Previous research demonstrated that global phase alone can be used to faithfully represent visual scenes. Here we provide a reconstruction algorithm by using only local phase information. We also demonstrate that local phase alone can be effectively used to detect local motion. The local phase-based motion detector is akin to models employed to detect motion in biological vision, for example, the Reichardt detector. The local phase-based motion detection algorithm introduced here consists of two building blocks. The first building block measures/evaluates the temporal change of the local phase. The temporal derivative of the local phase is shown to exhibit the structure of a second order Volterra kernel with two normalized inputs. We provide an efficient, FFT-based algorithm for implementing the change of the local phase. The second processing building block implements the detector; it compares the maximum of the Radon transform of the local phase derivative with a chosen threshold. We demonstrate examples of applying the local phase-based motion detection algorithm on several video sequences. We also show how the locally detected motion can be used for segmenting moving objects in video scenes and compare our local phase-based algorithm to segmentation achieved with a widely used optic flow algorithm.

1. Introduction

Following Marr, the design of an information processing system can be approached on multiple levels [1]. Figure 1 illustrates two levels of abstraction, namely, the algorithmic level and the physical circuit level. On the algorithmic level, one studies procedurally how the information is processed independently of the physical realization. The circuit level concerns the actual realization of the algorithm in physical hardware, for example, a biological neural circuit or silicon circuits in a digital signal processor. In this paper, we put forth a simple motion detection algorithm that is inspired by motion detection models of biological visual systems (in vivo neural circuit) and provide an efficient realization that can easily be implemented on commodity (in silico) DSP chips.

Visual motion detection is critical to the survival of animals. Many biological visual systems have evolved highly efficient/effective neural circuits to detect visual motion. Motion detection is performed in parallel with other visual coding circuits and starts already in the early stages of visual processing. In the retina of vertebrates, it is known that at least three types of Direction-Selective Ganglion Cells (DSGC) are responsible for signaling visual motion at this early stage [2]. In flies, direction-selective neurons are found in the optic lobe, 3 synapses away from the photoreceptors [3].

The small number of synapses between photoreceptors and direction-selective neurons suggests that the processing involved in motion detection is not highly complex but still very effective. In addition, the biological motion detection circuits are organized in a highly parallel way to enable fast, concurrent computation of motion. It is also interesting to note that the early stages of motion detection are carried out largely in the absence of spiking neurons, indicating that initial stages of motion detection are preferably performed in the “analog” domain. Taking advantage of continuous time processing may be critical for quickly processing motion since motion intrinsically elicits fast and large changes in the intensity levels, that is, large amounts of data under stringent time constraints.

Modern, computer-based motion detection algorithms often employ optic flow techniques to estimate spatial changes in consecutive image frames [4, 5]. Although, often
In polar coordinates, the Fourier transform of $u$ can be expressed as $\mathcal{F}(\mathcal{F}^{-1} u) = \mathcal{F}^{-1}(\mathcal{F}^{-1} u)$, which is equivalent to $\mathcal{F}(u) = \mathcal{F}(u^*)^*$.

The Fourier shift property clearly suggests the relation between the global shift of an image and the global phase shift in the frequency domain. We use this property to compute the change of local phase to indicate motion. The model relies on inhibition to compensate motion in the null direction.

In this paper, we provide an alternative motion detection algorithm based on local phase information of the visual scene. Similar to mechanisms in other biological models, it operates in continuous time and in parallel. Moreover, the motion detection algorithm we propose can be efficiently implemented on parallel hardware. This is, again, similar to the properties of biological motion detection systems. Rather than focusing on velocity of motion, we focus on localization, that is, where the motion occurs in the visual field as well as the direction of motion.

It has been shown that images can be represented by their global phase alone [11]. Here we provide a reconstruction algorithm of visual scenes by only using local phase information, thereby demonstrating the spectrum of the representational capability of phase information.

The Fourier shift property clearly suggests the relationship between the global shift of an image and the global phase shift in the frequency domain. We elevate this relationship by computing the change of local phase to indicate motion that appears locally in the visual scene. The local phases are computed using window functions that tile the visual field with overlapping segments, making it amenable for a highly parallel implementation. In addition, we propose a Radon transforms-based motion detection index on the change of local phases for the readout of the relation between local phases and motion.

Interestingly, phase information has been largely ignored in the field of linear signal processing and for good reason. Phase-based processing is intrinsically non-linear. Recent researches, however, shown that phase information can be smartly employed in speech processing [12] and visual processing [13]. For example, spatial phase in an image is indicative of local features such as edges when considering phase congruency [14]. The role of spatial phase in computational and biological vision, emergence of visual illusions, and pattern recognition is discussed in [15]. Together with our result for motion detection, these studies suggest that phase information has a great potential for achieving efficient visual signal processing.

This paper is organized as follows. In Section 2, we show that local phase information can be used to faithfully represent visual information in the absence of amplitude information. In Section 3, we develop a simple local phase-based algorithm to detect motion in visual scenes and provide an efficient way for its implementation. We then provide examples and applications of the proposed motion detection algorithm to motion segmentation. We also compare our results to those obtained using motion detection algorithms based on optic flow techniques in Section 4. Finally, we summarize our results in Section 5.

2. Representation of Visual Scenes Using Phase Information

The use of complex valued transforms is widespread, for both representing and processing images. When represented in polar coordinates, the output of a complex valued transform of a signal can be split into amplitude and phase. In this section, we define two types of phases of an image: global phase and local phase. We then argue that both types of phases can faithfully represent an image and we provide a reconstruction algorithm that recovers the image from local phase information. This indicates that phase information alone can largely represent an image or video signal.

It will become clear in the sections that follow that the use of phase for representing of image and video information leads to efficient ways to implement certain types of image/video processing algorithms, for example, motion detection algorithms.

2.1. The Global Phase of Images. Recall that the Fourier transform of a real valued image $u = u(x,y), (x,y) \in \mathbb{R}^2$, is given by

$$\mathcal{F}(u) = \int_{\mathbb{R}^2} u(x,y) e^{-j(\omega_x x + \omega_y y)} dx dy$$

with $\mathcal{F}(\omega_x, \omega_y) \in \mathbb{C}$ and $(\omega_x, \omega_y) \in \mathbb{R}^2$.

In polar coordinates, the Fourier transform of $u$ can be expressed as

$$\mathcal{F}(\omega_x, \omega_y) = \mathcal{A}(\omega_x, \omega_y) e^{j\phi(\omega_x, \omega_y)},$$

where $\mathcal{A}$ and $\phi$ represent the amplitude and phase components, respectively.
where \( \hat{A}(\omega_x, \omega_y) \in \mathbb{R} \) is the amplitude and \( \hat{\phi}(\omega_x, \omega_y) \) is the phase of the Fourier transform of \( u \).

**Definition 1.** The amplitude of the Fourier transform of an image \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), is called the global amplitude of \( u \). The phase of the Fourier transform of an image \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), is called the global phase of \( u \).

It is known that the global phase of an image plays an important role in the representation of natural images [11]. A classic example is to take two images and to exchange their global phases before their reconstruction using the inverse Fourier transform. The resulting images are slightly smeared but largely reflect the information contained in the global phase.

