithm we used to extract the heart rate, which we did not describe in detail in our SIGGRAPH 2012 paper [59].

tion 3.3). Phase variations of the coefficients of a complex-valued steerable pyramid over time correspond to motion, and can be temporally processed and amplified to reveal imperceptible motions, or attenuated to remove distracting changes. In comparison to the aforementioned method, this phase-based method has a higher computational overhead, but supports larger amplification factors and is significantly less sensitive to noise. These improved capabilities significantly broaden the set of applications for Eulerian motion processing in video.

■ 3.2 Eulerian Video Magnification

Our goal is to reveal temporal variations in videos that are difficult or impossible to see with the naked eye and display them in an indicative manner. Our method, which we call Eulerian Video Magnification, takes a standard video sequence as input, and applies spatial decomposition, followed by temporal filtering to the frames. The resulting signal is then amplified to reveal hidden information. Using our method, we are able to visualize the flow of blood as it fills the face and also to amplify and reveal small motions. Our technique can run in real time to show phenomena occurring at temporal frequencies selected by the user.

We make several contributions. First, we demonstrate that nearly invisible changes in a dynamic environment can be revealed through *Eulerian* spatio-temporal processing of standard monocular video sequences. Moreover, for a range of amplification values that is suitable for various applications, explicit motion estimation is not required to amplify motion in natural videos. Our approach is robust and runs in real time. Second, we provide an analysis of the link between temporal filtering and spatial motion and show that our method is best suited to small displacements and lower spatial frequencies. Third, we present a single framework that can be used to amplify both spatial motion and purely temporal changes, e.g., the heart pulse, and can be adjusted to amplify particular temporal frequencies—a feature which is not supported by Lagrangian methods. Finally, we analytically and empirically compare Eulerian and Lagrangian motion magnification approaches under different noisy conditions. To demonstrate our approach, we present several examples where our method makes subtle variations in a scene visible.

■ 3.2.1 Space-time Video Processing

Consider a local fixed region in a video. Color changes in the local region could be attributed to one of two possible reasons: (a) a static object inside the region has changed color, and (b) an

object in the region has moved, in which case the region may now contain a different part of the object that was there before, or a different object altogether. Amplifying these color changes locally and rendering them back into the video will make them more perceptible to the human eye.

Our approach combines spatial and temporal processing to emphasize subtle temporal changes in a video. The process is illustrated in Figure 3.2. We first decompose the video sequence into different spatial frequency bands. These bands might be magnified differently because (a) they might exhibit different signal-to-noise ratios, or (b) they might contain spatial frequencies for which the linear approximation used in our motion magnification does not hold (Sect. 3.2.2). In the latter case, we reduce the amplification for these bands to suppress artifacts. When the goal of spatial processing is simply to increase temporal signal-to-noise ratio by pooling multiple pixels, we spatially low-pass filter the frames of the video and downsample them for computational efficiency. In the general case, however, we compute a full Laplacian pyramid [8].

We then perform temporal processing on each spatial band. We consider the time series corresponding to the value of a pixel in a frequency band and apply a bandpass filter to extract the frequency bands of interest. For example, we might select frequencies within 0.4-4Hz, corresponding to 24-240 beats per minute, if we wish to magnify a pulse. If we are able to extract the pulse rate, we can use a narrow band around that value. The temporal processing is uniform for all spatial levels, and for all pixels within each level. We then multiply the extracted bandpassed signal by a magnification factor α. This factor can be specified by the user, and may be attenuated automatically according to guidelines in Sect. 3.2.2. Possible temporal filters are discussed in Sect. 3.2.4. Next, we add the magnified signal to the original and collapse the spatial pyramid to obtain the final output. Since natural videos are spatially and temporally smooth, and since our filtering is performed uniformly over the pixels, our method implicitly maintains spatiotemporal coherency of the results.

■ 3.2.2 Eulerian Motion Magnification

Our processing can amplify small motion even though we do not track motion as in Lagrangian methods [26, 56]. In this section, we show how temporal processing produces motion magnification using an analysis that relies on the first-order Taylor series expansions common in optical flow analyses [30, 22].

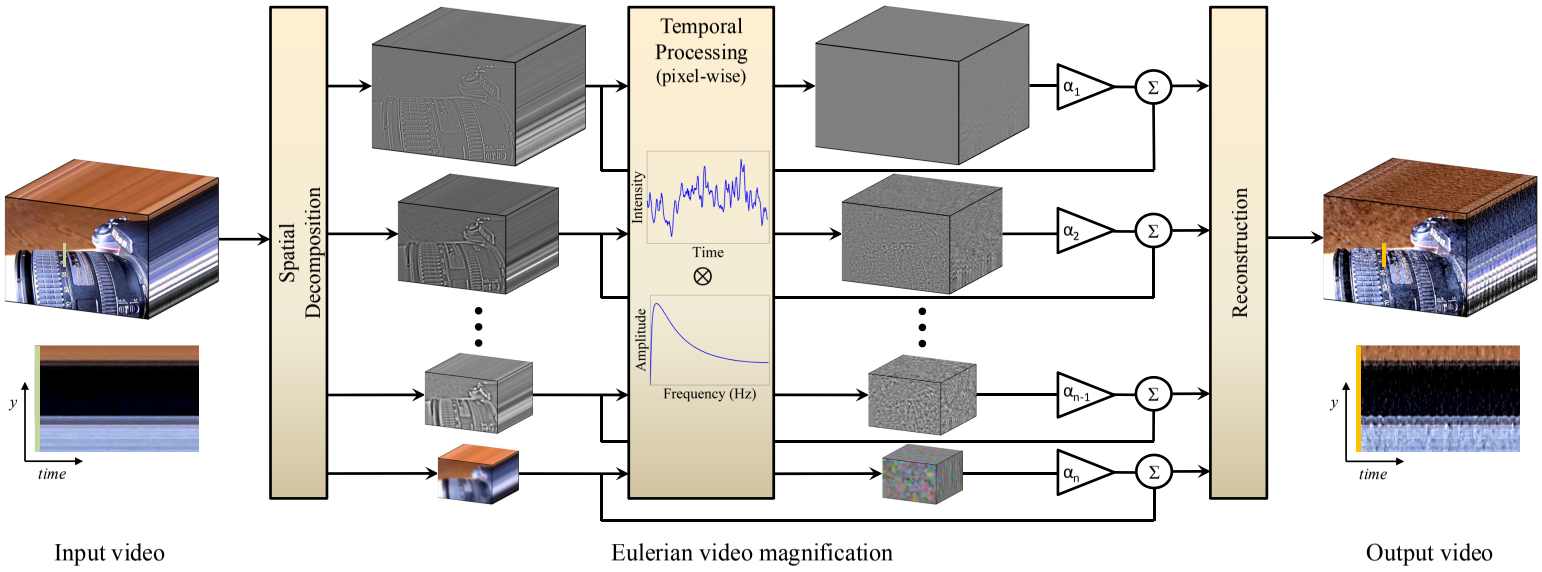


Figure 3.2: Overview of the Eulerian video magnification framework. The system first decomposes the input video sequence into different spatial frequency bands, and applies the same temporal filter to all bands. The filtered spatial bands are then amplified by a given factor α, added back to the original signal, and collapsed to generate the output video. The choice of temporal filter and amplification factors can be tuned to support different applications. For example, we use the system to reveal unseen motions of a Digital SLR camera, caused by the flipping mirror during a photo burst (*camera*; full sequences are available in the supplemental video).

First-order motion To explain the relationship between temporal processing and motion magnification, we consider the simple case of a 1D signal undergoing translational motion. This analysis generalizes directly to locally-translational motion in 2D.

Let $I(x,t)$ denote the image intensity at position x and time t. Since the image undergoes translational motion, we can express the observed intensities with respect to a displacement function $\delta(t)$, such that $I(x,t) = f(x+\delta(t))$ and $I(x,0) = f(x)$. The goal of motion magnification is to synthesize the signal

$$\hat{I}(x,t) = f(x+(1+\alpha)\delta(t)) \tag{3.1}$$

for some amplification factor α.

Assuming the image can be approximated by a first-order Taylor series expansion, we write the image at time t, $f(x+\delta(t))$ in a first-order Taylor expansion about x, as

$$I(x,t) \approx f(x) + \delta(t)\frac{\partial f(x)}{\partial x}. \tag{3.2}$$

Let $B(x,t)$ be the result of applying a broadband temporal bandpass filter to $I(x,t)$ at every position x (picking out everything except $f(x)$ in Equation 3.2). For now, let us assume the motion signal, $\delta(t)$, is within the passband of the temporal bandpass filter (we will relax that assumption later). Then we have

$$B(x,t) = \delta(t)\frac{\partial f(x)}{\partial x}. \tag{3.3}$$

In our process, we then amplify that bandpass signal by α and add it back to $I(x,t)$, resulting in the processed signal

$$\tilde{I}(x,t) = I(x,t) + \alpha B(x,t). \tag{3.4}$$

Combining Eqs. 3.2, 3.3, and 3.4, we have

$$\tilde{I}(x,t) \approx f(x) + (1+\alpha)\delta(t)\frac{\partial f(x)}{\partial x}. \tag{3.5}$$

Assuming the first-order Taylor expansion holds for the amplified larger perturbation, $(1+\alpha)\delta(t)$, we can relate the amplification of the temporally bandpassed signal to motion magnification. The processed output is simply

$$\tilde{I}(x,t) \approx f(x+(1+\alpha)\delta(t)). \tag{3.6}$$

This shows that the processing magnifies motions—the spatial displacement $\delta(t)$ of the local image $f(x)$ at time t, has been amplified to a magnitude of $(1+\alpha)$.

This process is illustrated for a single sinusoid in Figure 3.3. For a low frequency cosine wave and a relatively small displacement, δ, the first-order Taylor series expansion serves as a good approximation for the translated signal at time $t+1$. When boosting the temporal signal by α and adding it back to $I(x,t)$, we approximate that wave translated by $(1+\alpha)\delta$.

For completeness, let us return to the more general case where $\delta(t)$ is not entirely within the passband of the temporal filter. In this case, let $\delta_k(t)$, indexed by k, represent the different temporal spectral components of $\delta(t)$. Each $\delta_k(t)$ will be attenuated by the temporal filtering by a factor γ_k. This results in a bandpassed signal (compare with Equation 3.3),

$$B(x,t) = \sum_k \gamma_k \delta_k(t) \frac{\partial f(x)}{\partial x} \tag{3.7}$$

Because of the multiplication in Equation 3.4, this temporal frequency dependent attenuation can equivalently be interpreted as a frequency-dependent motion magnification factor, $\alpha_k = \gamma_k \alpha$, resulting in a motion magnified output,

$$\tilde{I}(x,t) \approx f(x + \sum_k (1+\alpha_k)\delta_k(t)) \tag{3.8}$$

The result is as would be expected for a linear analysis: the modulation of the spectral components of the motion signal becomes the modulation factor in the motion amplification factor, α_k, for each temporal subband, δ_k, of the motion signal.

Bounds In practice, the assumptions in Sect. 3.2.2 hold for smooth images and small motions. For quickly changing image functions (i.e., high spatial frequencies), $f(x)$, the first-order Taylor series approximations becomes inaccurate for large values of the perturbation, $1+\alpha\delta(t)$, which increases both with larger magnification α and motion $\delta(t)$. Figures 3.4 and 3.5 demonstrate the effect of higher frequencies, larger amplification factors and larger motions on the motion-amplified signal of a sinusoid.

As a function of spatial frequency, ω, we can derive a guide for how large the motion amplification factor, α, can be, given the observed motion $\delta(t)$. For the processed signal, $\tilde{I}(x,t)$ to be approximately equal to the true magnified motion, $\hat{I}(x,t)$, we seek the conditions under which

$$\begin{aligned} \tilde{I}(x,t) &\approx \hat{I}(x,t) \\ \Rightarrow f(x) + (1+\alpha)\delta(t)\frac{\partial f(x)}{\partial x} &\approx f(x + (1+\alpha)\delta(t)) \end{aligned} \tag{3.9}$$

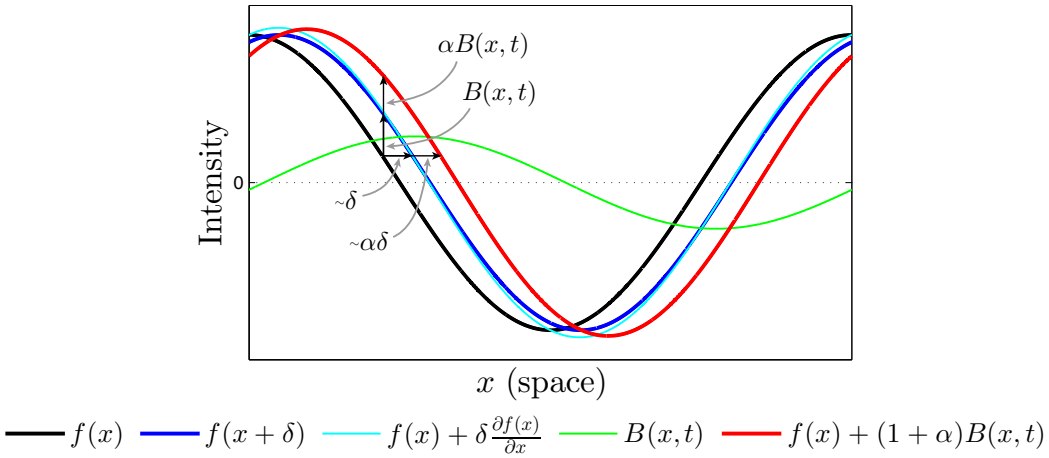


Figure 3.3: Temporal filtering can approximate spatial translation. This effect is demonstrated here on a 1D signal, but equally applies to 2D. The input signal is shown at two time instants: $I(x,t) = f(x)$ at time t and $I(x,t+1) = f(x+\delta)$ at time $t+1$. The first-order Taylor series expansion of $I(x,t+1)$ about x approximates well the translated signal. The temporal bandpass is amplified and added to the original signal to generate a larger translation. In this example $\alpha = 1$, magnifying the motion by 100%, and the temporal filter is a finite difference filter, subtracting the two curves.

Let $f(x) = \cos(\omega x)$ for spatial frequency ω, and denote $\beta = 1 + \alpha$. We require that

$$\cos(\omega x) - \beta\omega\delta(t)\sin(\omega x) \approx \cos(\omega x + \beta\omega\delta(t)) \tag{3.10}$$

Using the addition law for cosines, we have

$$\begin{aligned}\cos(\omega x) - \beta\omega\delta(t)\sin(\omega x) = \\ \cos(\omega x)\cos(\beta\omega\delta(t)) - \sin(\omega x)\sin(\beta\omega\delta(t))\end{aligned} \tag{3.11}$$

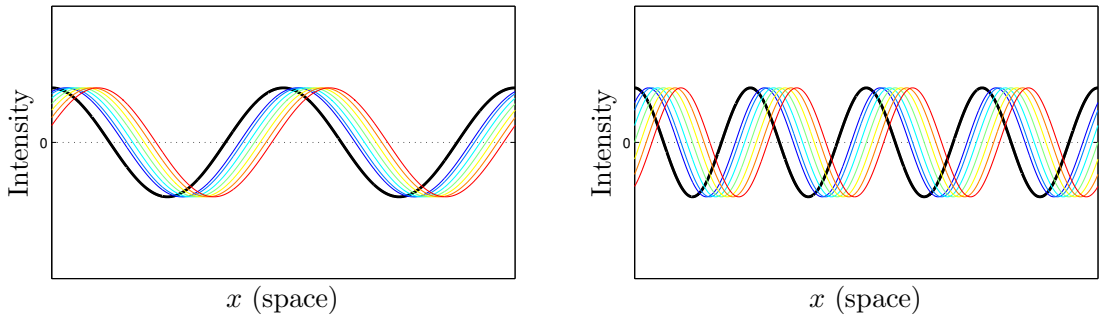
Hence, the following should approximately hold

$$\cos(\beta\omega\delta(t)) \approx 1 \tag{3.12}$$

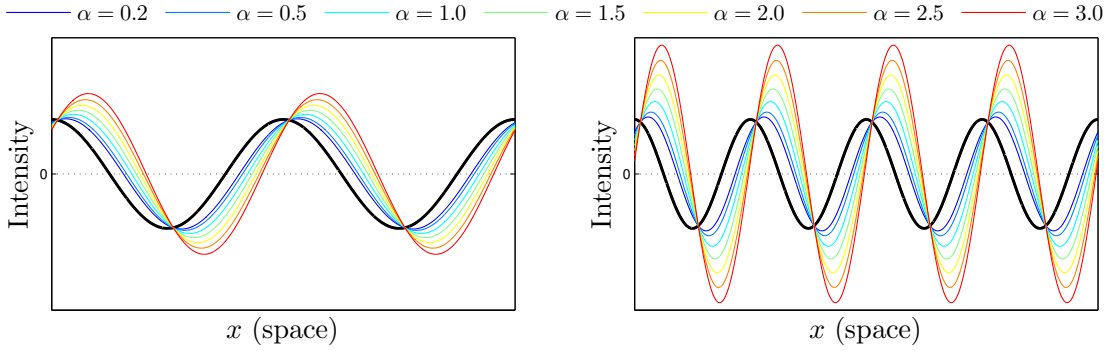
$$\sin(\beta\omega\delta(t)) \approx \beta\delta(t)\omega \tag{3.13}$$

The small angle approximations of Eqs. (3.12) and (3.13) will hold to within 10% for $\beta\omega\delta(t) \leq \frac{\pi}{4}$ (the sine term is the leading approximation and we have $\sin(\frac{\pi}{4}) = 0.9\frac{\pi}{4}$). In terms of the spatial wavelength, $\lambda = \frac{2\pi}{\omega}$, of the moving signal, this gives

$$(1+\alpha)\delta(t) < \frac{\lambda}{8}. \tag{3.14}$$



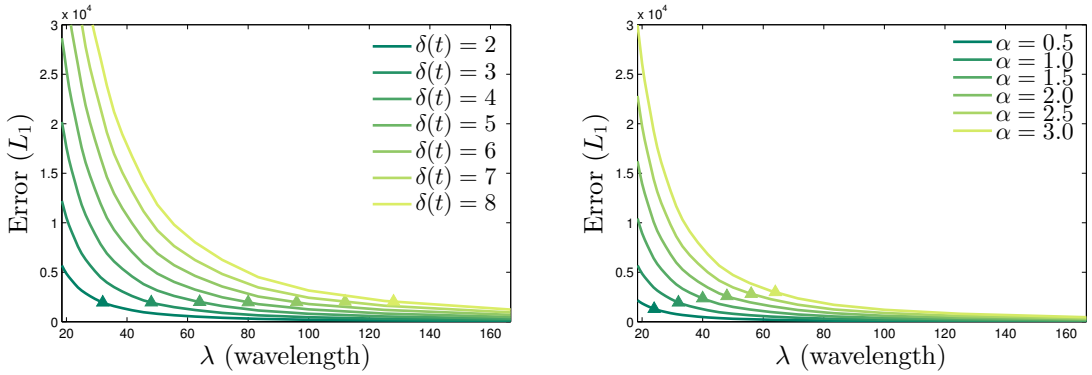
(a) True motion amplification: $\hat{I}(x,t) = f(x + (1+\alpha)\delta(t))$.