2.2. The Local Phase of Images. The global phase indicates the offset of sinusoids of different frequencies contained in the entire image. However, it is not intuitive to relate the global phase to local image features such as edges and their position in an image. To study these local features, it is necessary to modify the Fourier transform such that it reflects properties of a restricted region of an image. It is natural to consider the Short-Time(-Space) Fourier Transform (STFT):

\[
U(\omega_x, \omega_y, x_0, y_0) = \int_{\mathbb{R}^2} u(x, y) \cdot e^{-j(\omega_x x + \omega_y y)} \cdot \delta(x-x_0, y-y_0) \cdot dx \cdot dy, \tag{3}
\]

where \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), is a real valued window function centered at \((x_0, y_0) \in \mathbb{R}^2\). Typical choices of window functions include the Hann window and the Gaussian window. The (effectively) finite support of the window restricts the Fourier transform to local image analysis.

Similarly to the Fourier transform, the STFT can be expressed in polar coordinates as

\[
U(\omega_x, \omega_y, x_0, y_0) = \hat{A}(\omega_x, \omega_y) \cdot e^{j\hat{\phi}(\omega_x, \omega_y, x_0, y_0)}, \tag{4}
\]

where \( \hat{A}(\omega_x, \omega_y, x_0, y_0) \in \mathbb{R} \) is the amplitude and \( \hat{\phi}(\omega_x, \omega_y, x_0, y_0) \) is the phase of the STFT.

**Definition 2.** The amplitude of the STFT of an image \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), is called the local amplitude of \( u \). The phase of the STFT of an image \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), is called the local phase of \( u \).

Note that when \( w \) is a Gaussian window, the STFT of \( u \) evaluated at \((\omega_x, \omega_y, x_0, y_0) \) can be equivalently viewed as the response of a complex-valued Gabor receptive field

\[
h(x, y) = e^{-((x-x_0)^2+(y-y_0)^2)/2\sigma^2} \cdot e^{-j(\omega_x x + \omega_y y)} \tag{5}
\]

to \( u \).

In this case, the window of the STFT clearly is given by

\[
w(x, y) = e^{-((x-x_0)^2+(y-y_0)^2)/2\sigma^2}. \tag{6}
\]

Therefore, the STFT can be realized by an ensemble of Gabor receptive fields that are common in modeling simple and complex cells (neurons) in the primary visual cortex [16].

2.3. Reconstruction of Images from Local Phase. Amplitude and phase can be interpreted as measurements/projections of images that are indicative of their information content. Classically, when both the global amplitude and phase are known, it is straightforward to reconstruct the image. The reconstruction calls for computing the inverse Fourier transform given the global amplitude and phase. Similarly, when using local amplitude and phase, if the sampling functions form a basis or frame in a space of images, the reconstruction is provided by the formalism of wavelet theory, for example, using Gabor wavelets [17].

Amplitude or phase represents partial information extracted from visual scenes. They are obtained via nonlinear sampling, that is, a nonlinear operation for extracting the amplitude and phase information from images. The nonlinear operation makes reconstruction from either amplitude or phase alone difficult. Earlier studies and recent development in solving quadratic constraints, however, suggest that it is possible to reconstruct images from global or local amplitude information [18, 19].

While computing the amplitude requires a second order (quadratic) nonlinearity, computing the phase calls for higher order nonlinear operators (Volterra kernels). It is possible, however, to reconstruct up to a constant scale an image from its global phase information alone without explicitly using the amplitude information. An algorithm was provided for solving this problem in the discrete signal processing domain in [11]. The algorithm smartly avoids using the inverse tangent function by reformulating the phase measurements as a set of linear equations that are easy to solve.

Using a similar argument, we demonstrate in the following that, up to a constant scale, a bandlimited signal \( u = u(x, y), (x, y) \in \mathbb{R}^2 \), can be reconstructed from its local phase alone. We first formulate the encoding of an image \( u \) by local phase, using the Gabor receptive fields as a special case. It is straightforward then to formulate the problem with local phase computed from other types of STFTs.

Formally, we consider an image, \( u = u(x, y), \) on the domain \( \mathbb{R}^2 \), to be an element of a space of trigonometric polynomials \( \mathcal{H} \) of the form

\[
u(x, y) = \sum_{I_x = -L_x}^{L_x} \sum_{I_y = -L_y}^{L_y} c_{I_x} \cdot e_{I_x, I_y}(x, y), \tag{7}
\]

where

\[
e_{I_x, I_y} = \exp \left( j \frac{\Omega_x}{L_x} x \right) \exp \left( j \frac{\Omega_y}{L_y} y \right), \tag{8}
\]

are the set of basis functions of \( \mathcal{H} \), \( \Omega_x, L_x \) are the bandwidth and the order, respectively, of the space in the \( x \) dimension, and \( \Omega_y, L_y \) are the bandwidth and the order, respectively, of the space in the \( y \) dimension.

\( \mathcal{H} \) is a Reproducing Kernel Hilbert Space with inner product [20, 21]

\[
\langle u_1, u_2 \rangle_{[0, T_x] \times [0, T_y]} = \int_{[0, T_x] \times [0, T_y]} u_1(x, y) \cdot u_2(x, y) \cdot dx \cdot dy. \tag{9}
\]
where $T_x = 2\pi L_x/\Omega_x$, $T_y = 2\pi L_y/\Omega_y$ are the period of the space in the $x$ and $y$ dimensions, respectively.

Consider a bank of $N$ Gabor receptive fields

$$h_{klmn}(x, y) = \mathcal{T}_{kl}(w(x, y)e^{-j[\omega_{x_m}x + \omega_{y_n}y]})$$

(10)

where $\mathcal{T}_{kl}$ is the translation operator with $\mathcal{T}_{kl}u = u(x - kl_0, y - lb_0)$, $k, l \in \mathbb{Z}$, $b_0 > 0$, $0 \leq kl_0 \leq T_x$, and $0 \leq lb_0 \leq T_y$, and $\omega_{x_m} = m\omega_0$, $\omega_{y_n} = n\omega_0$, $m, n \in \mathbb{Z}$, $\omega_0 > 0$. $-\Omega_x \leq ma_0 \leq \Omega_x$, and $-\Omega_y \leq na_0 \leq \Omega_y$. The responses of the Gabor receptive fields to the input $u(x, y)$ are given by

$$\int_{\mathbb{R}^2} u(x, y)h_{klmn}(x, y)\, dx\, dy = \int_{\mathbb{R}^2} u(x, y)$$

$$\cdot \mathcal{T}_{kl}(w(x, y)e^{-j[\omega_{x_m}x + \omega_{y_n}y]})\, dx\, dy = A_{klmn}e^{j\phi_{klmn}},$$

where $A_{klmn} \geq 0$, $A_{klmn} \in \mathbb{R}$, is the local amplitude and $\phi_{klmn} \in [0, 2\pi)$ is the local phase.

Dividing both sides of (11) by $e^{j\phi_{klmn}}$, we have

$$\int_{\mathbb{R}^2} u(x, y)w(x - kl_0, y - lb_0)$$

$$\cdot e^{-j[\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn}]}\, dx\, dy = A_{klmn}.$$  (12)

Since $A_{klmn} \in \mathbb{R}$, we have

$$\int_{\mathbb{R}^2} u(x, y)w(x - kl_0, y - lb_0)$$

$$\cdot \cos(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn})\, dx\, dy$$

$$= A_{klmn},$$

(13)

$$\int_{\mathbb{R}^2} u(x, y)w(x - kl_0, y - lb_0)$$

$$\cdot \sin(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn})\, dx\, dy$$

$$= 0.$$  (14)

Note that $w(x - kl_0, y - lb_0)\sin(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn})$ is a real-valued Gabor receptive field with a preferred phase at $\phi_{klmn} + \pi/2 - \omega_{x_m}kl_0 - \omega_{y_n}lb_0$.