(b) Motion amplification via temporal filtering: $\tilde{I}(x,t) = I(x,t) + \alpha B(x,t)$.

Figure 3.4: Illustration of motion amplification on a 1D signal for different spatial frequencies and α values. For the images on the left side, $\lambda = 2\pi$ and $\delta(1) = \frac{\pi}{8}$ is the true translation. For the images on the right side, $\lambda = \pi$ and $\delta(1) = \frac{\pi}{8}$. (a) The true displacement of $I(x,0)$ by $(1+\alpha)\delta(t)$ at time $t = 1$, colored from blue (small amplification factor) to red (high amplification factor). (b) The amplified displacement produced by our filter, with colors corresponding to the correctly shifted signals in (a). Referencing Equation 3.14, the red (far right) curves of each plot correspond to $(1+\alpha)\delta(t) = \frac{\lambda}{4}$ for the left plot, and $(1+\alpha)\delta(t) = \frac{\lambda}{2}$ for the right plot, showing the mild, then severe, artifacts introduced in the motion magnification from exceeding the bound on $(1+\alpha)$ by factors of 2 and 4, respectively.

Equation 3.14 above provides the guideline we seek, giving the largest motion amplification factor, α, compatible with accurate motion magnification of a given video motion $\delta(t)$ and image structure spatial wavelength, λ. Figure 3.4 (b) shows the motion magnification errors for a sinusoid when we boost α beyond the limit in Equation 3.14. In some videos, violating the approximation limit can be perceptually preferred and we leave the λ cutoff as a user-modifiable parameter in the multiscale processing.



(a) (b)

Figure 3.5: Motion magnification error, computed as the L_1-norm between the true motion-amplified signal (Figure 3.4(a)) and the temporally-filtered result (Figure 3.4(b)), as function of wavelength, for different values of $\delta(t)$ (a) and α (b). In (a), we fix $\alpha = 1$, and in (b), $\delta(t) = 2$. The markers on each curve represent the derived cutoff point $(1 + \alpha)\delta(t) = \frac{\lambda}{8}$ (Equation 3.14).

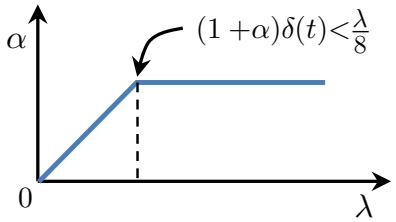


Figure 3.6: Amplification factor, α, as function of spatial wavelength λ, for amplifying motion. The amplification factor is fixed to α for spatial bands that are within our derived bound (Equation 3.14), and is attenuated linearly for higher spatial frequencies.

Multiscale analysis The analysis in Sect. 3.2.2 suggests a *scale-varying* process: use a specified α magnification factor over some desired band of spatial frequencies, then scale back for the high spatial frequencies (found from Equation 3.14 or specified by the user) where amplification would give undesirable artifacts. Figure 3.6 shows such a modulation scheme for α. The spatial frequency content of the different levels of the Laplacian pyramid can be estimated as shown in Figure 3.7. Although areas of high spatial frequencies (sharp edges) will be generally amplified less than lower frequencies, we found the resulting videos to contain perceptually appealing magnified motion. Such effect was also exploited in the earlier work of Freeman et al. [16] to create the illusion of motion from still images.

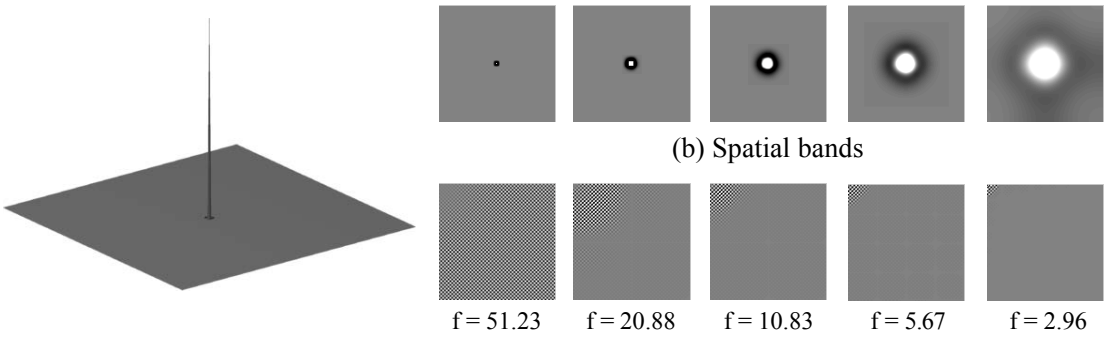


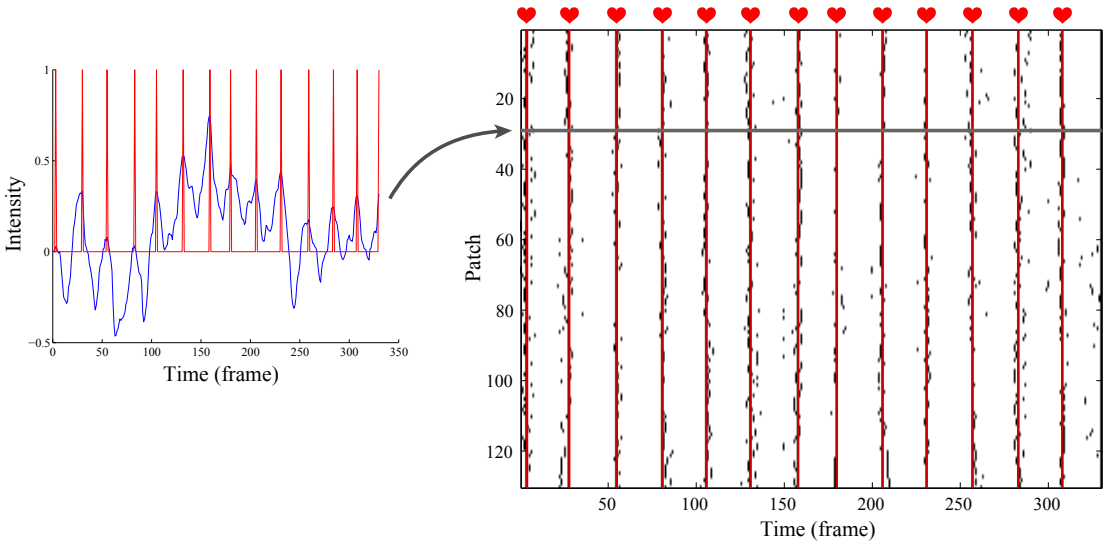
Figure 3.7: Spatial frequencies of the Laplacian pyramid. To estimate the spatial frequencies at each pyramid level, we decompose an impulse image (a) to its spatial frequency bands (b), and compute the DCT coefficients for each band (c). We estimate the spatial frequency for each band (given below the DCT coefficients in (c)) as the average magnitude of its corresponding DCT coefficients, weighted by their distance from the origin (upper left corner in (c)).

■ 3.2.3 Pulse and Heart Rate Extraction

Our Eulerian processing can also be useful for *quantitative* analysis of temporal variation. In particular, although not the focus of this thesis, we show how we can utilize the subtle color variation of the human skin due to blood flow in order to extract heart rate (HR) measurements from standard RGB video.

State-of-the-art techniques for measuring cardiac pulse either use electrocardiogram (ECG, *a.k.a.* EKG) or pulse oximetry, which require the patient to either wear patches or attach an external device. Measuring the pulse without direct contact can be useful in a variety of cases when attaching patches or devices may be undesirable or impossible. Those include remote medical diagnosis and checkups, vital-sign monitoring in prenatal babies, and search and rescue operations where the subject may be out of reach. This has led to recent attempts in both the academia and industry to create video-based solutions for contactless measurement of HR and other vital signs [36, 35, 32].

As we have seen earlier, the color of the skin changes slightly with blood flow. When recorded by video, we found this signal corresponds to roughly 0.5 intensity units in an 8-bit scale (each pixel records an intensity value within $0 - 255$) . This subtle signal can be reliably extracted with our Eulerian processing, to visualize spatial blood flow patterns as previously discussed, and also to recover the actual pulse signal and HR of the subject being recorded.



(a) Bandpassed signal and detected peaks (b) Point-wise pulse onsets and detected pulse locations

Figure 3.8: Heart rate extraction. (a) The spatially-averaged, temporally-bandpassed signal from one point on the face in the *face* video from Figure 3.1, with the local maxima marked, indicating the pulse onsets. (b) Pulse locations used to estimate the pulse signal and heart rate. Each row on the y-axis corresponds to a different point on the face in no particular ordering. Black pixels represent the detected peak locations at each point, and the red vertical lines correspond to the final estimated pulse locations that are used to compute the heart rate.

Our process works as follows. We apply the same spatial decomposition described in Section 3.2.1 using a Gaussian pyramid of $3-4$ levels, and filter the signal at the coarse pyramid level temporally selecting frequencies between $0.4-4$Hz (corresponding to $24-240$ beats per minute). The spatial pooling of the Gaussian pyramid averages over enough pixels to reveal the underlying pulse in the signal, and the temporal filter further removes some of the noise.

The instantaneous HR can be determined by a frequency analysis of the color variation in the extracted bandpassed signal, however we also explored a basic method that can recover the actual pulse signal over time (and from it the HR), which we will describe next. This preliminary method was later extended by [58] to more rigorously account for the onset differences when determining the HR.

Specifically, we detect the pulse onset times at each location on the face in the coarse pyramid level, by finding maxima points in the bandpassed signal within a local temporal window. The size of the temporal window can be chosen according to the detected frequency of the

color variation in the bandpassed signal, however we generally found that a temporal window of $10 - 15$ frames for a 30 frames-per-second video produced good results. In figure 3.8 we show the bandpassed signal with its detected peak positions for one point from a smooth region on the face in the *face* video (Figure 3.1).

Typically, the quality of the pulse signal extracted from different points on the face may vary. This is both because the amplitude of the signal may be modulated according to the density of peripheral blood vessels, as well as due to high spatial-contrast regions of the body or face from which it is harder to extract the signal. There can also be errors in detecting the peaks in the bandpassed signal.

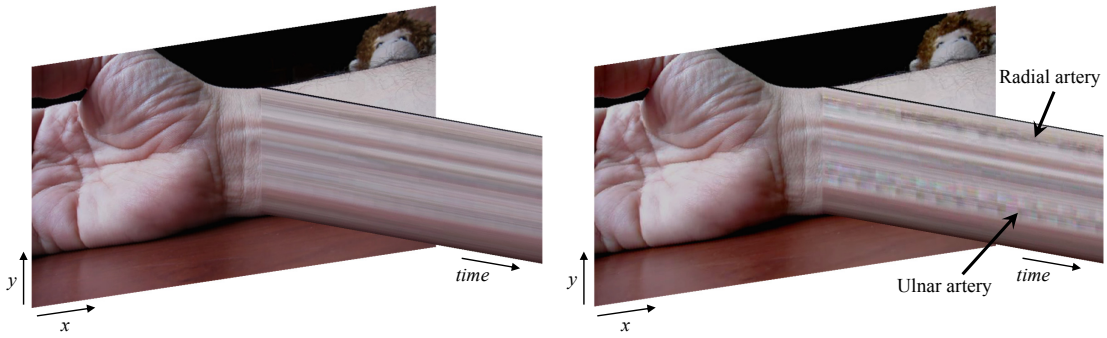
Thus, to get a robust estimate of the pulse onset times we integrate the signal extracted from the different locations in the video. We ignore all the points for which the temporal frequency deviates sufficiently (parameter) from the dominant temporal frequency in the face region (the face region can be provided by a user, or identified automatically using a face detector). We then consider the rest of the points as containing a pulse signal, and determine the pulse locations as the local (temporally) centers of mass of the detected peaks in all the points (Figure 3.8(b)). Notice that even though the pulse estimation at particular points on the face may be noisy, combining the estimation from multiple points provides a robust estimation of the pulse signal. Finally, from the estimated pulse signal we directly establish the HR as the number of pulses within a small temporal window (we used a 5 seconds window).

An important difference between our technique and that of Poh et al. [36] is that they average the pixel values over the entire face region, while in our approach only pulsatile regions on the face will be used for the HR estimation. In fact, we noticed empirically that many pixels on the face, especially those within high spatial-contract regions, do not provide reliable cardiac signal (Figure 3.29). Thus, averaging over those regions, as done by Poh et al., may reduce the pulse SNR and the accuracy of the estimated HR.

■ 3.2.4 Results

The results were generated using non-optimized MATLAB code on a machine with a six-core processor and 32 GB RAM. The computation time per video was on the order of a few minutes. We used a separable *binomial filter* of size five to construct the video pyramids.

To process an input video by Eulerian video magnification, there are four steps a user needs to take: (1) select a temporal bandpass filter; (2) select an amplification factor, α; (3) select a spatial frequency cutoff (specified by spatial wavelength, λ_c) beyond which an attenuated



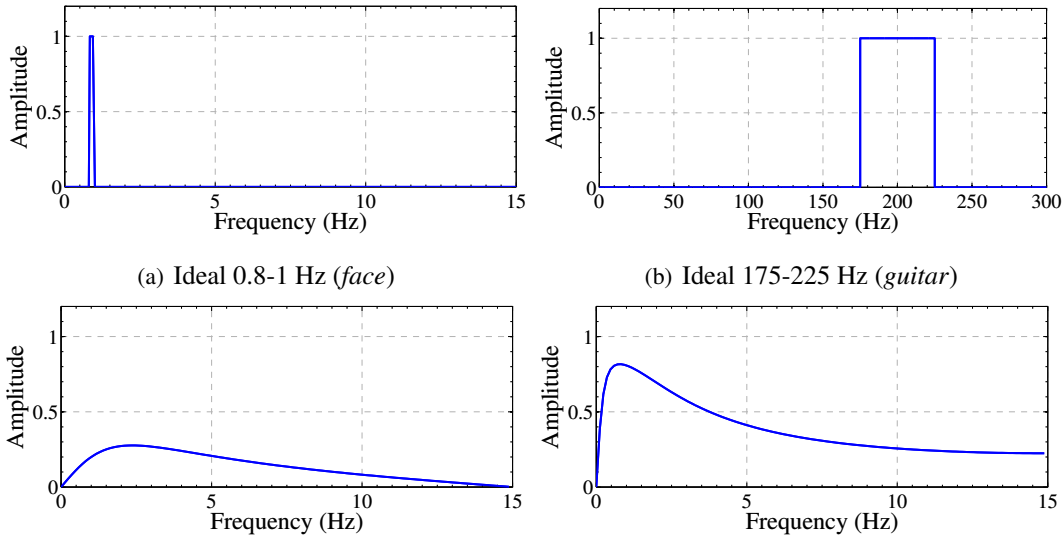
(a) Input (*wrist*) (b) Motion-amplified

Figure 3.9: Eulerian video magnification used to amplify subtle motions of blood vessels arising from blood flow. For this video, we tuned the temporal filter to a frequency band that includes the heart rate—0.88 Hz (53 bpm)—and set the amplification factor to $\alpha = 10$. To reduce motion magnification of irrelevant objects, we applied a user-given mask to amplify the area near the wrist only. Movement of the radial and ulnar arteries can barely be seen in the input video (a) taken with a standard point-and-shoot camera, but is significantly more noticeable in the motion-magnified output (b). The motion of the pulsing arteries is more visible when observing a spatio-temporal YT slice of the wrist (a) and (b). The full *wrist* sequence can be found in the supplemental video.

version of α is used; and (4) select the form of the attenuation for α—either force α to zero for all $\lambda < \lambda_c$, or linearly scale α down to zero. The frequency band of interest can be chosen automatically in some cases, but it is often important for users to be able to control the frequency band corresponding to their application.

We first select the temporal bandpass filter to pull out the motions or signals that we wish to be amplified (step 1 above). The choice of filter is generally application dependent. For motion magnification, a filter with a broad passband is preferred; for color amplification of blood flow, a narrow passband produces a more noise-free result. Figure 3.10 shows the frequency responses of some of the temporal filters used in this paper. We use ideal bandpass filters for color amplification, since they have passbands with sharp cutoff frequencies. Low-order IIR filters can be useful for both color amplification and motion magnification and are convenient for a real-time implementation. In general, we used two first-order lowpass IIR filters with cutoff frequencies ω_l and ω_h to construct an IIR bandpass filter.