**Remark 3.** Assuming that the local phase information $\phi_{klmn}$ is obtained via measurements, that is, filtering the image $u$ with pairs of Gabor receptive fields (10), the set of linear equations (14) has a simple interpretation: the image $u$ is orthogonal to the space spanned by the functions

$$w(x - kl_0, y - lb_0)$$

$$\cdot \sin(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn}),$$

(15)

where $(k, l, m, n) \in \mathfrak{F}$ with $\mathfrak{F} = \{(k, l, m, n) \in \mathbb{Z}^4 \mid 0 \leq kl_0 \leq T_x, 0 \leq lb_0 \leq T_y, -\Omega_x \leq ma_0 \leq \Omega_x, -\Omega_y \leq na_0 \leq \Omega_y\}$.

We are now in the position to provide a reconstruction algorithm of the image from phase $\phi_{klmn}$, $(k, l, m, n) \in \mathfrak{F}$.

**Lemma 4.** $u$ can be reconstructed from $\phi_{klmn}$, $(k, l, m, n) \in \mathfrak{F}$ as

$$u(x, y) = \sum_{l_x = -L_x}^{L_x} \sum_{l_y = -L_y}^{L_y} q_{k,l}(x, y),$$

(16)

with

$$\Phi_c = 0,$$  (17)

where $\Phi$ is a matrix whose $p$th row and $q$th column entry are

$$[\Phi]_{pq} = \int_{\mathbb{R}^2} w(x - kl_0, y - lb_0)$$

$$\cdot \sin(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn})\, dx\, dy.$$  (18)

Here $p$ traverses the set $\mathfrak{F}$, and $q = (2L_y + 1)(L_x + 1) + (L_y + L_y + 1)$. $c$ is a vector of the form

$$c = \begin{bmatrix} c_{L_x, -L_y}, c_{L_x, -L_y + 1}, \ldots, c_{L_x, L_y - 1}, c_{L_x, L_y}, c_{L_x, L_y + 1}, \ldots, c_{-L_x, -L_y}, c_{-L_x, L_y - 1}, \ldots \end{bmatrix}^T$$

(19)

that belongs to the null space of $\Phi$. A necessary condition for perfect reconstruction of $u$, up to a constant scale, is that $N \geq (2L_x + 1)(2L_y + 1) - 1$, where $N$ is the number of phase measurements.

**Proof.** Substituting (7) into (14), we obtain

$$\sum_{l_x = -L_x}^{L_x} \sum_{l_y = -L_y}^{L_y} q_{k,l}(x, y)\int_{\mathbb{R}^2} w(x - kl_0, y - lb_0)$$

$$\cdot \sin(\omega_{x_m}(x - kl_0) + \omega_{y_n}(y - lb_0) + j\phi_{klmn})\, dx\, dy = 0.$$  (20)

for all $k, l, m, n \in \mathbb{Z}$. Therefore, we have

$$\Phi_c = 0,$$  (21)

inferring that $c$ is in the null space of $\Phi$.

If $N < (2L_x + 1)(2L_y + 1) - 1$, it follows from the rank-nullity theorem that dim(null($\Phi$)) $> 1$ leading to multiple linearly independent solutions to $\Phi_c = 0$. □

**Example 5.** In Figure 2 an example of reconstruction of an image is shown using only local phase information. The reconstructed signal was scaled to match the original signal. The SNR of the reconstruction is 44.48 [dB]. An alternative way to obtain a unique reconstruction is to include an additional measurement, for example, the mean value of the signal $\int_{\mathbb{R}^2} u(x, y)\, dx\, dy$ to the system of linear equations (14).
3. Visual Motion Detection from Phase Information

In this section we consider visual fields that change as a function of time. For notational simplicity \( u \) will denote here the space-time intensity of the visual field.

3.1. The Global Phase Equation for Translational Motion. Let \( u = u(x, y, t), (x, y) \in \mathbb{R}^2, t \in \mathbb{R} \), be a visual stimulus. If the visual stimulus is a pure translation of the signal at \( u(x, y, 0) \), that is,

\[
u(x, y, t) = u\left(x - s_x(t), y - s_y(t), 0\right), \tag{22}\]

where

\[
s_x(t) = \int_0^t v_x(s) \, ds, \tag{23}\]
\[
s_y(t) = \int_0^t v_y(s) \, ds\]

are the total length of translation at time \( t \) in each dimension and \( v_x(t) \) and \( v_y(t) \) are the corresponding instantaneous velocity components, then the only difference between \( u(x, y, t) \) and \( u(x, y, 0) \) in the Fourier domain is captured by their global phase. More formally, consider the following.

Lemma 6. The change (derivative) of the global phase is given by

\[
\frac{d\hat{\phi}(\omega_x, \omega_y, t)}{dt} = -\omega_x v_x(t) - \omega_y v_y(t) \tag{24}
\]

where, by abuse of notation, \( \hat{\phi}(\omega_x, \omega_y, t) \) denotes the global phase of \( u(x, y, t) \) and \( \hat{\phi}(\omega_x, \omega_y, 0) \) is the initial condition.

Proof. If \( (\mathcal{F}u(\cdot, 0))(\omega_x, \omega_y) \) is the 2D (spatial) Fourier transform of \( u = u(x, y, 0), (x, y) \in \mathbb{R}^2 \), by the Fourier shift theorem, we have

\[
(\mathcal{F}u(\cdot, t))(\omega_x, \omega_y) = (\mathcal{F}u(\cdot, 0))(\omega_x, \omega_y) e^{-j(\omega_x s_x(t) + \omega_y s_y(t))}. \tag{25}
\]

For a certain frequency component \( (\omega_x, \omega_y) \), the change of its global phase over time amounts to

\[
\frac{d\hat{\phi}(\omega_x, \omega_y, t)}{dt} = -\omega_x \frac{ds_x(t)}{dt} - \omega_y \frac{ds_y(t)}{dt} = -\omega_x v_x(t) - \omega_y v_y(t) \tag{26}
\]

Therefore, in the simple case where the entire visual field is shifting, the derivative of the phase of Fourier components indicates motion, and it can be obtained by the inner product between the component frequency and the velocity vector.

3.2. The Change of Local Phase

3.2.1. The Local Phase Equation for Translational Motion. The analysis in Section 3.1 applies to global motion. This type of motion occurs most frequently when the imaging device, either an eye or a camera, moves. Visual motion in the natural environment, however, is more diverse across the screen since it is, often time, produced by multiple moving objects. The objects can be small and the motion more localized in the visual field.

Taking the global phase of \( u \) will not simply reveal where motion of independent objects takes place or their direction/velocity of motion. The ease of interpretation of motion by using the Fourier transform, however, motivates...
us to reuse the same concept in detecting local motion. This can be achieved by restricting the domain of the visual field where the Fourier transform is applied.

To be able to detect local motion, we consider the local phase of \( u(x, y, t) \) by taking the STFT with window function \( w(x, y) \). Note that, the STFT and its ubiquitous implementation in DSP chips can be extensively used in any dimension. For simplicity and without loss of generality, we consider the window to be centered at \((0, 0)\). The STFT is given by

\[
\mathcal{F}_k w(x, y) = \int_{\mathbb{R}^2} u(x, y, t) w(x, y) e^{-j(\omega_x x + \omega_y y)} dxdy,
\]

where, by abuse of notation, \( A_{00}(\omega_x, \omega_y, t) \) is the amplitude and \( \phi_{00}(\omega_x, \omega_y, t) \) the local phase.

Before we move on to the mathematical analysis, we can intuitively explain the relation between the change in local phase and visual motion taking place across the window support. First, if the stimulus undergoes a uniform change of intensity or it changes proportionally over time due to lighting conditions, for example, the local phase does not change since the phase is invariant with respect to intensity scaling. Therefore, the local phase does not change for such nonmotion stimuli. Second, a rigid edge moving across the window support will induce a phase change.

For a strictly translational signal within the window support (footprint), for example,

\[
u(x, y, t) = u(x - s_x(t), y - s_y(t), 0),
\]

where \( s_x(t) \) and \( s_y(t) \) are as defined in (23), we have the following result.

**Lemma 7.** Consider the following:

\[
\frac{d\phi_{00}}{dt}(\omega_x, \omega_y, t) = -\frac{ds_x(t)}{dt} \omega_x - \frac{ds_y(t)}{dt} \omega_y + \upsilon_{00}(\omega_x, \omega_y, t),
\]

where, by abuse of notation, \( \phi_{00}(\omega_x, \omega_y, t) \) is the local phase of \( u(x, y, t) \) and \( \phi_{00}(\omega_x, \omega_y, 0) \) is the initial condition. The functional form of the term \( \upsilon_{00} = \upsilon_{00}(\omega_x, \omega_y, t) \) is provided in the Appendix.

**Proof.** The derivation of (29) and the functional form of \( \upsilon_{00} \) are given in the Appendix.

**Remark 8.** We notice that the derivative of the local phase has similar structure to that of the global phase, but for the added term \( \upsilon_{00} \). Through simulations, we observed that the first two terms in (29) dominate over the last term for an ON or OFF moving edge [3]. For example, Figure 3 shows the derivative of the local phase given in (29) for an ON edge moving with velocity \((40, 0)\) pixels/sec.

\[
\text{Derivative of the local phase for an ON edge moving in the positive x direction.}
\]
are less robust to noise. Larger windows would enhance object motion detection if the object size is comparable to the window size. However, there would be an increased likelihood of independent movement of multiple objects within the same window, which is not modeled here and thereby may not be robustly detected.

Therefore, for each block \((k, l)\), we obtain \(M^2\) measurements of the phase \(\phi_{kl}(\omega_x, \omega_y, t)\) at every time instant \(t\), with \((\omega_x, \omega_y) \in \mathbb{D}^2\), where

\[
\mathbb{D}^2 = \left\{ (\omega_x = m\omega_0, \omega_y = n\omega_0), \ m, n = -M/2, \\
-M/2 + 1, \ldots, M/2 - 1 \right\},
\]

with \(\omega_0 = 2\pi/M\). We then compute the temporal derivative of the phase, that is, \((d\phi_{kl}/dt)(\omega_x, \omega_y, t)\) for \((\omega_x, \omega_y) \in \mathbb{D}^2\).

We further illustrate an example of the block structure in Figure 4. Figure 4(a) shows an example of an image of \(64 \times 64\) pixels. Four Gaussian windows are shown each with a standard deviation of 4 pixels. The distance between the centers of two neighboring Gaussian windows is 6 pixels. The red solid square shows a \(32 \times 32\)-pixel block with \(k = 3, l = 7\), which encloses effective support of the Gaussian window centered at pixel \((1,1)\). The green dashed square shows another \(32 \times 32\)-pixel block with \(k = 3, l = 7\). The two Gaussian windows on the right are associated with the blocks \(k = 7, l = 3\) and \(k = 7, l = 7\), respectively. Cross section of all Gaussian windows with \(k = 7, l \in [0, 10]\), that is, those centered on the magenta line, are shown in Figure 4(b). The red and blue curve in Figure 4(b) correspond to the two Gaussian windows shown in Figure 4(a). Figure 4(b) also suggests that some of the Gaussian windows are cut off on the boundaries. This is, however, equivalent to assuming that the pixel values outside the boundary are always zero, and it will not significantly affect motion detection based on the change of local phase.

Since the phase, and thereby the phase change, is noisier when the local amplitude is low, an additional denoising step can be employed to discount the measurements of \((d\phi_{kl}/dt)(\omega_x, \omega_y, t)\) for low amplitude values \(A_{kl}(\omega_x, \omega_y, t)\). The denoising is given by

\[
\frac{d\phi_{kl}}{dt}(\omega_x, \omega_y, t) = \frac{A_{kl}(\omega_x, \omega_y, t)}{1/M^2 \sum_{(\omega_x, \omega_y) \in \mathbb{D}^2} A_{kl}(\omega_x, \omega_y, t) + \epsilon},
\]

where \(\epsilon > 0\) is a constant, and \((\omega_x, \omega_y) \in \mathbb{D}^2\).

3.3. The Phase-Based Detector. We propose here a block FFT based algorithm to detect motion using phase information. Such an algorithm is, due to its simplicity and parallelism, highly suitable for an in silico implementation.

3.3.1. Radon Transform on the Change of Phases. We exploit the approximately linear structure of the phase derivative for blocks exhibiting motion by computing the Radon transform of \((d\phi_{kl}/dt)(\omega_x, \omega_y, t)\) over a circular bounded domain \(C = \{(\omega_x, \omega_y) \mid (\omega_x, \omega_y) \in \mathbb{D}^2, \omega_x^2 + \omega_y^2 < \pi^2\}.

Figure 4: Example of block structure. (a) Four yellow disks show 4 Gaussian window functions with translation parameter \(k = 3, l = 3\) (top-left), \(k = 3, l = 7\) (bottom-left), \(k = 7, l = 3\) (top-right), and \(k = 7, l = 7\) (bottom-right). The red solid square shows a \(32 \times 32\)-pixel block approximating the Gaussian window with \(k = 3, l = 3\), and the green dashed square shows a \(32 \times 32\)-pixel block approximating the Gaussian window with \(k = 3, l = 7\). (b) Cross section of all 11 Gaussian windows with translation parameters \(k = 7, l \in [0, 10]\). The cross section is taken as indicated by the magenta line in (a), and the red and blue curve correspond to cross sections of the two Gaussian windows shown in (a) centered on the magenta line.
The Radon transform of the change of phase in the domain $C$ is given by

$$
\left( \mathcal{R} \frac{d\phi_k}{dt} \right)(\rho, \theta, t) = \int_{-\infty}^{\infty} \frac{d\phi_k}{dt}(\rho \cdot \cos\theta - s \cdot \sin\theta, \rho \cdot \sin\theta + s \cdot \cos\theta) \cdot \sin\theta + s \cdot \cos\theta, t) \cdot 1_C(\rho \cdot \cos\theta - s \cdot \sin\theta, \rho \cdot \sin\theta + s \cdot \cos\theta) \, ds,
$$

where

$$
1_C(\omega_x, \omega_y) = \begin{cases} 1 & \text{if } (\omega_x, \omega_y) \in C \\ 0 & \text{otherwise.} \end{cases}
$$

The Radon transform $(\mathcal{R}(d\phi_k/dt))(\rho, \theta, t)$ evaluated at a particular point $(\rho_0, \theta_0, t_0)$ is essentially an integral of $(d\phi_k/dt)(\omega_x, \omega_y, t_0)$ along a line oriented at angle $\pi/2 + \theta_0$ with the $\omega_x$ axis and at distance $|\rho_0|$ along the $(\cos(\theta_0), \sin(\theta_0))$ direction from $(0, 0)$.