Next, we select the desired magnification value, α, and spatial frequency cutoff, λ_c (steps 2 and 3). While Equation 3.14 can be used as a guide, in practice, we may try various α and λ_c values to achieve a desired result. Users can select a higher α that violates the bound to



(a) Ideal 0.8-1 Hz (*face*)

(b) Ideal 175-225 Hz (*guitar*)

(c) Butterworth 3.6-6.2 Hz (*subway*)

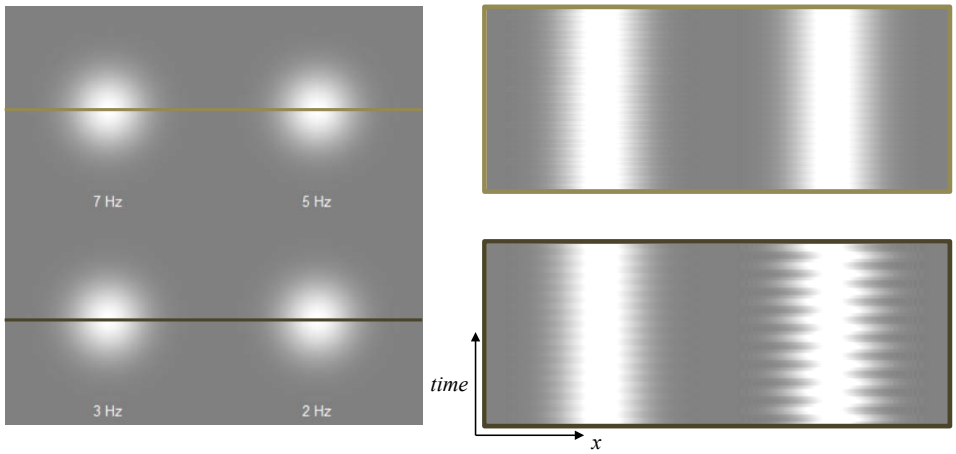
(d) Second-order IIR (pulse detection)

Figure 3.10: Temporal filters used in the thesis. The ideal filters (a) and (b) are implemented using DCT. The Butterworth filter (c) is used to convert a user-specified frequency band to a second-order IIR structure that can be used for real-time processing (Section 3.4). The second-order IIR filter (d) also allows user input. These second-order filters have a broader passband than an ideal filter.

exaggerate specific motions or color changes at the cost of increasing noise or introducing more artifacts. In some cases, one can account for color clipping artifacts by attenuating the chrominance components of each frame. Our approach achieves this by doing all the processing in the YIQ space. Users can attenuate the chrominance components, I and Q, before conversion to the original color space.

For human pulse color amplification, where we seek to emphasize low spatial frequency changes, we may force $\alpha = 0$ for spatial wavelengths below λ_c. For motion magnification videos, we can choose to use a linear ramp transition for α (step 4).

Color Amplification We evaluated our method for color amplification using a few videos: two videos of adults with different skin colors and one of a newborn baby. An adult subject with lighter complexion is shown in *face* (Figure 3.1), while an individual with darker complexion is shown in *face2* (Table C.1). In both videos, our objective was to amplify the color change as the blood flows through the face. In both *face* and *face2*, we applied a Laplacian pyramid and set α for the finest two levels to 0. Essentially, we downsampled and applied a spatial lowpass filter

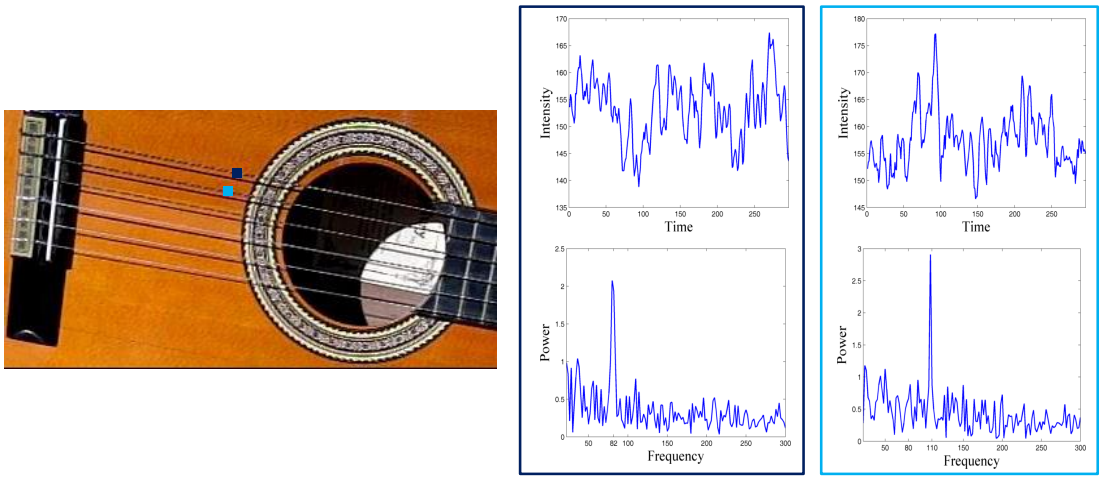


(a) Input (*sim4*) (b) Motion-amplified spatiotemporal slices

Figure 3.11: Selective motion amplification on a synthetic sequence (*sim4* on left). The video sequence contains blobs oscillating at different temporal frequencies as shown on the input frame. We apply our method using an ideal temporal bandpass filter of 1-3 Hz to amplify only the motions occurring within the specified passband. In (b), we show the spatio-temporal slices from the resulting video which show the different temporal frequencies and the amplified motion of the blob oscillating at 2 Hz. We note that the space-time processing is applied uniformly to all the pixels. The full sequence and result can be found in the supplemental video.

to each frame to reduce both quantization and noise and to boost the subtle pulse signal that we are interested in. For each video, we then passed each sequence of frames through an ideal bandpass filter with a passband of 0.83 Hz to 1 Hz (50 bpm to 60 bpm). Finally, a large value of $\alpha \approx 100$ and $\lambda_c \approx 1000$ was applied to the resulting spatially lowpass signal to emphasize the color change as much as possible. The final video was formed by adding this signal back to the original. We see periodic green to red variations at the heart rate and how blood perfuses the face.

Heart Rate Extraction We demonstrate heart rate measurement on a video of a newborn recorded *in situ* at the Nursery Department at Winchester Hospital in Massachusetts (*baby2*). The video was taken with a standard DSLR camera, and we obtained ground truth vital signs from a hospital-grade monitor that was recorded in sync with the video. We used this information to confirm the accuracy of our heart rate estimate and to verify that the color amplification signal extracted from our method matches the photoplethysmogram, an optically obtained measurement of the perfusion of blood to the skin, as measured by the monitor. This video is part of



(a) Input (*guitar*) (b) Low E (82.4 Hz) (c) A (100 Hz)

Figure 3.12: Selective motion amplification on a natural video (*guitar*). Each of the guitar's strings (a) vibrates at different frequency. (b-c) The signals of intensities over time (top) and their corresponding power spectra (bottom) are shown for two pixels located on the low E and A strings, respectively, superimposed on the representative frame in (a). The power spectra clearly reveal the vibration frequency of each string. Using an appropriate temporal bandpass filter, we are able to amplify the motion of particular strings while maintaining the motion of the others. The result is available in the supplemental video.

a larger study done by [58] involving several prenatal babies, and we refer the interested reader to their paper for more details and a comprehensive evaluation.

Motion Amplification To evaluate our method for motion magnification, we used several different videos: *face* (Figure 3.1), *sim4* (Figure 3.11), *wrist* (Figure 3.9), *camera* (Figure 3.2), *face2*, *guitar*, *baby*, *subway*, *shadow*, and *baby2* (Table C.1). For all videos, we used a standard Laplacian pyramid for spatial filtering. For videos where we wanted to emphasize motions at specific temporal frequencies (e.g., in *sim4* and *guitar*), we used ideal bandpass filters. In *sim4* and *guitar*, we were able to selectively amplify the motion of a specific blob or guitar string by using a bandpass filter tuned to the oscillation frequency of the object of interest. These effects can be observed in the supplemental video. The values used for α and λ_c for all of the videos discussed in this paper are shown in Table 3.1.

For videos where we were interested in revealing broad, but subtle motion, we used temporal filters with a broader passband. For example, for the *face2* video, we used a second-order

IIR filter with slow roll-off regions. By changing the temporal filter, we were able to magnify the motion of the head rather than amplify the change in the skin color. Accordingly, $\alpha = 20, \lambda_c = 80$ were chosen to magnify the motion.

By using broadband temporal filters and setting α and λ_c according to Equation 3.14, our method is able to reveal subtle motions, as in the *camera* and *wrist* videos. For the *camera* video, we used a camera with a sampling rate of 300 Hz to record a Digital SLR camera vibrating while capturing photos at about one exposure per second. The vibration caused by the moving mirror in the SLR, though invisible to the naked eye, was revealed by our approach. To verify that we indeed amplified the vibrations caused by the flipping mirror, we secured a laser pointer to the camera and recorded a video of the laser light, appearing at a distance of about four meters from the source. At that distance, the laser light visibly oscillated with each exposure, with the oscillations in sync with the magnified motions.

Our method is also able to exaggerate visible, yet subtle motion, as seen in the *baby*, *face2*, and *subway* videos. In the subway example we deliberately amplified the motion beyond the derived bounds of where the first-order approximation holds in order to increase the effect and to demonstrate the algorithm's artifacts. We note that most of the examples in our paper contain oscillatory movements because such motion generally has longer duration and smaller amplitudes. However, our method can be used to amplify non-periodic motions as well, as long as they are within the passband of the temporal bandpass filter. In *shadow*, for example, we process a video of the sun's shadow moving linearly yet imperceptibly over 15 seconds. The magnified version makes it possible to see the change even within this short time period.

Finally, some videos may contain regions of temporal signals that do not need amplification, or that, when amplified, are perceptually unappealing. Due to our Eulerian processing, we can easly allow the user to manually restrict magnification to particular areas by marking them on the video (this was used for *face* and *wrist*).

■ 3.2.5 Sensitivity to Noise.

The amplitude variation of the signal of interest is often much smaller than the noise inherent in the video. In such cases direct enhancement of the pixel values will not reveal the desired signal. Spatial filtering can be used to enhance these subtle signals. However, if the spatial filter applied is not large enough, the signal of interest will not be revealed (Figure 3.13).

Assuming that the noise is zero-mean white and wide-sense stationary with respect to space, it can be shown that spatial low pass filtering reduces the variance of the noise according to the

Video	α	λ_c	ω_l (Hz)	ω_h (Hz)	f_s (Hz)
baby	10	16	0.4	3	30
baby2	150	600	2.33	2.67	30
camera	120	20	45	100	300
face	100	1000	0.83	1	30
face2 motion	20	80	0.83	1	30
face2 pulse	120	960	0.83	1	30
guitar Low E	50	40	72	92	600
guitar A	100	40	100	120	600
shadow	5	48	0.5	10	30
subway	60	90	3.6	6.2	30
wrist	10	80	0.4	3	30

Table 3.1: Table of $\alpha, \lambda_c, \omega_l, \omega_h$ values used to produce the various output videos. For *face2*, two different sets of parameters are used—one for amplifying pulse, another for amplifying motion. For *guitar*, different cutoff frequencies and values for (α, λ_c) are used to "select" the different oscillating guitar strings. f_s is the frame rate of the camera.

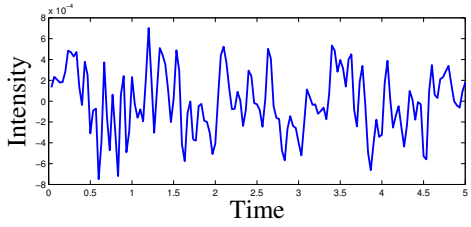
area of the low pass filter. To demonstrate this, let σ^2 be the noise variance in the original image, and $N(i,j)$ be the noise at position (i,j). Also let $g(i,j)$ be the filter value at position (i,j), such that $\sum_{i,j} g(i,j) = 1$. Then the noise variance in the spatially filtered image, σ'^2, can be expressed as

$$\begin{aligned}
\sigma'^2 &= E\left[N'(k,l)^2\right] - \left(E\left[N'(k,l)\right]\right)^2 \\
&= E\left[\left(\sum_{i,j} g(i,j)N(i-k,j-l)\right)^2\right] - 0 \\
&= \sum_{i,j} g(i,j)^2 E[N(i-k,j-l))^2] \\
&= \sigma^2 \sum_{i,j} g(i,j)^2. \qquad (3.15)
\end{aligned}$$

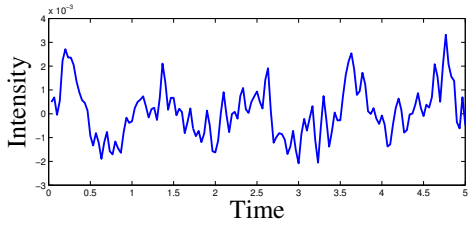
For this low pass filter, the noise power level is thus inversely proportional to the area of the filter.

In order to boost the power of a specific signal, e.g., the pulse signal in the face, we can use the spatial characteristics of the signal to estimate the spatial filter size. Assuming our prior

(a) Input noisy frame



(b) Insufficient spatial pooling



(c) Sufficient spatial pooling

Figure 3.13: Proper spatial pooling is imperative for revealing the signal of interest. (a) A frame from the *face* video (Figure 3.1) with white Gaussian noise added ($\sigma = 0.1$). On the right are intensity traces over time for the pixel marked blue on the input frame, where (b) shows the trace obtained when the (noisy) sequence is processed with the same spatial filter used to process the original *face* sequence, a separable binomial filter of size 20, and (c) shows the trace when using a filter tuned according to the estimated radius in Equation 3.16, a binomial filter of size 80. The pulse signal is not visible in (b), as the noise level is higher than the power of the signal, while in (c) the pulse is clearly visible (the periodic peaks about one second apart in the trace).

on signal power over spatial wavelengths is $S(\lambda)$, we want to find a spatial low pass filter with radius r such that the signal power is greater than the noise in the filtered frequency region. The wavelength cut off of such a filter is proportional to its radius, r, so the signal prior can be represented as $S(r)$. The noise power, σ^2, can be estimated by examining pixel values in a stable region of the scene, from a gray card, or by using a technique as in [25]. Since the filtered noise power level, σ'^2, is inversely proportional to r^2, we can solve the following equation for r,

$$S(r) = \sigma'^2 = k\frac{\sigma^2}{r^2} \tag{3.16}$$

where k is a constant that depends on the shape of the low pass filter. This equation gives an estimate for the size of the spatial filter needed to reveal the signal at a certain noise power level.

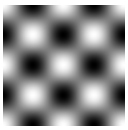
■ 3.2.6 Eulerian vs. Lagrangian Processing.

Because the two methods take different approaches to motion—Lagrangian approaches explicitly track motions, while our Eulerian approach does not—they can be used for complementary motion domains. Lagrangian approaches, e.g. [26], work better to enhance motions of fine point features and support larger amplification factors, while our Eulerian method is better suited to smoother structures and small amplifications. We note that our technique does not assume particular types of motions. The first-order Taylor series analysis can hold for general small 2D motions along general paths.

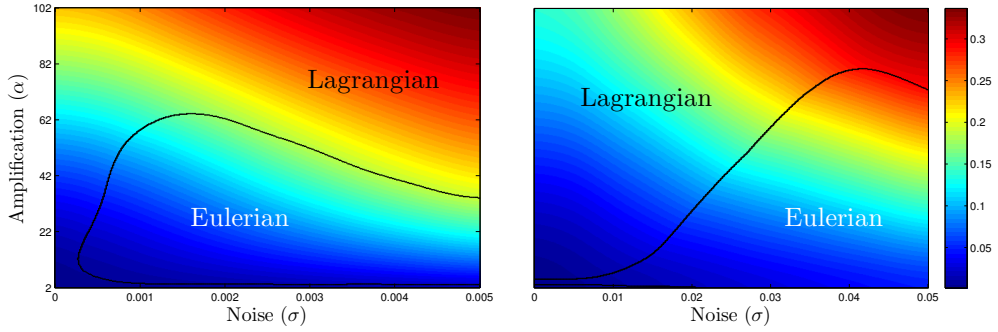
In Appendix A, we further derive estimates of the accuracy of the two approaches with respect to noise[2]. Comparing the Lagrangian error, ε_L (Equation A.18), and the Eulerian error, ε_E (Equation A.20), we see that both methods are equally sensitive to the temporal characteristics of the noise, n_t, while the Lagrangian process has additional error terms proportional to the spatial characteristics of the noise, n_x, due to the explicit estimation of the motion (Equation A.15). The Eulerian error, on the other hand, grows quadratically with α, and is more sensitive to high spatial frequencies (I_{xx}). In general, this means that Eulerian magnification would be preferable over Lagrangian magnification for small amplifications and larger noise levels.

We validated this analysis on a synthetic sequence of a 2D cosine oscillating at 2 Hz temporally and 0.1 pixels spatially with additive white spatiotemporal Gaussian noise of zero mean and standard deviation σ (Figure 3.14). The results match the error-to-noise and error-to-amplification relationships predicted by the derivation (Figure 3.14(c-d)). The region where the Eulerian approach outperforms the Lagrangian results (Figure3.14(b), left) is also as expected. The Lagrangian method is more sensitive to increases in spatial noise, while the Eulerian error is hardly affected by it (Figure 3.14(d)). While different regularization schemes used for motion estimation (that are harder to analyze theoretically) may alleviate the Lagrangian error, they did not change the result significantly (Figure 3.14(b), right). In general, our experiments show that for small amplifications the Eulerian approach strikes a better balance between performance and efficiency. Comparisons between the methods on natural videos are available on the project webpage.

[2]This is a more detailed version of the derivation we gave in Appendix A in our SIGGRAPH 2012 paper [59].

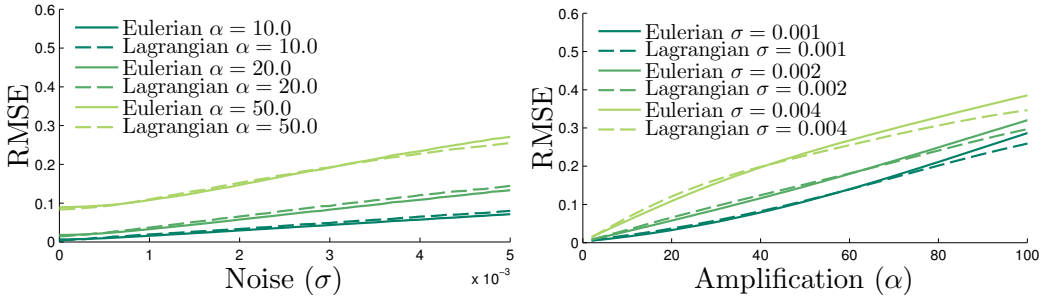


(a) A frame from the synthetic input sequence.

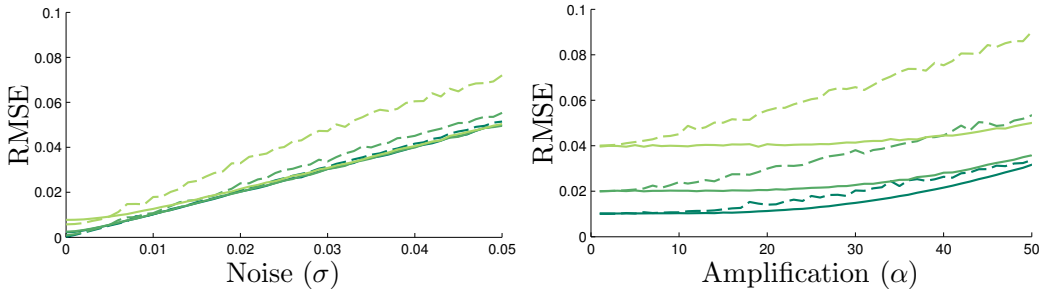


No regularization With regularization

(b) $\min(\varepsilon_E, \varepsilon_L)$, spatiotemporal noise



(c) Error as function of σ and α, Spatiotemporal noise



(d) Error as function of σ and α, Spatial noise only

Figure 3.14: Comparison between Eulerian and Lagrangian motion magnification on a synthetic sequence with additive noise (a). (b) The minimal error, $\min(\varepsilon_E, \varepsilon_L)$, computed as the (frame-wise) RMSE between each method's result and the true motion-magnified sequence, as function of noise and amplification, colored from blue (small error) to red (large error), with (left) and without (right) spatial regularization in the Lagrangian method. The black curves mark the intersection between the error surfaces, and the overlayed text indicate the best performing method in each region. (c) RMSE of the two approaches as function of noise (left) and amplification (right). (d) Same as (c), using spatial noise only.

■ 3.3 Phase-Based Video Motion Processing

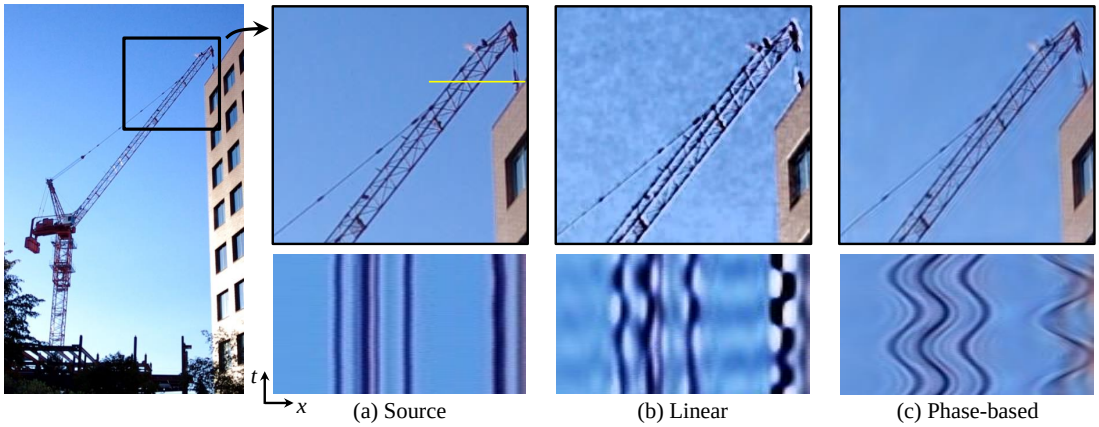
The Eulerian method we presented in Section 3.2, which we have shown produces a *linear* approximation to the magnified motion, is simple and fast, but suffers from two main drawbacks: (a) it supports relatively small magnification factors, and (b) it can significantly amplify noise when the magnification factor is increased (Figure 3.15(b)).

To counter these issues, we explored another Eulerian approach to motion processing, which is based on complex-valued steerable pyramids [48, 37], and inspired by phase-based optical flow [15, 19] and motion without movement [16]. Just as the phase variations of Fourier basis functions (sine waves) are related to translation via the the Fourier shift theorem, the phase variations of the complex steerable pyramid correspond to local motions in spatial subbands of an image. We compute the local phase variations to measure motion without explicit optical flow computation, and perform temporal processing to amplify motion in selected temporal frequency bands, and then reconstruct the modified video.

In this section, we show how this *phase-based* technique improves on the previous, *linear* Eulerian magnification method in two important aspects (Figure 3.15): (a) it supports larger magnification, and (b) it has substantially better noise performance. Because the linear method amplifies temporal brightness changes, the amplitude of noise is amplified linearly. In contrast, the method we propose here modifies phases, not amplitudes, which does not increase the magnitude of spatial noise.

We start from the relation between motion and phase in steerable pyramids and show that by increasing the phase variations by a multiplicative factor we can amplify subtle motions. We then use this relation to analyze the limits of the method, which are set by the spatial support of the steerable basis functions. To amplify motions further, we extend the complex steerable pyramid to *sub-octave bandwidth pyramids*, comprised of filters with larger spatial support in the primal domain. While this new image representation is over-complete by a larger factor, it supports larger amplification of motions at all spatial frequencies, leading to fewer artifacts. This expands the set of small-scale physical phenomena that can be visualized with Eulerian motion magnification techniques.

The extension to sub-octave bandwidth pyramids is mostly due to Wadhwa [54]. We briefly describe it here for completeness (Section 3.3.5, Appendix B), and refer the interested reader to his thesis for a more comprehensive discussion of sub-octave pyramid construction and other implementation details.



(a) Source (b) Linear (c) Phase-based

Figure 3.15: Motion magnification of a crane imperceptibly swaying in the wind. (a) Top: a zoom-in onto a patch in the original sequence (*crane*) shown on the left. Bottom: a spatiotemporal XT slice of the video along the profile marked on the zoomed-in patch. (b-c) Linear (Section 3.2) and phase-based motion magnification results, respectively, shown for the corresponding patch and spatiotemporal slice as in (a). The previous, linear method visualizes the crane's motion, but amplifies both signal and noise and introduces artifacts for higher spatial frequencies and larger motions, shown by the clipped intensities (bright pixels) in (b). In comparison, our new phase-based method supports larger magnification factors with significantly fewer artifacts and less noise (c). The full sequences are available in the supplemental video.

■ 3.3.1 Background

Phase-based Optical Flow. Fleet and Jepson [15] tracked constant phase contours by computing the phase gradient of a spatio-temporally bandpassed video, and showed that it provides a good approximation to the motion field, and that phase is more robust than amplitude to image changes due to contrast and scale. Gautama and Van Hulle [19] used a similar technique in which they computed the temporal gradient of the phases of a spatially bandpassed video to estimate the motion field. We build on this link between phase and motion, but seek to avoid the explicit computation of flow vectors, and instead directly manipulate the phase variations in videos.

Complex Steerable Pyramids. The steerable pyramid [48, 47] is an overcomplete transform that decomposes an image according to spatial scale, orientation, and position. The basis functions of the transform resemble Gabor wavelets, sinusoids windowed by a Gaussian envelope, and are steerable. We don't exploit the steerability of those basis functions in this work, but

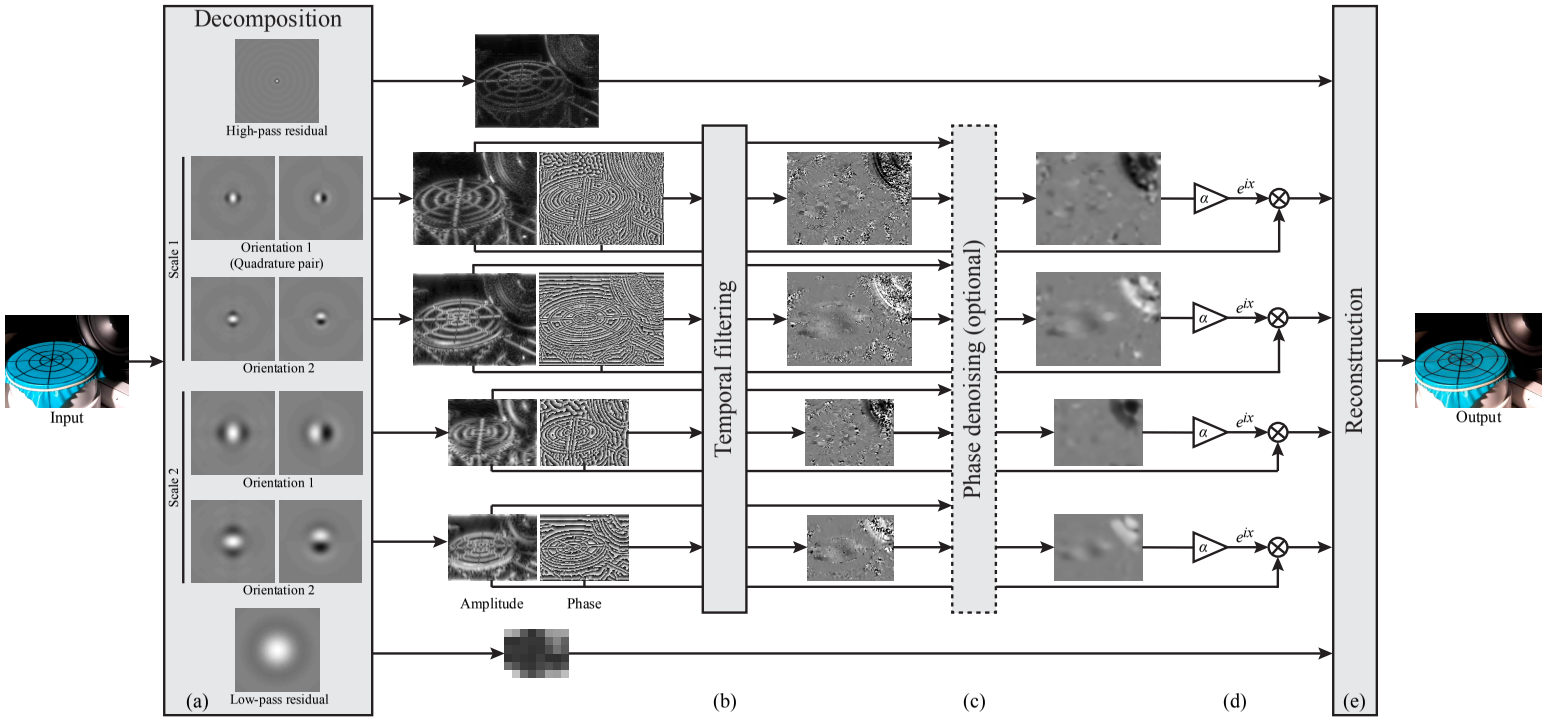


Figure 3.16: Our phase-based approach manipulates motion in videos by analyzing the signals of local phase over time in different spatial scales and orientations. We use complex steerable pyramids to decompose the video and separate the amplitude of the local wavelets from their phase (a). We then temporally filter the phases independently at at each location, orientation and scale (b). Optionally, we apply amplitude-weighted spatial smoothing (c, Sect. 3.3.6) to increase the phase SNR, which we empirically found to improve the results. We then amplify or attenuate the temporally-bandpassed phases (d), and reconstruct the video (e). This example shows the processing pipeline for the *membrane* sequence (Sect. 3.3.7), using a pyramid of two scales and two orientations (the relative difference in size between the pyramid levels is smaller in this figure for clarity of the visualization).

the transform has other properties which are important for our motion analysis: non-aliased subbands and quadrature phase filters.

We measure phase within each subband using the pairs of even and odd-phase oriented spatial filters whose outputs are the complex-valued coefficients in the steerable pyramid [48], The sub-sampling scheme of the steerable pyramid avoids spatial aliasing and thus allows meaningful signal phase measurements from the coefficients of the pyramid.The real part of each coefficient represents the even-symmetric filter (cosine), while its imaginary counterpart represents an odd-symmetric filter (sine). While twice as over-complete as a real-valued pyramid, the complex-valued pyramid allows simple measurement of local amplitude and phase, which we exploit to process motion.

The steerable pyramid has non-oriented, real-valued high and low-pass coefficients describing residual signal components not captured by the bandpass filters [48]. The frequency domain transfer functions in the oriented bands of the steerable pyramid, $\Psi_{\omega,\theta}$, are scaled and rotated copies of a basic filter, indexed by scale ω and orientation θ.

The steerable pyramid is built by applying these transfer functions to the discrete Fourier transform $\tilde{I}$ of an image I to decompose it into different spatial frequency bands $S_{\omega,\theta}$ which have DFT $\tilde{S}_{\omega,\theta}(x,y) = \tilde{I}\Psi_{\omega,\theta}$. Each filter isolates a continuous region of the frequency domain and therefore has an impulse response that is localized in space (Figure 3.18(Impulse Response)). The resulting spatial frequency band is localized in space, scale and orientation (see [37] for filter design steps). The transfer functions of a complex steerable pyramid only contain the positive frequencies of the corresponding real steerable pyramid's filter. That is, the response of $2\cos(\omega x) = e^{i\omega x} + e^{-i\omega x}$ is $e^{i\omega x}$ so that there is a notion of both amplitude and phase.

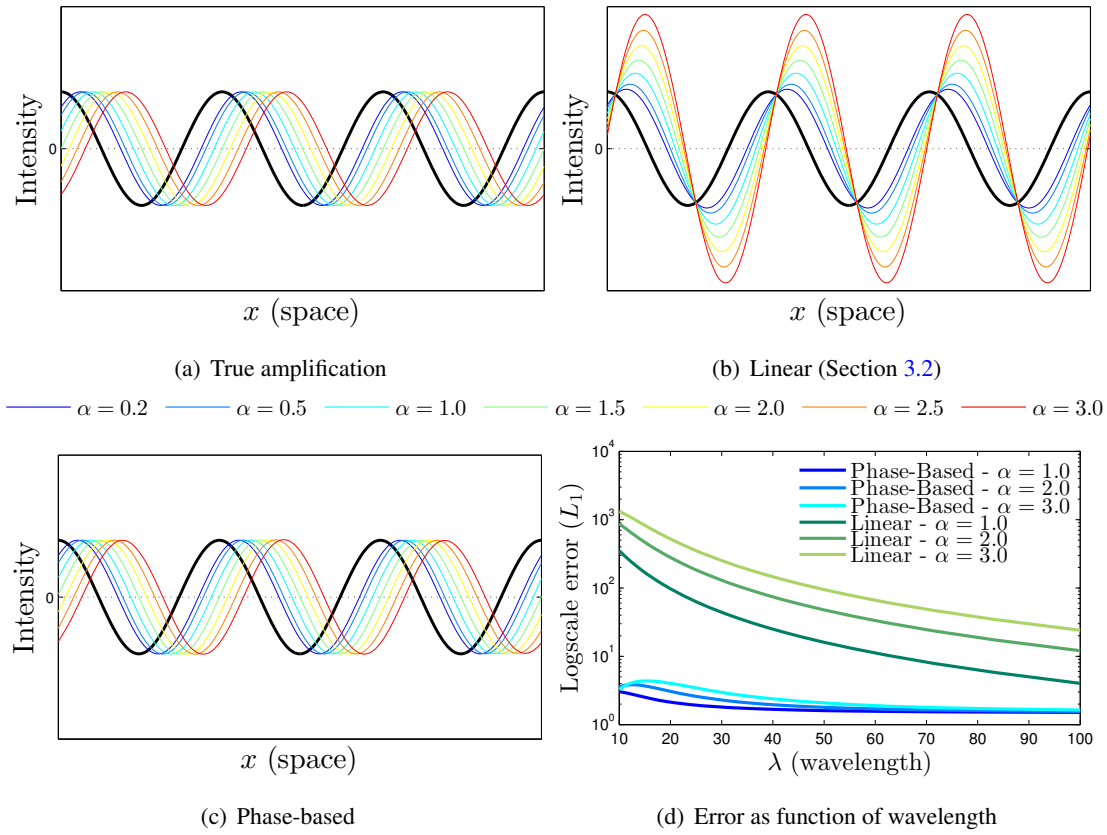
In the frequency domain, the process to build and then collapse the pyramid is given by

$$\tilde{I}_R = \sum \tilde{S}_{\omega,\theta}\Psi_{\omega,\theta} = \sum \tilde{I}\Psi^2_{\omega,\theta} \tag{3.17}$$

where the sums are over all of the scales and orientations in the pyramid, yielding the reconstructed image, I_R. We perform filtering in the frequency domain.

■ 3.3.2 Phase-based Motion Processing

Our processing amplifies small motions by modifying local phase variations in a complex steerable pyramid representation of the video. In this section, we describe our approach and discuss why the phase-based technique has better noise handling and maximum magnification than the



(a) True amplification

(b) Linear (Section 3.2)

(c) Phase-based

(d) Error as function of wavelength

Figure 3.17: Phase-based motion magnification is perfect for sinusoidal functions. In these plots, the initial displacement is $\delta(t) = 1$. While the errors for the linear technique (Section 3.2) are dependent on wavelength for sinusoids, there is no such dependence for the present technique and the error is uniformly small. The vertical axis in (d) is logarithmic.

linear Eulerian motion magnification technique (Section 3.2). To give intuition and to demonstrate that the phase variations correspond to motion, we show how our technique works on sinusoidal waves (Fourier basis elements). For non-periodic image structures, phase-based motion magnification is bounded by the spatial support of the complex steerable pyramid filters. We overcome this bound by using sub-octave bandwidth complex steerable pyramids that have wider spatial support.

■ 3.3.3 Motion Magnification

The phase-based approach relies on complex-valued steerable pyramids because they allow us to measure and modify local motions. To give intuition for our phase-based motion processing, we first give an example using a global Fourier basis and consider the case of a 1D image intensity profile f under global translation over time, $f(x + \delta(t))$, for some displacement function $\delta(t)$ (not to be confused with a Dirac function). We wish to synthesize a sequence with modified motion, $f(x + (1 + \alpha)\delta(t))$, for some magnification factor α. We will discuss the general case at the end of this section.