If, for a particular $k$ and $l$, we have $(d\phi_k/dt)(\omega_x, \omega_y, t) = -v_x(t)\omega_x - v_y(t)\omega_y$, we have

$$
\left( \mathcal{R} \frac{d\phi_k}{dt} \right)(\rho, \theta, t) = \rho \left[ -v_x(t) \cos\theta - v_y(t) \sin\theta \right],
$$

where

$$
\epsilon(\rho, \theta) = \int_{-\infty}^{\infty} 1_C(\rho \cdot \cos\theta - s \cdot \sin\theta, \rho \cdot \sin\theta + s \cdot \cos\theta) \, ds.
$$

$\epsilon(\rho, \theta)$ is a correction term due to different length of line integrals for different values of $(\rho, \theta)$ in the bounded domain $C$.

After computing the Radon transform of $(d\phi_k/dt)(\omega_x, \omega_y, t)$ for every block $(k, l)$ at time $t_0$, we compute the Phase Motion Indicator (PMI), defined as

$$
\text{PMI}_{kl} = \max_{\theta \in [0, \pi]} \sum_{\rho} \left| \frac{\left( \mathcal{R} \left( \frac{d\phi_k}{dt} \right) \right)(\rho, \theta, t_0)}{\epsilon(\rho, \theta)} \right|.
$$

If the PMI value is comparatively small for these blocks (see also Section 3.3.3), the PMI value is deemed to occur in block $(k, l)$ at time $t_0$. Moreover, the direction of motion, for block $(k, l)$ where motion is detected, can be easily computed as

$$
\widehat{\theta}_{kl} = \pi \left( \frac{\arg\max_{\theta \in [0, \pi]} \left| \sum_{\rho} \left( \mathcal{R} \left( \frac{d\phi_k}{dt} \right) \right)(\rho, \alpha_{kl}, t_0) / \epsilon(\rho, \alpha_{kl}) \right| + 1}{2} \right)
$$

This follows from (36).

3.3.2. The Phase-Based Motion Detection Algorithm. We formally summarize the above analysis as Algorithm 1. Figure 5 shows a schematic diagram of the proposed phase-based motion detection algorithm.

The algorithm is subdivided into two parts. The first part computes local phase changes and the second part is the phase-based motion detector.

In the first part, the screen is divided into overlapping blocks. For example, the red, green, and blue blocks in the plane “divide into overlapping blocks” correspond to the squares of the same color covering the video stream. A Gaussian window is then applied on each block, followed by a 2D FFT operation that is used to extract the local phase. A temporal high-pass filter is then employed to extract phase changes.

In the second part, the PMI is evaluated for each block based on the Radon transform of the local phase changes in each block. Motion is detected for blocks with PMI larger than a preset threshold, and the direction of motion is computed as in (39).

It is easy to notice that the algorithm can be highly parallelized.

3.3.3. Example. We provide an illustrative example in Figure 6 showing how motion is detected using Algorithm 1. The full video of this example can be found in Supplementary Video SI; see Supplementary Material available online at http://dx.doi.org/10.1155/2016/7915245. Figure 6(a) depicts a still from the “highway video” in the Change Detection 2014 dataset [22] evaluated at a particular time $t_0$. As suggested by the algorithm, the screen in Figure 6(a) is divided into $26 \times 19$ overlapping blocks and the window functions are applied to each block. Local phases can then be extracted from the 2D FFT of each windowed block, and the local phase changes are obtained by temporal high-pass filtering. The phase change is shown in Figure 6(b) for all blocks, with block $(12, 11)$ enlarged in Figure 6(c) and block $(23, 6)$ enlarged in Figure 6(d) (see also the plane “2D FFT and extract phase change” in Figure 5). Note that at the time of the video frame, block $(12, 11)$ covers a part of the vehicle in motion.
**Input:** Visual stream or Video data \( u(x, y, t) \)

**Output:** Detected Motion

Construct Gaussian window of size \( M \times M \), denote as \( w \);

for \( t = \Delta T, 2\Delta T, \ldots, N\Delta T \) do

- Divide the screen of frame at time \( t \) into multiple \( M \times M \) blocks \( u_{kl}(t) \), with overlaps;

  foreach block \( u_{kl}(t) \) do

    - Multiply pixel-wise the block \( u_{kl}(t) \) with the Gaussian window \( u_{kl}(t)w \);
    
    - Take \( M \times M \) 2D FFT of \( u_{kl}(t)w \), denote as phase as \( \phi_{kl}(\omega_x, \omega_y, t) \) and amplitude as \( A_{kl}(\omega_x, \omega_y, t) \);
    
    foreach frequency \( \omega_x, \omega_y \) do
    
    - Highpass filter, temporally, \( \phi_{kl}(\omega_x, \omega_y, t) \), denote \( \phi_{kl}(\omega_x, \omega_y, t) \); Optionally, denoise by \( \phi_{kl}(\omega_x, \omega_y, t) \leftarrow \phi_{kl}(\omega_x, \omega_y, t)/(1/M^2) \sum_{(\omega_x,\omega_y) \in D^2} A_{kl}(\omega_x, \omega_y, t) + \epsilon \);
    
    end

    - Compute the radon transform \( \mathcal{R}\phi_{kl}(\rho, \theta, t) \);
    
    - Compute PMI, according to (38);
    
    if \( \text{PMI}_{kl}(t) > \text{threshold} \) then
    
    - Motion Detected at block \( (k, l) \) at time \( t \);
    
    - Compute direction of motion according to (39);
    
    end
  
end

end

**Algorithm 1:** Phase-based motion detection algorithm using the FFT.

**Figure 5:** Schematic diagram of proposed phase-based motion detection algorithm.
in the front, and block (2,3,6) corresponds to an area of the highway pavement where no motion occurs.

Figure 6(f) depicts, for each block (k,l), the maximum phase change over all \((\omega_x,\omega_y) \in \mathbb{D}^2\); that is,

$$\max_{(\omega_x,\omega_y) \in \mathbb{D}^2} \left| \frac{d\phi_{31}}{dt}(\omega_x,\omega_y,t_0) \right|.$$  \hspace{1cm} (41)

We observe from the figure that, for regions with low amplitude, such as the region depicting the road, when the normalization constant is absent, the derivative of the phase can be noisy. For these blocks the maximum of \(|(d\phi_{31}/dt)(\omega_x,\omega_y,t_0)|\) over all \((\omega_x,\omega_y) \in \mathbb{D}^2\) is comparable to the maximum obtained for blocks that cover the vehicles in motion.