Using the Fourier series decomposition, we can write the displaced image profile, $f(x + \delta(t))$, as a sum of complex sinusoids,

$$f(x + \delta(t)) = \sum_{\omega=-\infty}^{\infty} A_\omega e^{i\omega(x+\delta(t))} \tag{3.18}$$

in which each band corresponds to a single frequency ω.

From Equation 3.18, the band for frequency ω is the complex sinusoid

$$S_\omega(x, t) = A_\omega e^{i\omega(x+\delta(t))}. \tag{3.19}$$

Because S_ω is a sinusoid, its phase $\omega(x + \delta(t))$ contains motion information. Like the Fourier shift theorem, we can manipulate the motion by modifying the phase.

To isolate motion in specific temporal frequencies, we temporally filter the phase $\omega(x + \delta(t))$ (Equation 3.19) with a DC balanced filter. To simplify the derivation, we assume that the temporal filter has no other effect except to remove the DC component ωx. The result is

$$B_\omega(x, t) = \omega\delta(t). \tag{3.20}$$

We then multiply the bandpassed phase $B_\omega(x, t)$ by α and increase the phase of sub-band $S_\omega(x, t)$ by this amount to get the motion magnified sub-band

$$\hat{S}_\omega(x, t) := S_\omega(x, t)e^{i\alpha B_\omega} = A_\omega e^{i\omega(x+(1+\alpha)\delta(t))}. \tag{3.21}$$

The result $\hat{S}_\omega(x, y)$ is a complex sinusoid that has motions exactly $1 + \alpha$ times the input (Figure 3.17). We can reconstruct the motion-magnified video by collapsing the pyramid. In this analysis, we would do this by summing all the sub-bands to get the motion magnified sequence $f(x + (1 + \alpha)\delta(t))$.

Figure 3.18: A comparison between octave and sub-octave bandwidth pyramids for motion magnification. Each color in the idealized frequency response represents a different filter. (a) The original steerable pyramid of Portilla and Simoncelli [37]. This pyramid has octave bandwidth filters and four orientations. The impulse response of the filters is narrow (rows $2-3$), which reduces the maximum magnification possible (rows $4-5$). (b-c) Pyramid representations with two and four filters per octave, respectively. These representations are more over-complete, but support larger magnification factors.

In general, motions in a video are local and $\delta(t)$ is actually $\delta(x,t)$. We use the complex steerable pyramid to deal with local motions as its filters have impulse responses with finite spatial support (Figure 3.18(Impulse Response)). Specifically, our method works as follows (Figure 3.16). We compute the local phase over time at every spatial scale and orientation of a steerable pyramid. Then, we temporally bandpass these phases to isolate specific temporal frequencies relevant to a given application and remove any temporal DC component. These temporally bandpassed phases correspond to motion in different spatial scales and orientations. To synthesize magnified motion, we multiply the bandpassed phases by an amplification factor α. We then use these amplified phase differences to magnify (or attenuate) the motion in the sequence by modifying the phases of each coefficient by this amount for each frame.

■ 3.3.4 Bounds

As we move an image feature by phase-shifting each complex pyramid filter covering that feature, we eventually reach a limit beyond which we can't move the feature because of the limited spatial support of each pyramid filter (Figure 3.16(a) and Figure 3.18(1D Wavelets)).

As an approximate analytic model of an image feature moved by the localized filters of the steerable pyramid, we consider the case of a single Dirac under uniform translation over time, moved by phase shifting Gabor filters, complex sinusoids modulated by a Gaussian window function. As the Dirac is phase-shifted, it is attenuated by the Gaussian window of the Gabor filters. Therefore, we bound the maximum phase shift such that the Dirac is only attenuated by a small amount.

A one dimensional Gabor filter has frequency domain transfer function

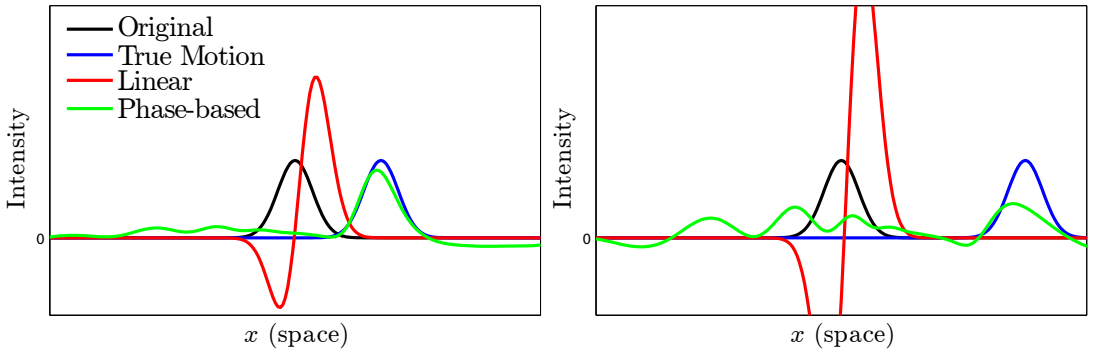
$$e^{-2\pi(\omega_x-\omega_0)^2\sigma^2}, \tag{3.22}$$

where ω_0 is the frequency the filter selects for and $\frac{1}{\sqrt{2\sigma}}$ is the width of Gaussian window in the frequency domain. Typically, σ depends on the frequency ω_0 (self-similar wavelets). The inverse Fourier transform gives us the following impulse response in the spatial domain (up to a constant factor):

$$S_\omega(x,0) = e^{-x^2/(2\sigma^2)}e^{2\pi i\omega_0 x}, \tag{3.23}$$

a complex sinusoid windowed by a Gaussian envelope. Respectively, the impulse response of a Dirac function shifted by $\delta(t)$ pixels (not to be confused with the Dirac function) at time t is

$$S_\omega(x,t) = e^{-(x-\delta(t))^2/(2\sigma^2)}e^{2\pi i\omega_0(x-\delta(t))} \tag{3.24}$$



(a) $\alpha = 6$ (b) $\alpha = 14$

Figure 3.19: For general non-periodic structures, we achieve performance at least four times that of the linear technique, and do not suffer from clipping artifacts (a). For large amplification, the different frequency bands break up due to the higher bands having a smaller window (b).

Note that the spatial Gaussian envelope (the left term on the RHS of Equation 3.24) does not affect the phase.

Applying a finite difference bandpass filter ($[1 - 1]$) to the phase at time 0 and time t, gives

$$B_\omega(x, t) = 2\pi\omega_0\delta(t), \tag{3.25}$$

and the synthesized phase difference for modulating the motion by α is then

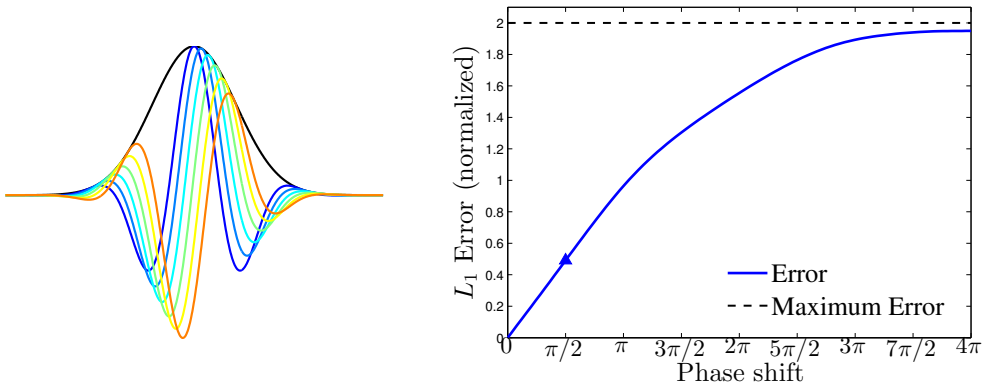
$$2\pi\omega_0\alpha\delta(t). \tag{3.26}$$

This phase difference corresponds to a shift of the Dirac by an additional $\alpha\delta(t)$ pixels. We need to bound the shift $\alpha\delta(t)$ such that the amplified shift approximates well the true shifted signal. We use one standard deviation of the Gaussian window as our bound. This maintains roughly 61% of the amplitude (Figure 3.18 (1D Wavelets), Figure 3.20), and so we have

$$\alpha\delta(t) < \sigma. \tag{3.27}$$

In the octave-bandwidth steerable pyramid of Portilla and Simoncelli [37] (Figure 3.18(a)), there is approximately one period of the sinusoid under the Gaussian envelope. That is, $4\sigma \approx \frac{1}{\omega_0}$, which gives the bound $\alpha\delta(t) < \sigma = \frac{1}{4\omega_0}$. By equating the spatial wavelength $\lambda = \frac{1}{\omega_0}$, we get[1]

$$\alpha\delta(t) < \frac{\lambda}{4}. \tag{3.29}$$



(a) (b)

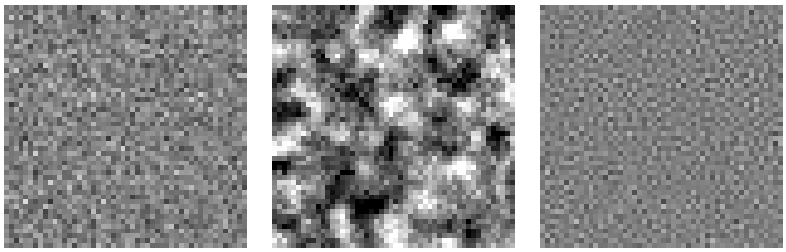
Figure 3.20: The impulse response of the steerable filter bank illustrates the artifacts that arise when modulating phase to magnify motion. (a) The impulse response of a wavelet being phase shifted. As the phase increases (orange corresponds to $\frac{3\pi}{4}$), the primary peak shifts to the right decreasing under the Gaussian window. A secondary peak forms to the left of the primary peak. (c) Error in magnification of the impulse response as the impulse is moved under the Gaussian window. The maximum (normalized) error occurs when the phase-shifted wavelet no longer overlaps with the true-shifted one. The constant $C = \sigma$ is marked on the curve.

From Equation 3.29, we see that the motions of the low spatial frequencies can be magnified more than those of the high spatial frequencies. Indeed, from Equation 3.25, phase changes between frames will be much greater for the high frequency components than for the low frequency components. While derived for an impulse image feature moved by Gabor filters, we find the bound (and its extension below for sub-octave bandwidth pyramids) to be valid for both synthetic examples (Figure 3.19) and natural videos (Figure 3.15, Figure 3.18, Sect. 3.3.7).

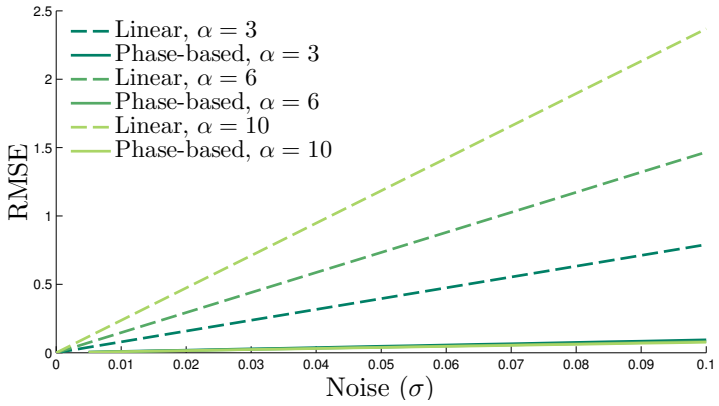
Exceeding the bound in Equation 3.30 manifests as artifacts or blur, as not all image pyramid components are present in their proper ratios to reconstruct the desired translated feature. In Figure 3.19(b), a Gaussian function magnified using our approach breaks up.

■ 3.3.5 Sub-octave Bandwidth Pyramids

We see, therefore, that the bound in Equation 3.29 is directly related to the spatial support of the filters. The smaller the filters in the frequency domain the larger their support is in the spatial domain, which allows us to shift the signals underneath their windows further. In the limit of a having a filter for every frequency band, the representation becomes equivalent to the Fourier



(a) Input (b) Linear (c) Phase-based



(d) Error

Figure 3.21: Comparison between linear and phase-based Eulerian motion magnification in handling noise. (a) A frame in a sequence of IID noise. In both (b) and (c), the motion is amplified by a factor of 50, where (b) uses the linear technique (Section 3.2) and (c) uses the phase-based approach. (d) shows a plot of the error as function of noise for each method, using several magnification factors.

transform and motion magnification is achieved via the shift theorem. However, we then lose the ability to measure or synthesize any spatial variation in the amount of motion. We found a good compromise between localization and magnification ability when using pyramid filters about two times as wide (in the sinusoidally varying spatial direction) as those described in Portilla and Simoncelli [37]. They specify their steerable pyramid filters as being self-similar and having octave bandwidth (Figure 3.18(a)), and we extend their representation to sub-octave bandwidth pyramids (Figure 3.18(b,c)).

A simple way to accomplish this is to scale the filters in log space. This method works well for a half-octave bandwidth pyramid, while pyramids with more filters per octave need to be constructed differently, as discussed in Appendix B.

For the half octave pyramid, there are 2 periods under the Gaussian envelope of the wavelet. Thus, $4\sigma \approx \frac{2}{\omega_0}$, and the bound on the amplification (Equation 3.29) becomes

$$\alpha\delta(t) < \frac{\lambda}{2}. \tag{3.30}$$

This bound improves over the one we derived in Section 3.2 using a Taylor series approximation by a factor of 4.[1]

There is a trade-off between the compactness of the representation and the amount of motion-magnification we can achieve. The 4-orientation, octave-bandwidth pyramid of Portilla and Simoncelli (Figure 3.18(a)) is over-complete by a factor of 12 (each orientation contributes a real and imaginary part), and can easily support real time processing, but limits the amount of motion-magnification that can be applied. On the other hand, an 8-orientation half-octave pyramid (Figure 3.18(b)) supports larger amplification, but is over-complete by a factor of 33.

In Figure 3.22, we show how the overcompleteness varies with the bound on the amplification factor.

■ 3.3.6 Noise handling

Phase-based motion magnification has excellent noise characteristics. As the amplification factor is increased, noise is translated rather than amplified. At a particular scale and orientation band, the response for a noisy image $I + \sigma_n n$ might be

$$S_\omega = e^{i\omega(x+\delta(t))} + \sigma_n N_\omega(x,t), \tag{3.31}$$

where $N_\omega(x,t)$ is the response of n to the complex steerable pyramid filter indexed by ω. We assume that σ_n is much lower in magnitude than the noiseless signal, so that temporal filtering of the phase is approximately $\omega\delta(t)$ as in Equation 3.20. To magnify the motion, the response in the Equation 3.31 is shifted by $e^{i\alpha\omega\delta(t)}$, so that the motion magnified band is

$$\hat{S}_\omega = e^{i\omega(x+(1+\alpha)\delta(t))} + \sigma_n e^{i\alpha\omega\delta(t)} N_\omega(x,t) \tag{3.32}$$

[1]Notice that the bound on the phase-based method is expressed in terms of $\alpha\delta(t)$, while in Section 3.2 it is expressed in terms of $(1+\alpha)\delta(t)$. This is because in this section, we express the motion magnified image profile at time t as generated by modifying (phase-shifting) the shifted, but unamplified image profile at time t, whereas in the analysis in Section 3.2, the motion magnified image profile at time t is generated by modifying the unshifted image profile at time 0.

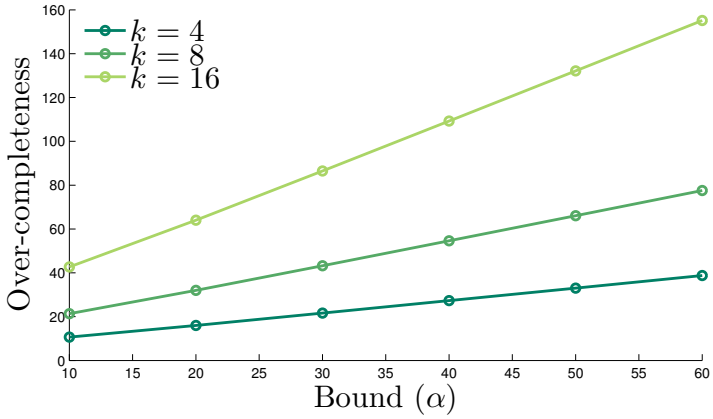


Figure 3.22: Over-completeness as function of the bound on the amplification factor, α, in our pyramid representation with different number of orientations, k, $1-6$ filters per octave (points left to right), and assumed motion $\delta(t) = 0.1$ pixels. For example, a half-octave, 8-orientation pyramid is 32x over-complete, and can amplify motions up to a factor of 20, while a similar quarter-octave pyramid can amplify motions by a factor of 30, and is 43x over-complete.

The only change to the noise after processing is a phase shift. When the pyramid is collapsed, this phase shift corresponds to a translation of the noise. In contrast, the linear magnification method amplifies the noise linearly in α (Figure 3.21).

Still, noise in the input sequence can also cause the phase signal itself to be noisy, which can result in incorrect motions being amplified. We found that we consistently got better results when low-passing the phase signal spatially as a simple way to increase its SNR. However, as the phase-signal in regions of low amplitude is not meaningful, we use an amplitude-weighted spatial Gaussian blur on the phases. For each band i of the representation and each frame k, we have a phase signal $\phi_{i,k}$ and amplitude $A_{i,k}$. We compute a weighted Gaussian blur:

$$\frac{(\phi_{i,k}A_{i,k}) * K_\rho}{A_{i,k} * K_\rho} \tag{3.33}$$

where K_ρ is a Gaussian kernel given by $\exp(-\frac{x^2+y^2}{\rho^2})$. We chose ρ to be equal to that of the spatial domain filter widths. This step incurs a small computational cost that may be avoided for performance considerations, as the results without it are usually good.

3.3.7 Results

Our algorithm allows users to see small motions without excessive noise or computational cost, as well as remove motions that may distract from an underlying phenomena of interest. We show several applications of our algorithm in this section. Please refer to the supplemental video for the full sequences and results.