However, (29) suggests that the local phase change from multiple filter pairs centered at the same spatial position \((k,l)\) can provide a constraint to robustly estimate motion and its direction. Given the block structure employed in the computation of the local phase, it is natural to utilize phase change information from multiple sources.

Indeed, if, for a particular block \((k,l)\), \((d\phi_{31}/dt)(\omega_x,\omega_y,t) = -v_x(t)\omega_x - v_y(t)\omega_y\), then it is easy to see that \((d\phi_{31}/dt)(\omega_x,\omega_y,t)\) will be zero on the line \(v_1(t)\omega_x + v_2(t)\omega_y = 0\) and have opposite sign on either side of this line. For example, in Figures 6(b) and 6(c), \((d\phi_{31}/dt)(\omega_x,\omega_y,t)\) clearly exhibits this property for blocks that cover a vehicle in motion. The PMI is a tool to evaluate this property.

\[
\frac{d\phi_{31}(\omega_x,\omega_y,t)}{dt} = \frac{(db(\omega_x,\omega_y,t)/dt)a(\omega_x,\omega_y,t) - (da(\omega_x,\omega_y,t)/dt)b(\omega_x,\omega_y,t)}{[a(\omega_x,\omega_y,t)]^2 + [b(\omega_x,\omega_y,t)]^2}, \quad (42)
\]

where \(a(\omega_x,\omega_y,t)\) and \(b(\omega_x,\omega_y,t)\) are, respectively, the real and imaginary parts of \(A_{31}(\omega_x,\omega_y,t)e^{j(\omega_x\omega_x + \omega_y\omega_y)}\).

We notice that the denominator of (42) is the square of the local amplitude of \(u\), and the numerator is of the form of a second order Volterra kernel. This suggests that the time derivative of the local phase can be viewed as a second order Volterra kernel that processes two normalized spatially filtered inputs \(v_1\) and \(v_2\).

We consider an elaborated Reichardt motion detector as shown in Figure 7. It is equipped with a quadrature pair of Gabor filters whose outputs are \(r_1(t) = a(\omega_x,\omega_y,t)\) and \(r_2(t) = b(\omega_x,\omega_y,t)\), respectively, for a particular value of \((\omega_x,\omega_y)\). The pair of Gabor filters that provide these outputs are the real and imaginary parts of \(u(x,y)e^{-j(\omega_xt+\omega_yy)}\). It also consists of a temporal high-pass \(g_1(t)\) and temporal low-pass filter \(g_2(t)\) [9]. The output of the elaborated Reichardt detector follows the diagram in Figure 7 and can be expressed as

\[
(r_2 \ast g_1)(t)(r_1 \ast g_2)(t) - (r_1 \ast g_1)(t)(r_2 \ast g_2)(t). \quad (43)
\]

The response can also be characterized by a second order Volterra kernel. We notice the striking similarity between (43) and the numerator of (42). In fact, the phase-based motion detector shares some properties with the Reichardt motion detector. For example, it is straightforward to see that a single phase-based motion detector is tuned to the temporal frequency of a moving sinusoidal grating.

Since the motion energy detector is formally equivalent to an elaborated Reichardt motion detector [9], the structure of the motion energy detector with divisive normalization is also similar to the phase-based motion detector.

4. Exploratory Results

In this section, we apply the phase-based motion detection algorithm on several video sequences and demonstrate its efficiency and effectiveness in detecting local motion. The motion detection algorithm is compared to two well-known biological motion detectors, namely, the Reichardt motion detector and the Barlow-Levick motion detector. We then show that the detected local motion can be used in motion segmentation tasks. The effectiveness of the segmentation is compared to segmentation using motion information obtained from a widely used optic flow based algorithm available in the literature [4].
Figure 6: Evaluating the motion detection algorithm at time $t_0$ (see also Supplementary Video S1). (a) A still from a $156 \times 112$-pixel video at time $t_0$. (b) $(d\phi_{kl}/dt)(\omega_x, \omega_y, t_0)$ for all $26 \times 19$ blocks, each of size $32 \times 32$. The blocks are concatenated to create a large “phase image” with their relative neighbors kept. (c) $(d\phi_{12,11}/dt)(\omega_x, \omega_y, t_0)$ shown in the $(\omega_x, \omega_y) \in D^2$ space. (d) $(d\phi_{23,6}/dt)(\omega_x, \omega_y, t_0)$ shown in the $(\omega_x, \omega_y) \in D^2$ space. (e) Phase Motion Indicator (PMI) for all blocks at $t_0$. Each pixel $(k, l)$ is color coded for PMI $k_l$. (f) $\max_{(\omega_x, \omega_y) \in D^2}|(d\phi_{kl}/dt)(\omega_x, \omega_y, t_0)|$ for all blocks at $t_0$. Each pixel $(k, l)$ is color coded for this maximum value of block $(k, l)$.

Figure 7: Diagram of an elaborated Reichardt motion detector. It consists of a quadrature pair of Gabor filters whose outputs $r_1$ and $r_2$ provide inputs to the high-pass filters $g_1(t)$ and the low-pass filters $g_2(t)$.

Table 1: Processing Speeds of the proposed algorithm, the Reichardt motion detector, and the Barlow-Levick Motion detector.

<table>
<thead>
<tr>
<th>Screen size</th>
<th>Processing capability (frames per second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposed</td>
</tr>
<tr>
<td>$320 \times 240$</td>
<td>420</td>
</tr>
<tr>
<td>$720 \times 576$</td>
<td>135</td>
</tr>
<tr>
<td>$1280 \times 720$</td>
<td>60</td>
</tr>
<tr>
<td>$1920 \times 1080$</td>
<td>27</td>
</tr>
</tbody>
</table>

4.1. Efficient Parallel Implementation. The algorithm was implemented in PyCUDA [23] and tested on an NVIDIA GeForce GTX TITAN GPU. All computations use single precision floating points. The processing speeds of the algorithm for several screen sizes are listed in Table 1. Clearly, the proposed phase-based motion detection algorithm has real-time capability to process video even with full High Definition screen size.

For comparison, we implemented the Reichardt motion detector and the Barlow-Levick motion detector. Their respective diagrams are shown in Figure 8. Note that we
Figure 8: (a) A Reichardt motion detector detecting motion between the two points \( u_{kl} \) and \( u_{k+1,l} \) in the blurred and downsampled video. (b) A Barlow-Levick motion detector detecting motion between the two points \( u_{kl} \) and \( u_{k+1,l} \) in the blurred and downsampled video. \( g_1(t) \) denotes a high-pass filter and \( g_2(t) \) denotes a low-pass filter.
Figure 9: Top: motion detection of the “highway video” using the phase-based algorithm (a), the Reichardt motion detector (b), and the Barlow-Levick motion detector (c). Red arrows indicate detected motion and its direction. Bottom: the contrast of the video was artificially reduced by 5-fold and the mean was reduced to 3/5 of the original. The red arrows are duplicated from the motion detection result on the original video as in top. Blue arrows are the result of motion detection on the video with reduced contrast. If motion is detected both in the original video and in the video with reduced contrast, then the arrow is shown in magenta (as a mix of blue and red) (see Supplementary Video S2 for full video).

the algorithm only detects motion locally and does not take into account the overall shape of any object.

For comparison, the motion detection results for the Reichardt motion detector and the Barlow-Levick motion detector are shown in the top middle and top right panel of Figure 9 (see Supplementary Video S2 for full video). The Reichardt detector performed relatively well when vehicles are moving faster, but the direction it predicts for vehicles moving slower, for example, on the back of the image is not accurate. In addition, motion is still detected in some parts of the screen where the vehicles have just passed by. The detection result for the Barlow-Levick motion detector was poorer. In particular, the response to OFF edge movement is always the opposite to the actual movement direction.

We then squeezed the range of the screen intensity from $[0, 1]$ to $[0.2, 0.4]$, resulting in a video with a lower mean luminance and lower contrast. The motion detection results on the low-contrast video are shown in the bottom 3 panels of Figure 9 (see Supplementary Video S2 for full video). For reference, the motion detection results for the original video are shown in red arrows. Motion detected in the low-contrast video is shown in blue arrows if no motion is detected for the block in the original video. If motion is detected in both the original and the low-contrast video, the arrows are shown in magenta. Figure 9 clearly shows that while the motion detection performance is degraded for all three detectors, the phase-based motion detector performed still quite well in the low-contrast video, detecting most of the moving vehicles.

The other two detectors missed many of the blocks where motion was detected in the original movie.

To quantify how well each detector works under different contrast conditions, we computed the ratio of unthresholded output values of each motion detector between lower contrast and full contrast video. For example, in the case of phase-based motion detector, we computed, for each block, the ratio between the PMI index for the lower contrast video and that for the full contrast video. The ratios are then averaged across all blocks where motion is detected in the full contrast video. This average for different contrasts is shown in Figure 10 by the blue curve. In the ideal case when the normalization constant $\epsilon$ is 0, the phase detector should produce invariant responses to videos with different contrast. The curve shown here is mainly due to a nonzero $\epsilon$. The ratios for the Reichardt motion detector and the Barlow-Levick motion detector are computed similarly and are shown in red and yellow, respectively. It is clear that the phase-based motion detector has a superior performance across a range of contrast values. At 20% contrast, the phase-based detector still has 50% of the PMI index value for full contrast video. As expected, the response of Barlow-Levick motion detector is linear with respect to contrast, and the Reichardt motion detector has a quadratic relation to contrast. For a fixed threshold value a larger ratio equates to a more consistent performance at lower contrast.

The second video was captured by a surveillance camera in a train station (“train station video”). The video was
under moderate room light with a low noise level. The front side of the video had high contrast; illumination on the back side was quite low. The detected motion is shown in the video of Figure 11 (see Supplementary Video S3 for full video). Movements of people were successfully captured by the motion detection algorithm.

The third video was a “thermal video” with large amount of background noise (thermal video). The threshold for detecting motion was raised by 60% in order to mitigate the increased level of noise. The detected motion is shown in the video of Figure 12 (see Supplementary Video S4 for full video).

The last example we show here was taken from a highway surveillance camera at night (“winterstreet video”). The overall illumination on the lower-left side was low whereas illumination was moderate on the upper-right side where the road was covered by snow. The detected motions are shown in the video in Figure 13 (see Supplementary Video S5 for full video). We note that, overall, car movements were successfully detected. Car movements on the lower-left side, however, suffered from low illumination and some parts of the car were not detected well due to the trade-off employed for noise suppression.

With a higher threshold, the phase-base motion detection algorithm is able to detect motion under noisy conditions. We added to the original “highway video” and “train station video” Gaussian white noise with standard deviation 5% of the maximum luminance range. The results are shown, respectively, in Figures 14 and 15 (see Supplementary Videos S6 and S7 for full videos).

4.3. Examples of Motion Segmentation. We asked whether the detected motion signals in the video sequences can be useful for segmenting moving objects from the background. We applied a larger threshold to only signal motion for salient objects. The 32 × 32 blocks, however, introduce large boundaries around the moving objects. To reduce the boundary and to segment the moving object more closely to the actual object boundary, we applied the motion detection algorithm around the detected boundary with 16 × 16 blocks. If 16 × 16 blocks did not indicate motion, then the corresponding area was removed from the segmented object area.

For comparison, we performed motion segmentation based on local motion detection based on an optic flow algorithm [4]. The segmentation in this case was implemented by comparing the length of the optic flow vectors with an appropriate threshold.

We employed a simple thresholding for both phase-based motion detection algorithms and optic flow based motion detection. More sophisticated algorithms may produce better segmentation results. Thus, the segmentation was purely based on motion cues and no postprocessing at pixel level was performed. The state-of-the-art results for the Change Detection 2014 dataset utilize multiple cues such as motion, color, and background extraction to segment objects and thereby achieve better results. We are exploring here, however, only the case where motion is the only cue for segmentation. Therefore, the ground truth information from the dataset was not applicable to our test. We will, therefore, only show the effectiveness of motion segmentation visually.

We first applied motion based segmentation on 2-second segment of the “highway video.” The result using the local phase-based motion detection algorithm is shown in the video of Figure 16(a) (see Supplementary Video S8a for full video) and that using optic flow based motion detection algorithm is shown in the video of Figure 16(b) (see Supplementary Video S8b for full video). Both videos are played back at 1/4 of speed. With a higher threshold, the movement of the leaves was no longer picked up by the phase-based motion detector. Therefore, only the cars were identified as moving objects and they are indicated in red. Although the moving objects were not perfectly segmented on their boundary, they were mostly captured. For the optic flow based segmentation, since the regions of interest are set by thresholding the length of the velocity vector, objects moving at lower speed, for example, the cars on the top, were not always picked up by the segmentation algorithm.

We then applied the motion segmentation to a 2-second segment of the “train station video,” the “thermal video,” and the “winterstreet video.” The results are shown, respectively, in the videos of Figures 17, 18, and 19 (see Supplementary Videos S9a and S9b, S10a and S10b, and S11a and S11b, resp., for full videos).

These results show that for the purpose of detecting local motion and its use as a motion segmentation cue, the local phase-based motion detector works as good, if not better than a simple thresholding segmentation using an optic flow based algorithm.

5. Discussion

Previous research demonstrated that global phase information alone can be used to faithfully represent visual scenes. Here we provided a reconstruction algorithm of visual scenes.
by only using local phase information. More importantly, local phase information can be effectively used to detect local motion. Through a simple temporal derivative of the phase, we obtained a second order Volterra kernel that is applied on two normalized inputs. The structure of the second order Volterra kernel in the phase-based motion detector is akin to models employed to detect motion in biological vision, for example, the Reichardt detector [7] and the motion energy detector [9].

We then proposed an efficient, FFT-based algorithm employing the change in local phase for detecting motion. In order to exploit the special structure of the change in
Figure 13: Motion detection of the "winterstreet video" using the phase-based algorithm (a), the Reichardt motion detector (b), and the Barlow-Levick motion detector (c) (see Figure 9 for description of each panel) (see Supplementary Video S5 for full video).

Figure 14: Phase-based motion detection applied on the "highway video" with added Gaussian white noise (see Supplementary Video S6 for full video).