Unless mentioned otherwise, our processing was done using a complex steerable pyramid with a half-octave bandwidth filters and eight orientations. We computed the filter responses in the frequency domain. The processing was done in YIQ color space and processing was done on each channel independently. Processing a 512×512 video with 300 frames took 56 seconds with an octave-bandwidth pyramid and two orientations, and 280 seconds with the aforementioned half-octave pyramid, using non-optimized MATLAB code on a laptop with 4 cores and 16GB of RAM. With an octave-bandwidth pyramid and 2 orientations, our method can be efficiently implemented to run in real time similar to the linear method, as computing a compact steerable–rather than Laplacian–decomposition introduces a relatively minor performance overhead (about 8x slower, but still within the 30 frames per second range on 512×512 videos using an efficient C++ or GPU implementation). Also similar to the linear method, the user has control over the amplification factor and the temporal bandpass filter.

A Big World of Small Motions The world is full of subtle and small motions that are invisible to the naked eye. Our phase-based approach allows pushing motion magnification further than before, to reveal imperceptible phenomena, not previously visualized, in clarity and detail.

In *eye* (Table C.1), we were able to magnify subtle, involuntary, low amplitude (10-400 micron) movements in the human eye and head such as *microsaccades* [41]. This video was taken with a high speed camera at 500 Hz. A one second (500 frames) sequence was processed with an ideal bandpass filter with passband between $30-50$ Hz and the motions were amplified 150x. A spatial mask was applied to the phase shifts to emphasize the motion around the iris. Such a detection system may have medical applications, as the frequency content of ocular microtremor was shown to have clinical significance [5].

Man-made structures such as buildings and bridges are designed to sway in the wind, but their motion is often invisible. In *crane*, we took a video of a construction crane on a uniform background during a windy day. In the original video, the superstructure does not appear to move, however when amplifying low-frequency motions in the video within $0.2-0.4$ Hz 150x, the swaying of the crane's mast and undulation of its hook become apparent. For this sequence,

a half-octave pyramid yields good results. However, because the crane was a solitary moving object over a uniform background, we found that we were able to further increase the motion and remove artifacts by using a quarter-octave pyramid (Figure 3.18(c)).

Comparison with linear Eulerian motion magnification The main differences between the phase-based approach and the linear approach are summarized in Table 3.2. In particular, the new method supports larger amplification factors and gives a fundamentally better way of handling noise for Eulerian motion magnification. To demonstrate that, we compared the results from this method with those from the linear method described in Section 3.2. Several comparisons are available in Figure 3.15 and the supplemental video. To illustrate that shifting phases is better than directly modifying pixel intensities, we did not spatially-smooth the phase signal in these comparisons.

On *all* the sequences we tested, we found the proposed approach to perform better. In particular, the magnified motions in the phase-based results (e.g. the respiratory motions of the baby and the vibrations of the guitar strings) appear crisper, and contain significantly fewer artifacts and noise.

We also compared the phase-based results with noise removal processing: preceding and following the linear magnification by video denoising. We tested several denoising algorithms, namely NL-means [7], VBM3D [11], and the recent motion-based denoising algorithm by Liu and Freeman [24]. We tuned the denoising methods so as to produce the best result on each sequence. We achieved the overall best performance with VBM3D applied to the motion-magnified video (comparisons with all the denoising methods in pre- and post-processing are available in the supplementary material). We found that in some cases (e.g. *guitar*) denoising

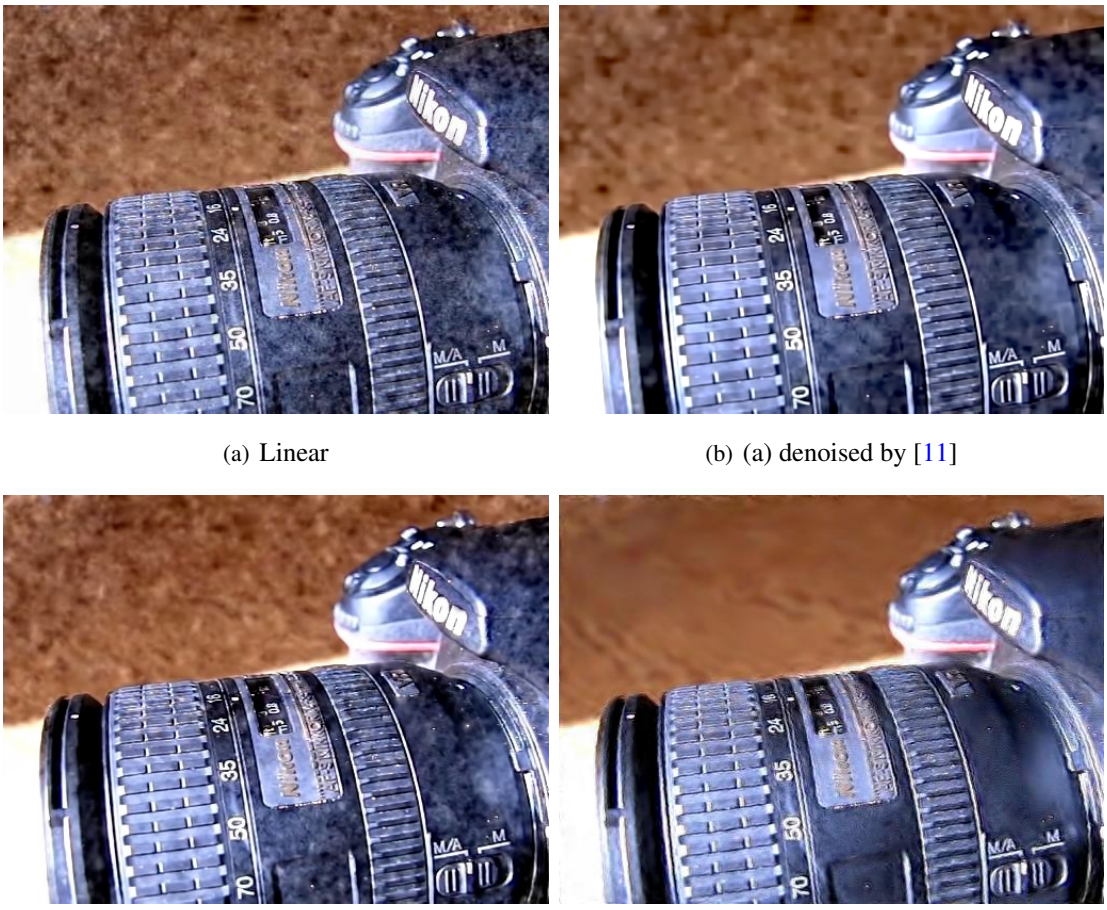
	Linear	**Phase-based**
Decomposition	Laplacian pyramid	Complex steerable pyramid
Over-complete	$4/3$	$2k/(1-2^{-2/n})$
Exact for	Linear ramps	Sinusoids
Bounds	$(1+\alpha)\delta(t) < \lambda/8$	$\alpha\delta(t) < \lambda n/4$
Noise	Magnified	Translated

Table 3.2: The main differences between the linear and phase-based approximations for motion magnification. The representation size is given as a factor of the original frame size, where k represents the number of orientation bands and n represents the number of filters per octave for each orientation.

the video before magnification in fact kills the low-amplitude motion signal we are after. For the low-noise *baby* and *guitar* sequences, the denoised results were visually comparable to that of the phase-based method, although achieved at a higher computational cost, 17 times slower. For the higher-noise *camera* and *eye* sequences, the denoised linear magnification result looks significantly worse than the phase-based results, as the denoising algorithms cannot do much with the medium frequency noise (Figure 3.23).

Controlled Experiments At the miniature scales of motion we are after, one might ask: are the signals we pick out and amplify real (the actual motion signals in the scene)? Would our magnified motions resemble the motions in the scene had they actually been actually larger? To answer these questions, we conducted two controlled experiments. In the first, we recorded ground truth motion data along with a (natural) video (*structure*, Figure 3.24). We induced small motions in a metal structure, and affixed an accelerometer to it to capture its vibrations. To induce the motion we used an impact hammer with a sensor at its tip allowing to record the exact amount of force applied. We then recorded the structure using a standard DSLR video camera at 60 frames per second, along with the accelerometer and impact hammer data. We applied our transform to every frame and recorded the phase changes between the Nth frame and the first frame in one level of the pyramid oriented in the direction of the motion for a salient region of pixels near the accelerometer. These phase changes corresponded to displacement. To recover acceleration, we took a second derivative of Gaussian filter. Once scaled and aligned, the resulting signal matched the data from the accelerometer very closely 3.24(c). We also took two different sequences of the structure, one in which the amplitude of the oscillatory motion was 0.1 pixels and another in which it was 5 pixels (50x larger, from a harder hammer hit). We magnified the former 50 times and found the result to be visually comparable to the latter (Figure 3.24(b)).

In a second experiment, we mount a sheet of rubber on a section of PVC pipe using a rubber band to create a tense membrane (Figure 3.16). We use a loudspeaker to vibrate air in specific frequencies that in turn vibrates the membrane, and capture the result with a high speed camera. Through experimentation, we found two modes of the membrane when waveforms at 76Hz and 110Hz were sent through the loudspeaker. We then took a video of the membrane when a composite waveform of these two frequencies was sent through the loudspeaker and used our algorithm to separate and amplify these two modes. The results of this experiment are in the supplemental material.

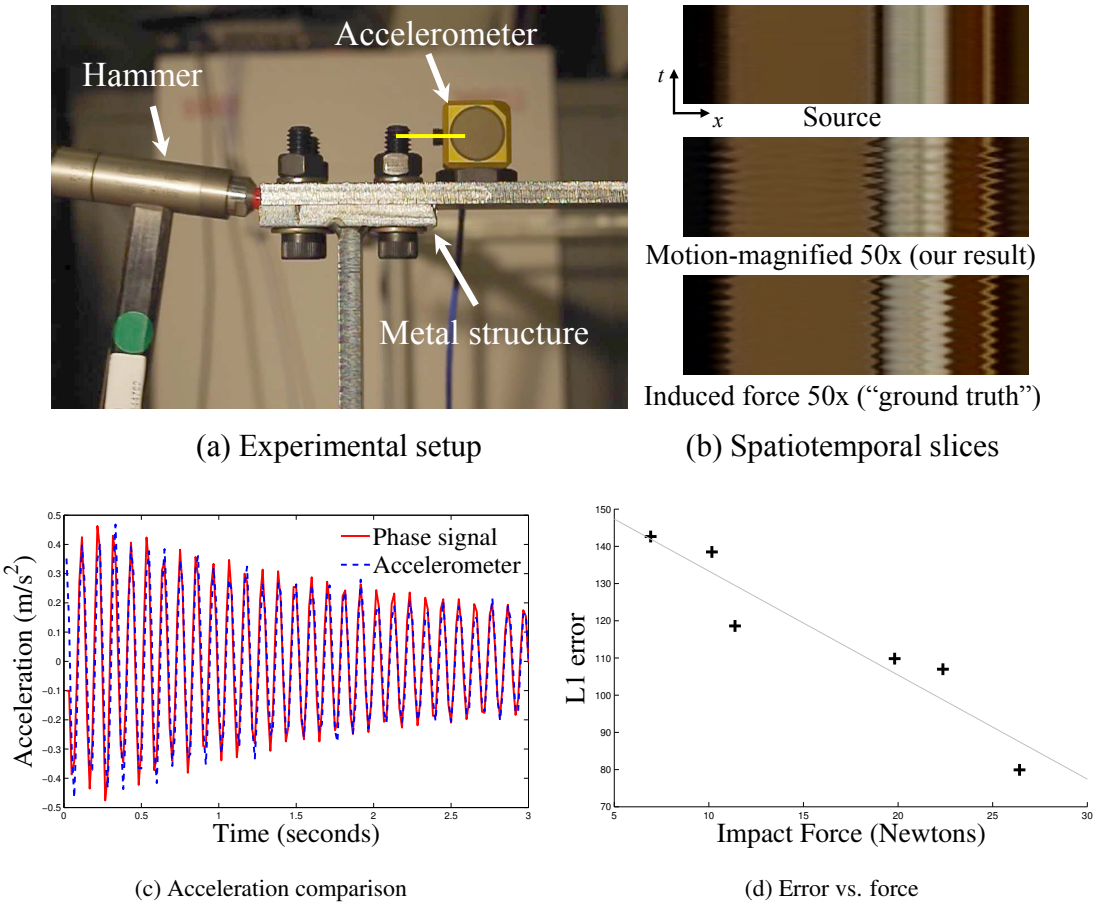


(a) Linear (b) (a) denoised by [11]

(c) (a) denoised by [24] (d) Phase-based

Figure 3.23: Comparison of the phase-based motion magnification result on the *camera* sequence (d) with the result of linear motion magnification (a), denoised by two state-of-the-art video denoising algorithms: VBM3D [11] (b) and motion-based denoising by Liu and Freeman [24] (c). The denoising algorithms cannot deal with the medium frequency noise, and are computationally intensive. The full videos and similar comparisons on other sequences are available in the supplementary material.

Motion Attenuation Our phase-based formulation also lends itself naturally to attenuation of motions in videos, which allows us to remove low-amplitude, short-term motions while larger amplitude motions continue to pass through. Motion attenuation is achieved by setting the amplification factor α to a negative value in the range $[-1, 0)$, where $\alpha = -1$ zeros-out all the phase changes over time within the desired frequency band, effectively canceling out the motions within that band. The result is not the same as a constant frame as the coefficient



(a) Experimental setup

(b) Spatiotemporal slices

(c) Acceleration comparison

(d) Error vs. force

Figure 3.24: A controlled motion magnification experiment to verify our framework. (a) A hammer strikes a metal structures which then moves with a damped oscillatory motion. (b) A sequence with oscillatory motion of amplitude 0.1 pixels is magnified 50 times using our algorithm and compared to a sequence with oscillatory motion of amplitude 5 pixels (50 times the amplitude). (c) A comparison of acceleration extracted from the video with the accelerometer recording. (d) The error in the motion signal we extract from the video, measured as in (c), as function of the impact force. Our motion signal is more accurate as the motions in the scene get larger. All videos are available in the supplementary material.

amplitudes are still evolving over time. This is similar to motion denoising we presented in Chapter 2 [44] and video de-animation [2], but can be done efficiently in our approach (when the motions in the scene are small enough).

We apply motion attenuation for two applications: turbulence removal and color ampli-

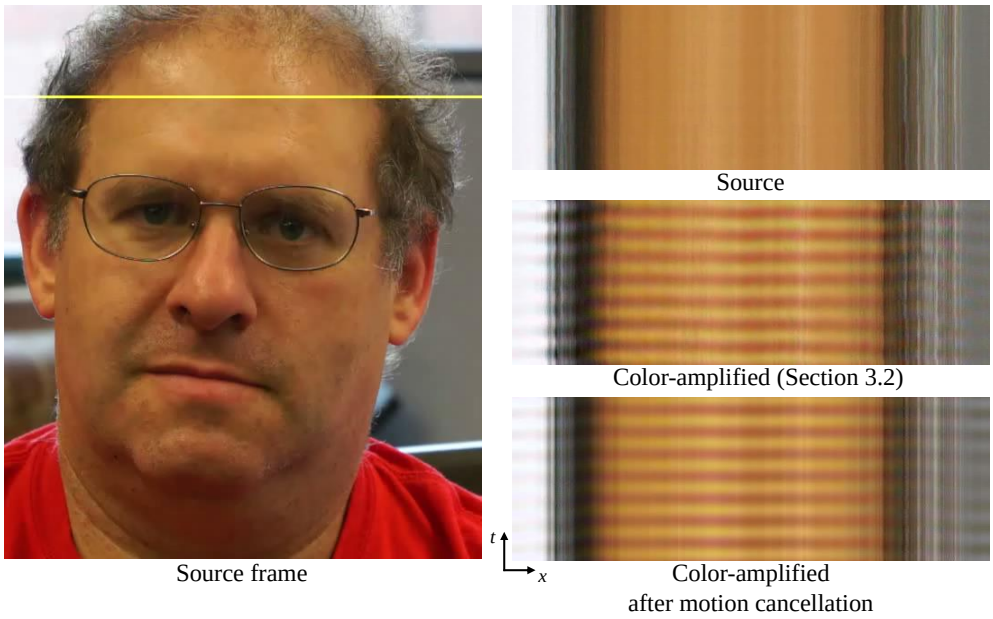


Figure 3.25: Motion attenuation stabilizes unwanted head motions that would otherwise be exaggerated by color amplification. The exaggerated motions appear as wiggles in the middle spatiotemporal slice on the right, and do not appear in the bottom right slice. The full sequence is available in the supplemental video.

fication (Figure 3.25). In *moon* (Table C.1, supplemental video), atmospheric turbulence is manifested as low-mid frequency jitters in a video of the moon as it passes through the night sky (see supplemental video). We pass a temporal window over the video (we used a window of 11 frames), transformed to our representation, and set the phases in each spatial scale and orientation of the center frame to the corresponding median phase of the transformed frames within the temporal window. This effectively *shifts* pixels in order to compensate for the turbulent motions.

Since the linear magnification method amplifies color changes and motions *jointly*, small motions of the face become much larger, visible when amplifying the color changes corresponding to the pulse, which may not be desirable. By canceling the motions as a pre-process to their algorithm, we are able to remove those motions from their results (Figure 3.25).

A similar color amplification result as that of the linear method can be achieved entirely with steerable pyramids. We can temporally bandpass the amplitude A_ω (Equation 3.18) and the low pass residual and add a multiple of the resulting amplitude variations to the amplitude

signal. This yields similar results because in both cases the same processing is applied to the low-pass residual band of an image pyramid (Laplacian pyramid in one case, steerable pyramid in the other).

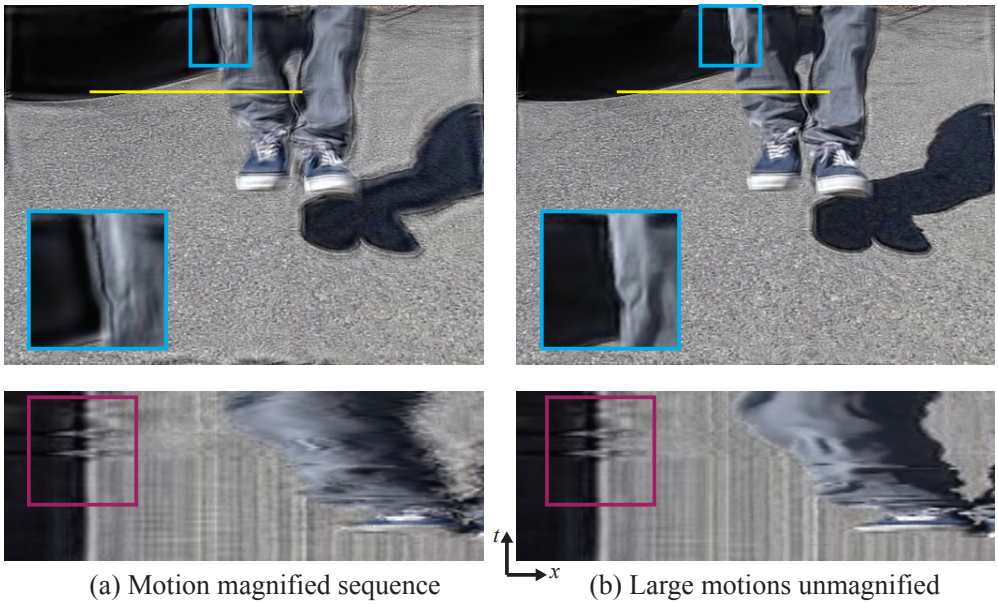
■ 3.3.8 Discussion and Limitations

Lagrangian approaches to motion magnification (e.g. [26]) are complementary to the Eulerian approach proposed in this paper. Such methods can amplify the motion in a video arbitrarily, but rely on accurate optical flow estimates, image segmentation, and inpainting. Such processing is difficult to do well and requires long computation times. In addition, we have shown that for moderate magnification and noisy inputs, the Eulerian approach performs better than Lagrangian (Appendix A). The phase-based method significantly reduces the sensitivity to noise of Eulerian video magnification over the linear method, as well as increases its supported range of amplification, which further expands the regime where it performs better than Lagrangian approaches.

While the analysis in Section 3.2 is exact in the case of linear ramps, the phase-based approach is exact for sinusoidal waves (Figure 3.17), since such signals contain only a single spatial frequency. However, both methods rely on spatial pyramids, where each level is band limited. We argue that such spatially bandpassed images are better approximated by sinusoidal waves than linear ramps.

Our half-octave bandwidth pyramid representation, in which the windowing function of the wavelets in the primal domain is larger, extends the magnification capability of the linear method by a factor of 4, and pyramids with more filters per octave may improve on it by even larger factors. While this allows us to magnify motions further, the wavelets are also more likely to span multiple motions as their support get larger, which may corrupt the phase signal and eventually lead to artifacts in the results. Currently, the user can select the desired representation based on the motions in the scene and the available computational resources.

If the input video has large motions, than the bandpassed phase (Equation 3.20) will not reflect the true motion in the scene and the motion magnified video will suffer from artifacts in the regions of large motion (Figure 3.26(a)). To mitigate this, we can automatically detect regions where phase exceeds our bound (or some user-specified threshold) and set the amplification to be zero in these regions. To increase robustness, we spatiotemporally lowpass the absolute value of the phase and compare the result to a threshold to determine which regions have large motions. The supplememntal video shows an example.



(a) Motion magnified sequence (b) Large motions unmagnified

Figure 3.26: Motion magnification can cause artifacts (cyan insets and spatiotemporal timeslices) in regions of large motion such as those in this sequence of a boy jumping on a platform (a). We can automatically remove such artifacts by identifying regions where the phase change exceeds our bound or a user-specified threshold (b). When the boy hits the platform, the time slice (purple highlights) shows that the subtle motions of the platform or the camera tripod due to the boy's jump are magnified in both cases.

Finally, for sequences in which the phase signal is noisy, parts of the image in the magnified video may appear to move incoherently. Using an image or motion prior to regularize the processing may improve the results in such cases, and is an interesting direction for future work.

■ 3.4 Visualizations and User Interfaces

Sections 3.2 and 3.3 show a variety of synthesized videos where the changes of interest have been amplified. We have also built a prototype application that allows users to reveal subtle changes in real-time from live video feeds, essentially serving as a microscope for temporal variations (Figure 3.27). The user can specify the temporal frequency band of interest by adjusting the high and low cutoff frequencies for the filter. The user can also adjust the amplification factor of the bandpass signal using a slider. Larger amplification results is a larger boost to

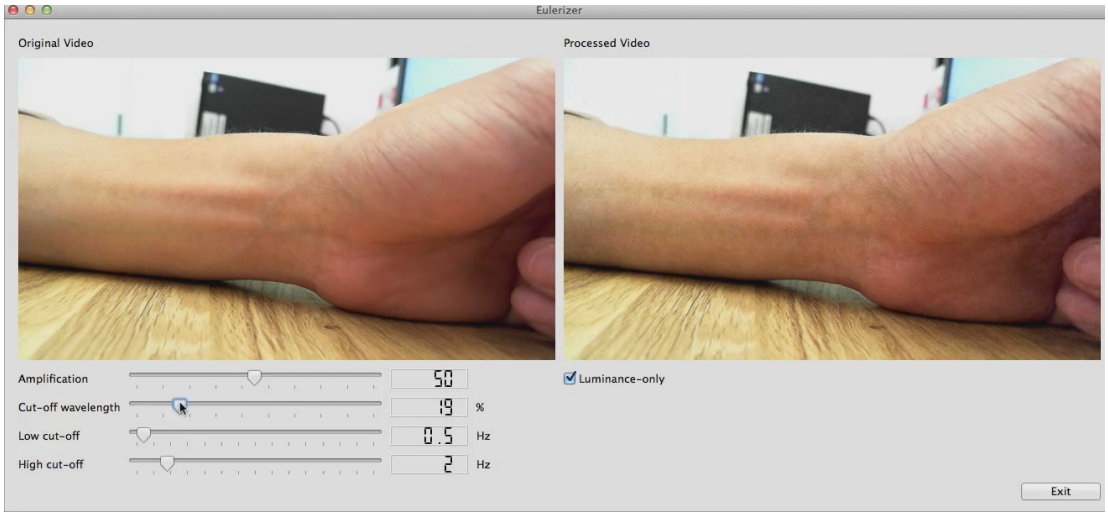


Figure 3.27: A real-time computational "microscope" for small visual changes. This snapshot was taken while using the application to visualize artery pulsation in a wrist. The application was running on a standard laptop using a video feed from an off-the-shelf webcam. A demo is available on the thesis webpage.

the temporal bandpass and makes the variations in the video more apparent. This application is implemented in C++, and is currently using the linear method (Section 3.2) because of its speed. It is entirely CPU-based, and processes 640×480 videos at 60 frames per second on standard laptops and tablets. It can be sped up further by utilizing GPUs. For performance considerations, we amplify only the luminance channel, which gives good enough results. A demo of the application is available in the accompanying video, and we hope to release for public use in the the future.

We have also experimented with a tool that allows the user to browse the variations at different frequencies using pre-configured passbands (Figure 3.28). We dub this process "frequency sweep", as it essentially lets the user sweep through the temporal frequencies to examine the temporal signals in the video. In this interface, the user is presented with a single slider that controls the temporal passband being amplified and displayed. Empirically, we found it useful to set the temporal frequency passband as function of the temporal frequency being shown. Namely, for lower frequencies the passband is smaller, and for higher sequences the passband is larger. In practice, we set the passband size to be the same as the frequency being shown, centered at that frequency. For example, to show temporal variations around 4Hz we use a passband of $2 - 6$Hz; for temporal variations around 32Hz, we use a passband of $16 - 48$Hz.

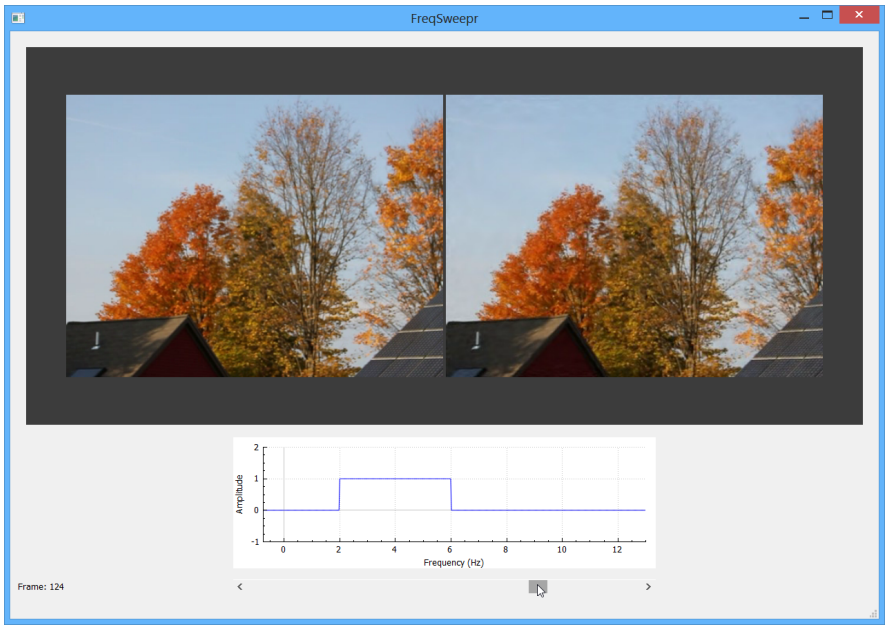


Figure 3.28: An interface that allows the user to *sweep* through the temporal frequency domain and examine temporal phenomena in a simple and intuitive manner. A demo is available in the accompanying video.

This tool can be run in real time with the linear method (Section 3.2) and with some configurations of the phase-based method (Section 3.3, or can be configured to run against pre-computed results using either of the methods.

We have used this tool successfully with several videos. *Trees* and *woman* (Figure 3.28 and the supplemental video) demonstrate ordinary videos that contain changes at different frequencies over time that we cannot normally perceive. We used the interface with *trees*, and found that lower temporal frequencies ($0.5 - 11$Hz) contain the swaying of the tree trunks. At mid-range frequencies ($1 - 2$Hz), we see the motion of the branches, and at high spatial frequencies ($2.5 - 4$Hz), the motion of the leaves is most visible. In *girl*, we can see different motions of the girl's head and shoulders at different frequencies.

The aforementioned tools allow the user to explore temporal signals in videos interactively, however we can also produce static summarizations showing the dominant temporal signals in a video, as shown in Figure 3.29. These visualizations were produced by color-coding each pixel according to the dominant temporal frequency at the pixel. The dominant frequency component is computed as the (temporal) frequency with maximum energy, although other

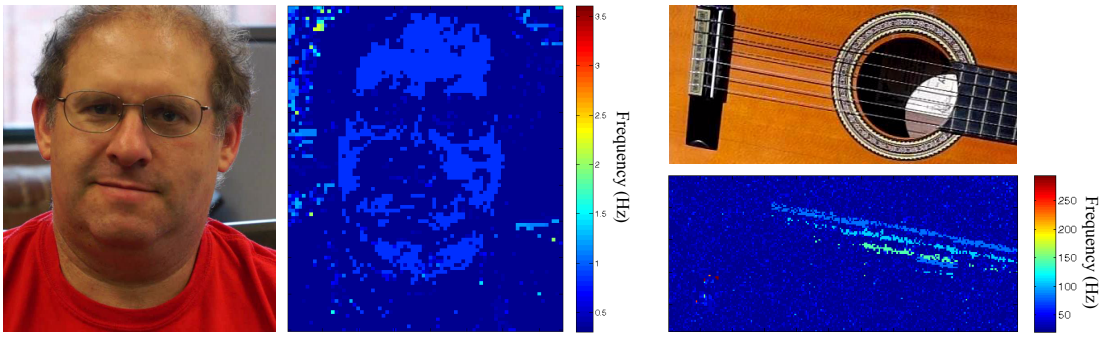


Figure 3.29: A visualization of the dominant temporal frequencies for two sequences: *face* (left; representative frame on the left, visualization on the right) and *guitar* (right; frame at the top, visualization at the bottom), produced by showing, at every pixel, the frequency of maximum energy in the temporal signal recorded at the pixel. This visualization clearly shows the pulsatile areas of the face from which a reliable pulse signal can be extracted, and the frequencies in which the different strings of the guitar vibrate.

methods to determine "dominance" can be used. Such an approach can also be used to detect informative temporal signals in videos automatically. That is an interesting direction for future work.

Chapter 4

Conclusion

We have proposed novel techniques for analyzing and visualizing temporal signals in video, and explored various applications for those techniques involving re-rendering of both larger and smaller temporal variation.

To remove distracting changes, we have introduced motion denoising – the process of decomposing videos into long- and short-term motions, allowing motion resynthesis in a way that maintains one and removes the other. We showed how motion denoising can be cast as an inference problem with a well-defined formulation that does not require explicit motion estimation. This allows the algorithm to operate on videos containing highly involved dynamics. Time-lapse videos fit particularly well within this decomposition model, and we presented a novel application whose goal is, given an input time-lapse sequence, to synthesize a new one that displays only the long-term changes not confounded by the random, short-term changes. We presented results on a set of challenging time-lapse videos. Our technique successfully generates filtered timelapse sequences that are visually faithful to the long-term trends and allow a better grasp of the underlying long-term temporal events in the scene.

To magnify motions and color changes, we described efficient *Eulerian* methods that temporally process pixels in fixed spatial regions, and successfully reveal informative small-amplitude signals in real-world videos. We first described a straightforward approach that merely magnifies temporal color changes using spatiotemporal processing, and showed how it can be used to exaggerate both subtle, purely-temporal color changes, as well as imperceptible spatial motions. This method, while simple and fast, suffers from several limitations, and we then described an improved method that addresses those limitations. In particular, we proposed a method for processing and manipulating small motions in videos by analyzing the local phase over time at different orientations and scales. The local phase is computed using complex steerable pyramids, which we extend to work with sub-octave bandwidth filters in order to increase the spatial

support of the filters and allow us to push motion magnification even further. Our method then magnifies the temporal phase differences in the corresponding bands of these pyramids to hallucinate bigger or smaller motions. We demonstrated that this phase-based technique improves the linear Eulerian motion processing both in theory and in practice, provides a fundamentally better way of handling noise, and produces high quality, photo-realistic videos with amplified or attenuated motions for a variety of applications.

Overall, this thesis makes several important contributions:

- **New models for analyzing images over time.** Different temporal signatures often correspond to different physical processes. Although significant efforts have been devoted to numerous problems related to temporal video analysis (*e.g.*, video stabilization [29, 21], summarization [39]), there is much less work on automatically identifying and characterizing temporal processes in visual data. In this thesis, we have taken several steps in this direction, by proposing novel techniques to decompose the temporal signals in a scene into different components, which can then be modeled or analyzed separately. We believe our approaches will inspire further research, and we expect this area to receive increasing attention in computer vision and graphics research in upcoming years.

- **Eulerian techniques for motion processing.** We have shown that small-amplitude motions can be analyzed and manipulated using Eulerian video processing; that is, without explicitly estimating the motions. This is an important finding since motion analysis and features tracking in video, while well-studied, remains a challenging and computationally-intensive task that is hard to do accurately. In contrast, the Eulerian methods we proposed do not involve any optimization, feature tracking or optical flow computation, but rather process the video separately in space and time. They provide an efficient *representation* of the spatiotemporal signal, on top of which different tasks can be carried out, such as quantitative analysis or visualization. The image decomposition techniques we use, namely Laplacian pyramids [8] and complex steerable pyramids [48], were not invented here, but we utilize them in ways that were not explored before.

- **A big world of small motions.** We demonstrated a plethora of phenomena that exhibit changes and motions that are too small to be perceived by the naked eye, and which can be recovered computationally from regular videos. We explored a variety of medical and

scientific applications that can potentially benefit from our analysis and visualization of those subtle temporal signals.

- **Visualization.** Measuring motions and variations by itself is not enough. A key aspect in the success of our methods is in their ability to efficiently *visualize* the variation. We claim that these visualizations, in many cases, give valuable insights on the phenomena being studied, that may be hard to grasp from quantitative data or point measurement alone.

The techniques we described in this thesis can be used in a number of different domains. For example, our methods to remove short-term variation can help scientists analyze and visualize time-lapse data more effectively, and can also allow photographers to retouch their time-lapse videos. Motion amplification can potentially be used to monitor and visualize vibrations of mechanical systems, or to analyze structural integrity of buildings, bridges, and railroads. Our analysis of subtle color and motion changes can also support contactless vital-sign monitoring for applications in health care, law enforcement, and search and rescue operations.

Appendix A

Eulerian and Lagrangian Motion Magnification

A.1 Derivation of Eulerian and Lagrangian Error

In this section we derive estimates of the error in the Eulerian and Lagrangian motion magnification results with respect to spatial and temporal noise. The derivation is done again for the 1D case for simplicity, and can be generalized to 2D. We use the same setup as in Section 3.2.2, where the true motion-amplified sequence is

$$\begin{aligned}\hat{I}(x,t) &= f(x+(1+\alpha)\delta(t)) \\ &= I(x+(1+\alpha)\delta(t),0).\end{aligned} \tag{A.1}$$

Both methods approximate the true motion-amplified sequence, $\hat{I}(x,t)$ (Equation A.1). Let us first analyze the error in those approximations on the clean signal, $I(x,t)$.

A.1.1 Without Noise

Lagrangian. In the Lagrangian approach, the motion-amplified sequence, $\tilde{I}_L(x,t)$, is achieved by directly amplifying the estimated motion, $\tilde{\delta}(t)$, with respect to the reference frame $I(x,0)$

$$\tilde{I}_L(x,t) = I(x+(1+\alpha)\tilde{\delta}(t),0). \tag{A.2}$$

In its simplest form, we can estimate $\delta(t)$ using point-wise brightness constancy (See Section 3.2.6 for discussion on spatial regularization)

$$\tilde{\delta}(t) = \frac{I_t(x,t)}{I_x(x,t)} \tag{A.3}$$

where $I_x(x,t) = \partial I(x,t)/\partial x$ and $I_t(x,t) = I(x,t) - I(x,0)$. From now on, we will omit the space (x) and time (t) indices when possible for brevity.