Figure 15: Phase-based motion detection applied on the "train station video" with added Gaussian white noise (see Supplementary Video S7 for full video).

phase in the frequency domain that is due to rigid motion, the phase-based motion detection algorithm also incorporates the Radon transform, a transform closely related to the Fourier transform. Based on the Radon transform, a motion indicator was proposed to robustly detect whether the phase change is caused by motion. Therefore, the algorithm can be efficiently implemented whenever the FFT is available-supported. We showed examples of applying the phase-based motion detection algorithm on several video sequences. We also showed that the locally detected motion can be used for segmenting moving objects in video scenes. We compared the segmentation of moving objects using our local phase-based algorithm to segmentation achieved using a widely used optic flow based algorithm. Our results suggest that spatial phase information may provide an efficient alternative to perform many visual tasks in silico as well as in modeling in vivo biological vision systems. This is consistent with other recent findings [15].

Note that phase information has been used for solving various visual tasks in the past. In fact, phase has been successfully employed in optic flow algorithms [24] and image registration for translation [25], both applied to motion related tasks. The phase-based optic flow algorithm applies the optic flow equation on the local phase of images rather than the intensity itself to achieve better resolution and robustness in estimating motion velocity. The phase correlation method computes the normalized cross-power spectrum of two images to extract the phase difference. It provides better accuracy and robustness as compared to the classical cross correlation approach applied to two consecutive images in a sequence. However, it has limitations when dealing with images with a repetitive structure.
Our method differs from the above two cases in the following ways. First, it employs a simple temporal derivative/high-pass filtering on the phase to extract local phase changes. The structure of the phase change is exploited for better detection of motion. On the contrary, the phase correlation method considers the structure of the phase itself. This also allowed us to make the motion detection a more local, continuous process rather than purely operating globally on discrete frames. Second, it explores the structure of the change of phase due to motion in the frequency domain rather than in the spatial domain, in which the key constraints of optic flow equations are based. Third,
instead of focusing on estimating the exact velocity or shift, our method is centered on the detection of motion with a coarse local estimate of direction. This is the case in the first steps of biological motion detection in the retina or optic lobe of insects; we discussed the resemblance of the method presented here to those of biological models of motion detection.

The proposed motion detection algorithm, however, shares several advantages with the other phase-based methods. For example, it is sensitive to motion that only induces a subpixel shift between frames and for very small differences in intensity. In addition, when compared to the amplitude, the local phase is robust under different contrast and illumination conditions. Consequently, the algorithm presented here can operate in a wide range of contrast/illumination conditions.

Furthermore, once the local phase is extracted from each block, motion detection becomes a localized, temporal operation on the local phase of each block. This forms the basis for the highly parallel structure in the phase-based motion detection algorithm. By contrast, traditional optic flow techniques often rely on explicit comparisons across spatial locations, which increase, for example, memory complexity since all states must be made available to their neighbors.

We also notice that, for each block, a large number of measurements of phase changes are obtained. From Figure 6, we see that this number is much higher than that of the original pixel space. In other words, in order to detect motion in a robust way, the local phase-based motion detection algorithm undergoes an expansion of measurements before settling down onto a single motion indicator value. This number of measurements, however, does not incur additional computational demand thanks to the highly efficient FFT algorithm. Biological visual systems often have a similar structure. For example, in the vertebrate retina, computations are carried out by an extraordinarily large number of neurons until measurements of the visual scene are projected by a fraction of the neurons onto the cortex [26]. Similar expansion takes place in the primary visual cortex as well.

We also highlight the ease of implementation of the intrinsically parallel algorithm proposed here. The algorithm introduced in Section 3.3 is based on the FFT algorithm and does not require solving an optimization problem. It can be efficiently implemented in hardware, for example, FPGAs, or in software, for example, using GPU accelerators. We note that extending the FFT to higher dimensions is straightforward and the implementations of higher dimensional FFTs are also highly efficient. Clearly, the methodology can be applied to motion detection of data in 3D or higher dimensional space, where the Radon transform operates over planes or hyperplanes.

Finally, we argue that the change of phase, although highly nonlinear, can be obtained through a normalization (gain control) followed by a second order Volterra kernel. This separation of a higher order nonlinearity into gain control block and a lower order nonlinear filter can be used for modeling motion detection circuits in biological systems.

Appendix

The Derivative of the Local Phase

Let
\[
a(\omega_x, \omega_y, t) = \int_{\mathbb{R}^2} u(x, y, t) w(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy,
\]
\[
b(\omega_x, \omega_y, t) = -\int_{\mathbb{R}^2} u(x, y, t) w(x, y) \sin(\omega_x x + \omega_y y) \, dx \, dy,
\]
and, therefore,
\[
A_{00}(\omega_x, \omega_y, t) e^{j\phi_{00}(\omega_x, \omega_y, t)} = a(\omega_x, \omega_y, t) + j b(\omega_x, \omega_y, t).
\]

The local phase amounts to \( \phi_{00}(\omega_x, \omega_y, t) = \arctan2(b(\omega_x, \omega_y, t), a(\omega_x, \omega_y, t)) \) and therefore
\[
\frac{d\phi_{00}}{dt} = \frac{a \frac{db}{dt} - b \frac{da}{dt}}{|a|^2 + |b|^2}.
\]
Now,
\[
\frac{da}{dt}(\omega_x, \omega_y, t)
= \int_{\mathbb{R}^2} \frac{\partial u}{\partial t}(x, y, t) w(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
\]
\[\text{(A.4)}\]

For a purely translational signal with instantaneous spatial shift given by \((s_x(t), s_y(t))\) within the support of the window we have
\[
\frac{da}{dt}(\omega_x, \omega_y, t)
= -\int_{\mathbb{R}^2} \frac{\partial u}{\partial x}(x, y, t) w(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
- \frac{ds_x}{dt} \int_{\mathbb{R}^2} u(x, y, t) w(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
+ \frac{ds_y}{dt} \int_{\mathbb{R}^2} u(x, y, t) w(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
- \omega_x \frac{ds_x}{dt} \int_{\mathbb{R}^2} u(x, y, t) w(x, y) \sin(\omega_x x + \omega_y y) \, dx \, dy
- \omega_y \frac{ds_y}{dt} \int_{\mathbb{R}^2} u(x, y, t) w(x, y) \sin(\omega_x x + \omega_y y) \, dx \, dy
= \frac{ds_x}{dt} \int_{\mathbb{R}^2} u(x, y, t) \frac{\partial w}{\partial x}(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
+ \frac{ds_y}{dt} \int_{\mathbb{R}^2} u(x, y, t) \frac{\partial w}{\partial y}(x, y) \cos(\omega_x x + \omega_y y) \, dx \, dy
+ \omega_x \frac{ds_x}{dt} b
+ \omega_y \frac{ds_y}{dt} b.
\]
\[\text{(A.5)}\]

Similarly,
\[
\frac{db}{dt}(\omega_x, \omega_y, t)
= -\frac{ds_x}{dt}
- \omega_x \frac{ds_x}{dt} a
- \omega_y \frac{ds_y}{dt} a.
\]
\[\text{(A.6)}\]

Plugging in the value of \(da/dt\) and \(db/dt\), the derivative of the local phase amounts to
\[\text{(A.7)}\]

**Disclosure**

The author’s names are alphabetically listed.

**Conflict of Interests**

The authors declare that there is no conflict of interests regarding the publication of this paper.

**Acknowledgment**

The research reported here was supported by AFOSR under Grant no. FA9550-12-1-0232.

**References**