The error in in the Lagrangian solution is directly determined by the error in the estimated motion, which we take to be second-order term in the brightness constancy equation (although it is usually not paid in optical flow formulations because of Newton iterations),

$$\begin{aligned} I(x,t) &= I(x+\delta(t),0) \\ &\approx I(x,0) + \delta(t)I_x + \frac{1}{2}\delta^2(t)I_{xx} \\ \Rightarrow \frac{I_t}{I_x} &\approx \delta(t) + \frac{1}{2}\delta^2(t)I_{xx}. \end{aligned} \tag{A.4}$$

The estimated motion, $\tilde{\delta}(t)$, is thus related to the true motion, $\delta(t)$, by

$$\tilde{\delta}(t) \approx \delta(t) + \frac{1}{2}\delta^2(t)I_{xx} \tag{A.5}$$

Plugging (A.5) in (A.2),

$$\begin{aligned} \tilde{I}_L(x,t) &\approx I\left(x + (1+\alpha)\left(\delta(t) + \frac{1}{2}\delta^2(t)I_{xx}\right), 0\right) \\ &\approx I\left(x + (1+\alpha)\delta(t) + \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}, 0\right). \end{aligned} \tag{A.6}$$

Using first-order Taylor expansion of I about $x + (1+\alpha)\delta(t)$, we have

$$\tilde{I}_L(x,t) \approx I(x + (1+\alpha)\delta(t), 0) + \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x. \tag{A.7}$$

Subtracting (A.1) from (A.7), the error in the Lagrangian motion-magnified sequence, ε_L, is

$$\varepsilon_L \approx \left|\frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x\right|. \tag{A.8}$$

Eulerian. In our Eulerian approach, the magnified sequence, $\hat{I}_E(x,t)$, is computed as

$$\begin{aligned} \tilde{I}_E(x,t) &= I(x,t) + \alpha I_t(x,t) \\ &= I(x,0) + (1+\alpha)I_t(x,t) \end{aligned} \tag{A.9}$$

similar to Equation 3.4, using a two-tap temporal filter to compute I_t.

Using a Taylor expansion of the true motion-magnified sequence, $\hat{I}$ (Equation A.1), about x, we have

$$\hat{I}(x,t) \approx I(x,0) + (1+\alpha)\delta(t)I_x + \frac{1}{2}(1+\alpha)^2\delta^2(t)I_{xx} \tag{A.10}$$

Plugging (A.4) into (A.10)

$$\begin{aligned}\hat{I}(x,t) &\approx I(x,0) + (1+\alpha)(I_t - \frac{1}{2}\delta^2(t)I_{xx}I_x) + \frac{1}{2}(1+\alpha)^2\delta^2(t)I_{xx} \\ &\approx I(x,0) + (1+\alpha)I_t - \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x + \frac{1}{2}(1+\alpha)^2\delta^2(t)I_{xx}. \end{aligned} \tag{A.11}$$

Subtracting (A.9) from (A.11) gives the error in the Eulerian motion-magnified solution, ε_E,

$$\varepsilon_E \approx \left| \frac{1}{2}(1+\alpha)^2\delta^2(t)I_{xx} - \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x \right|. \tag{A.12}$$

■ A.1.2 With Noise

Now, let $I'(x,t)$ be the noisy signal, such that

$$I'(x,t) = I(x,t) + n(x,t) \tag{A.13}$$

for additive noise $n(x,t)$.

Lagrangian. The estimated motion becomes

$$\tilde{\delta}(t) = \frac{I'_t}{I'_x} = \frac{I_t + n_t}{I_x + n_x}, \tag{A.14}$$

where $n_x = \partial n/\partial x$ and $n_t = n(x,t) - n(x,0)$.

Using a Taylor Expansion on (n_t, n_x) about $(0,0)$ (zero noise), and using (A.4), we have

$$\begin{aligned}\tilde{\delta}(t) &\approx \frac{I_t}{I_x} + n_t\frac{1}{I_x + n_x} + n_x\frac{I_t + n_t}{(I_x + n_x)^2} \\ &\approx \delta(t) + \frac{n_t}{I_x} - n_x\frac{I_t}{I_x^2} + \frac{1}{2}\delta^2(t)I_{xx}, \end{aligned} \tag{A.15}$$

where terms involving products of the noise components are ignored.

Plugging (A.15) into (A.2), and using a Taylor expansion of I about $x + (1+\alpha)\delta(t)$, we get

$$\tilde{I}'_L(x,t) \approx I(x + (1+\alpha)\delta(t), 0) + (1+\alpha)I_x\left(\frac{n_t}{I_x} - n_x\frac{I_t}{I_x^2} + \frac{1}{2}\delta^2(t)I_{xx}\right) + n. \tag{A.16}$$

Arranging terms, and substituting (A.4) in (A.16),

$$\begin{aligned}\tilde{I}'_L(x,t) &\approx I(x + (1+\alpha)\delta(t), 0) + (1+\alpha)\left(n_t - n_x\left(\delta(t) + \frac{1}{2}\delta^2(t)I_{xx}\right) + \frac{1}{2}\delta^2(t)I_{xx}I_x\right) + n \\ &= I(x + (1+\alpha)\delta(t), 0) + (1+\alpha)n_t - (1+\alpha)n_x\delta(t) - \frac{1}{2}(1+\alpha)n_x\delta^2(t)I_{xx} \\ &\quad + \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x + n \end{aligned} \tag{A.17}$$

Using (A.5) again and subtracting (A.1), the Lagrangian error as function of noise, $\varepsilon_L(n)$, is

$$\varepsilon_L(n) \approx \left|(1+\alpha)n_t - (1+\alpha)n_x\delta(t) - \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}n_x + \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x + n\right| \tag{A.18}$$

Eulerian. In this case, the noisy motion-magnified sequence becomes

$$\begin{aligned}\tilde{I}'_E(x,t) &= I'(x,0) + (1+\alpha)I'_t \\ &= I(x,0) + (1+\alpha)(I_t + n_t) + n \\ &= \tilde{I}_E(x,t) + (1+\alpha)n_t + n\end{aligned} \tag{A.19}$$

Using (A.12) and subtracting (A.1), the Eulerian error as function of noise, $\varepsilon_E(n)$, is

$$\varepsilon_E(n) \approx \left|(1+\alpha)n_t + \frac{1}{2}(1+\alpha)^2\delta^2(t)I_{xx} - \frac{1}{2}(1+\alpha)\delta^2(t)I_{xx}I_x + n\right| \tag{A.20}$$

Notice that if we set the noise to zero in (A.18) and (A.20), the resulting errors correspond to those derived for the non-noisy signal as shown in (A.8) and (A.12).

Appendix B

Sub-octave Bandwidth Pyramids

B.1 Improved Radial Windowing Function for Sub-octave Bandwidth Pyramids

When generalizing the complex steerable pyramid of Portilla and Simoncelli [37] to sub-octave bandwidth pyramids, we found empirically that their windowing function was well-suited for octave and half-octave pyramids. However, at a larger number of filters per octave (≥ 3 in our experiments) this scheme produces filters which are very sharp in the frequency domain and have noticeable ringing artifacts (shown in the 1D wavelet plot of Fig. 3.18(b)).

They define their filters in terms of independent radial and angular windowing functions. For quarter-octave and larger pyramids, we leave the angular windowing function unchanged and propose a different radial windowing function, given by

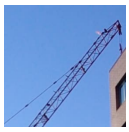
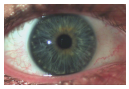
$$\cos^6(\log_2(r))I_{[-\pi/2,\pi/2]}(\log_2(r)). \tag{B.1}$$

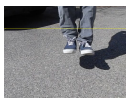
This function has two nice properties: (a) it is smoother, more similar to a Gaussian, and does not introduce ringing in the primal domain, and (b) squared copies scaled by a power of $\frac{\pi}{7}$ sum to a constant factor, so that the transform is invertible and we get perfect reconstruction (Eq. 3.17). An example quarter-octave pyramid generated with this windowing function in shown in Fig. 3.18(c) and its results for motion magnification are available in the supplemental video.

Appendix C

Videos

Table C.1 contains the video sequences used in Chapter 3. All videos and results are available through the thesis web page. When applicable, we list additional properties of the videos. *[this table is under work (MR)]*

Video	Camera	Lens	Resolution	Frame Rate	Bit Depth	Compressor
 *baby*	Canon EOS 60D	EF-S18-200mm f/3.5-5.6 IS	960x544	29.97	24	avc1
 *baby2*			1440x1080	29.97	24	avc1
 *camera*	Casio Exilim EX-F1		512x384	300	24	avc1
 *crane*	Pentax K-x	Sigma 18-200mm f/3.5-6.3 DC	1280x720	24	24	mjpeg
 *engine*	Phantom v10		1776x904	400	24	Uncompressed
 *eye*	Phantom v7.2		1152x896	500	24	Uncompressed

 *girl*	Casio Exilim EX-F1		640x480	60	24	
 *guitar*			432x192	600	24	avc1
 *face*			528x592	30	24	
 *face2*			570x718	30	24	
 *moon*			480x322	21	24	
 *subway*			1280x720	29.97	24	avc1
 *shodow*			960x624	30	24	
 *shuttle*			656x340	23	24	
 *stomp*			512x384	29	24	
 *trees*			640x480	60	24	

 *wrist*	Canon PowerShot S95	6.00–22.5mm	1280x720	24	24	avc1

Table C.1: Videos used in Chapter 3 and their properties. All the videos and results are available through the thesis web page.

Bibliography

[1] Extreme Ice Survey. http://www.extremeicesurvey.org/. 46

[2] Jiamin Bai, Aseem Agarwala, Maneesh Agrawala, and Ravi Ramamoorthi. Selectively de-animating video. *ACM Transactions on Graphics*, 2012. 91

[3] S. Baker, D. Scharstein, JP Lewis, S. Roth, M.J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *ICCV*, pages 1–8, 2007. 14, 40

[4] E.P. Bennett and L. McMillan. Computational time-lapse video. In *ACM SIGGRAPH*, volume 26, 2007. 36, 46

[5] S. Bojanic, T. Simpson, and C. Bolger. Ocular microtremor: a tool for measuring depth of anaesthesia? *British Journal of Anaesthesia*, 86(4):519–522, 2001. 87

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *TPAMI*, 23(11):1222–1239, 2002. 39

[7] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Nonlocal image and movie denoising. *International Journal of Computer Vision*, 76:123–139, 2008. 88

[8] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *IEEE Trans. Comm.*, 31(4):532–540, 1983. 55, 100

[9] Taeg Sang Cho, Moshe Butman, Shai Avidan, and William T. Freeman. The patch transform and its applications to image editing. In *CVPR*, 2008. 37

[10] Ryan A Chylinski. *Time-lapse Photography: A Complete Introduction to Shooting, Processing and Rendering Time-lapse Movies with a DSLR Camera*, volume 1. Cedar Wings Creative, 2012. 36

[11] K. Dabov, A. Foi, and K. Egiazarian. Video denoising by sparse 3d transform-domain collaborative filtering. In *Proc. 15th European Signal Processing Conference*, volume 1, page 7, 2007. 21, 88, 90

[12] Google Earth Engine. http://earthengine.google.org/. 27

[13] P.F. Felzenszwalb and D.P. Huttenlocher. Distance transforms of sampled functions. *Cornell Computing and Information Science Technical Report TR2004-1963*, 2004. 41

[14] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1):41–54, 2006. 42

[15] David J. Fleet and A. D. Jepson. Computation of component image velocity from local phase information. *Int. J. Comput. Vision*, 5(1):77–104, September 1990. 74, 75

[16] William T. Freeman, Edward H. Adelson, and David J. Heeger. Motion without movement. *ACM Comp. Graph.*, 25:27–30, 1991. 61, 74

[17] William T. Freeman and Wojciech Matusik. Analyzing images through time. NSF Award 1111415, September 2011. 27

[18] Martin Fuchs, Tongbo Chen, Oliver Wang, Ramesh Raskar, Hans-Peter Seidel, and Hendrik P.A. Lensch. Real-time temporal shaping of high-speed video streams. *Computers & Graphics*, 34(5):575–584, 2010. 53

[19] T. Gautama and M.A. Van Hulle. A phase-based approach to the estimation of the optical flow field using spatial filtering. *Neural Networks, IEEE Transactions on*, 13(5):1127 – 1136, sep 2002. 74, 75

[20] K Goda, KK Tsia, and B Jalali. Serial time-encoded amplified imaging for real-time observation of fast dynamic phenomena. *Nature*, 458(7242):1145–1149, 2009. 27

[21] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 225–232. IEEE, 2011. 100

[22] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 53, 55

[23] C. Liu, A. Torralba, W.T. Freeman, F. Durand, and E.H. Adelson. Motion magnification. In *ACM SIGGRAPH*, pages 519–526. ACM, 2005. 34

[24] Ce Liu and WilliamT. Freeman. A high-quality video denoising algorithm based on reliable motion estimation. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision ECCV 2010*, volume 6313 of *Lecture Notes in Computer Science*, pages 706–719. Springer Berlin Heidelberg, 2010. 21, 88, 90

[25] Ce Liu, W.T. Freeman, R. Szeliski, and Sing Bing Kang. Noise estimation from a single image. In *IEEE CVPR*, volume 1, pages 901 – 908, 2006. 71

[26] Ce Liu, Antonio Torralba, William T. Freeman, Frédo Durand, and Edward H. Adelson. Motion magnification. *ACM Trans. Graph.*, 24:519–526, 2005. 52, 53, 55, 72, 93

[27] Ce Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, pages 1972–1979, 2009. 41

[28] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3D video stabilization. In *ACM SIGGRAPH*, pages 1–9, 2009. 35

[29] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace video stabilization. *ACM Transactions on Graphics (TOG)*, 30(1):4, 2011. 100

[30] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of IJCAI*, pages 674–679, Apr 1981. 53, 55

[31] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization with motion inpainting. *TPAMI*, 28:1150–1163, 2006. 35

[32] Microsoft. Kinect, Xbox One. `http://www.xbox.com/en-US/xbox-one`, 2013. 62

[33] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001. 46

[34] M.K. Ozkan, M.I. Sezan, and AM Tekalp. Adaptive motion-compensated filtering of noisy image sequences. volume 3, pages 277–290. IEEE, 2002. 34

[35] Philips. Philips Vitals Signs Camera. http://www.vitalsignscamera.com, 2011. 51, 62

[36] Ming-Zher Poh, Daniel J. McDuff, and Rosalind W. Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Opt. Express*, 18(10):10762–10774, 2010. 51, 53, 62, 64

[37] Javier Portilla and Eero P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *Int. J. Comput. Vision*, 40(1):49–70, October 2000. 20, 74, 77, 80, 82, 84, 107

[38] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *ICCV*, pages 151–158, 2009. 37, 39

[39] Y. Pritch, A. Rav-Acha, and S. Peleg. Nonchronological video synopsis and indexing. *TPAMI*, 30(11):1971–1984, 2008. 36, 37, 100

[40] R. Raskar, A. Ilie, and J. Yu. Image fusion for context enhancement and video surrealism. In *ACM SIGGRAPH 2005 Courses*, page 4. 36

[41] Martin Rolfs. Microsaccades: Small steps on a long way. *Vision Research*, 49(20):2415 – 2441, 2009. 87

[42] Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu. Unsupervised joint object discovery and segmentation in internet images. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 7

[43] Michael Rubinstein, Ce Liu, and William T. Freeman. Annotation propagation in large image databases via dense image correspondence. *European Conference on Computer Vision (ECCV)*, pages 85–99, 2012. 7

[44] Michael Rubinstein, Ce Liu, Peter Sand, Fredo Durand, and William T. Freeman. Motion denoising with application to time-lapse photography. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 313–320, June 2011. 28, 30, 91

[45] Michael Rubinstein, Neal Wadhwa, Fredo Durand, and William T. Freeman. Revealing invisible changes in the world. *Science*, 339(6119):519, February 2013. 29

[46] A. Schödl, R. Szeliski, D.H. Salesin, and I. Essa. Video textures. In *ACM SIGGRAPH*, pages 489–498, 2000. 37

[47] E. P. Simoncelli and W. T. Freeman. The steerable pyramid: a flexible architecture for multi-scale derivative computation. In *Proceedings of the 1995 International Conference on Image Processing (Vol. 3)-Volume 3 - Volume 3*, ICIP '95, pages 3444–, Washington, DC, USA, 1995. IEEE Computer Society. 75

[48] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger. Shiftable multi-scale transforms. *IEEE Trans. Info. Theory*, 2(38):587–607, 1992. 74, 75, 77, 100

[49] Kalyan Sunkavalli, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Factored Time-Lapse Video. In *ACM SIGGRAPH*, volume 26, 2007. 36

[50] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *TPAMI*, pages 1068–1080, 2007. 38, 40

[51] M.F. Tappen and W.T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *ICCV*, pages 900–907, 2003. 41

[52] Yaron Ukrainitz and Michal Irani. Aligning sequences and actions by maximizing space-time correlations. In *ECCV*, pages 538–550, 2006. 37

[53] Wim Verkruysse, Lars O. Svaasand, and J. S. Nelson. Remote plethysmographic imaging using ambient light. *Opt. Express*, 16(26):21434–21445, 2008. 51

[54] Neal Wadhwa. *Automatic Detection and Re-Rendering of Motion in Video*. PhD thesis, Massachusetts Institute of Technology, In Progress. 74

[55] Neal Wadhwa, Michael Rubinstein, Fredo Durand, and William T. Freeman. Phase-based video motion processing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 32(3):80:1–80:12, July 2013. 29, 30

[56] Jue Wang, Steven M. Drucker, Maneesh Agrawala, and Michael F. Cohen. The cartoon animation filter. *ACM Trans. Graph.*, 25:1169–1173, 2006. 53, 55

[57] Y. Weiss. Deriving intrinsic images from image sequences. *ICCV*, 2:68–75, 2001. 36

[58] Hao-Yu Wu. Eulerian video processing and medical applications. Master's thesis, Massachusetts Institute of Technology, 2012. 63, 68

[59] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Fredo Durand, and William T. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Transactions on Graphics (Proceedings SIGGRAPH)*, 31(4):65:1–65:8, July 2012. 29, 30, 53, 72

[60] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003. 39