Les Cahiers du GERAD ISSN: 0711–2440

Decoupling spatial pattern and its movement via complex factorization over orthogonal filter pairs

Y. Mu, R. Dimitrakopoulos, F. Ferrie

G–2022–62

December 2022

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2022
– Bibliothèque et Archives Canada, 2022

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2022
– Library and Archives Canada, 2022

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Decoupling spatial pattern and its movement via complex factorization over orthogonal filter pairs

Yanyan Mu [a, c, e]

Roussos Dimitrakopoulos [b, c, e]

Frank Ferrie [a, d]

[a] *Centre for Intelligent Machines, McGill University, McConnell Engineering Building, Montréal (Qc), Canada, H3A 2A7*

[b] *COSMO – Stochastic Mine Planning Laboratory, McGill University, Montréal (Qc), Canada, H3A 2A7*

[c] *Department of Mining and Materials Engineering, McGill University, Montréal (Qc), Canada, H3A 2A7*

[d] *Department of Electrical and Computer Engineering, McGill University, Montréal (Qc), Canada, H3A 2A7*

[e] *GERAD, Montréal (Qc), Canada, H3T 1J4*

yanyanmu@cim.mcgill.ca
roussos.dimitrakopoulos@mcgill.ca
ferrie@cim.mcgill.ca

December 2022
Les Cahiers du GERAD
G–2022–62

Abstract : Variations between related images (e.g. due to motions) can caused by different independent factors. A qualified representation can decouple the underlying explanatory factors rather than keeping them mixed. After decoupling, each factor lies in a lower dimension abstract space. Different computer vision tasks can be done in different abstract spaces more efficiently than in the original pixel space. For example, conducting object recognition in appearance space can result in an invariant recognition; estimating object motion in location space yields a result regardless of the object itself. In this paper, we propose an algorithm to decouple object appearance and location to amplitude and phase in static images by using complex factorization over orthogonal filter pairs. In particular, we show that, i) Orthogonal filter pairs can be learned in an unsupervised manner from multiple consecutive frames; ii) Object movement is encoded in the factorized phase gradient between frames over time. As a proof of concept, we present experiments on the application of our framework to the recovery of the optical flow. Here object movement is successfully captured by phase gradient.

Acknowledgements: We gratefully acknowledge to Nvdia that offer us a Tesla K40 for supporting this research.

1 Introduction

For many computer vision tasks, we assume that several independent, generative factors give rise to the particular image data, with each factor comprising a source of appearance variation. Two or more factors changing simultaneously can result in the generated image varying dramatically. Consider a video sequence of a moving object. The object appearance can vary dramatically, e.g. a consequence of the relative position changes between the object and the camera. This appearance change makes invariant recognition for each frame difficult and can also make object tracking difficult as well. We propose an algorithm which decouples object appearance and movement into two independent abstract spaces. The activation in appearance space encodes an invariant representation of the object's appearance; the activation in motion space encodes the object motion no matter the variation on a pixel level. This decoupling problem is a big challenge for neural nets (NN) to solve. In most NN based algorithms, the input images are mapped to a latent space by: $aj = \sum_i I\omega_{ij}$, where I is the input image, aj are the latent coefficients and wij are the filters. As the appearance of the input image varies, the coefficients change accordingly. However, the coefficients themselves do not indicate which underlying factor makes them change. This ambiguity is a fundamental problem in computer vision. Depending on the particular task, one "irrelevant" factor could be considered as interference with other effective factors. It's thus crucial to understand which factors are relevant to task being accomplished and which are not. The framework herein can be used to extract information related to those factors. For example, in object tracking, the effective factor is movement, while appearance change is the noise. In object classification, the effective factor is object appearance and movement is the noise. Establishing a representation capable of simultaneously decoupling the movement and appearance factors would serve both the movement-based tasks, e.g. tracking and optical flow, as well as appearance-based tasks, e.g. object classification and recognition. Hence, object appearance and object movement are the two sides of the same coin.

Object recognition tasks focus on one side of this coin, the invariant represention of the object appearance. This problem has been well investigated in the last few decades. The most commonly used technique in deep learning to address this problem is spatial pooling, also known as sub-sampling in some literature (Lecun et al., 1998). The technique pools $n \times n$ local patches to a single value by some pre-determined rules for each filter. By training with data augmentation, pooling forces the object A and the shifted A to be equivariant. The output size after pooling is smaller than the input. Currently this is one of the most widely used pooling methods for the large scale neural nets (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014). A different pooling method which has gained more attention recently is subspace pooling, also called feature grouping in some literature, (Hyvärinen and Hoyer, 2000; Hyärinen and Köster, 2007). Rather than pooling over $n \times n$ local patches, this technique pools over n filter channels of the same receptive field. The output size remains the same after pooling but the number of channels decreases. Subspace pooling has an even longer history than spatial pooling. One example is the sum of square pooling in spatio-temporal energy models (Adelson and Bergen, 1985; Heeger, 1987; Heeger, 1988), where energy corresponds to the squared value. This technique applies pooling over the response of two orthogonal Gabor filters (with no feature learning involved). In much of the literature, these filters are commonly referred to as "complex steerable bases" or "quadrature filter pairs".

Both pooling methods reduce the variance along the factor of object appearance successfully. However, any movement information is completely lost since different activations are forced to be equivariant during training. This network will fail to reveal which is the effective factor that makes the image vary, and renders pooling to be only suitable for recognition or other appearance-based tasks. However, in many applications, the two factors, movement and appearance are both important. For example in tracking, we need to determine whether two patches I_t and I_{t+1} on two consecutive frames 1) contain the same object, and 2) their corresponding disparity in $[x, y]$ pixels. Pooling tells us if they contain the same object. However, to estimate the disparity, a separate searching and matching process (e.g. sliding window) is necessary. Instead of building an invariant representation along a

single factor by disregarding the variation of other factors, decoupling these factors could solve the above two issues concurrently without the need for explicit searching and matching.

Given the discussion above, what we would like is a representation which is invariant to the irrelevant factors while only decoupling the relevant factors. This desired representation will decouple movement and appearance to two sets of separate coefficients known as amplitude and phase. The set of amplitude coefficients represents the object appearance as a spatial pattern in a persistent manner independent of object movement; the set of phase coefficients is independent of object appearance, while representing the movement as the object's dynamic location over time. We propose a model to represent these two complementary aspects simultaneously using complex factorization over orthogonal filter pairs. After complex factorization, the phase gradient between frames is specifically tuned for representing object movement between consecutive frames. This representation allows us to learn object movement more effectively by training an end-to-end Convolutional Neural Net (CNN) for optical flow with a significantly smaller amount of labeled data. In this paper, we first introduce the orthogonal filter pairs and how to factorize their filter responses to amplitude and phase. We then describe how to learn these filter pairs from multiple frames using unsupervised learning. Finally, to validate our approach, we attach a separate NN to the factorized phase gradient to estimate optical flow between input frames.

2 Complex factorization

2.1 Amplitude and phase factorization over orthogonal filter pairs

First, we briefly review the commonly seen example of subspace pooling - the sum of squares pooling. Independent Subspace Analysis (ISA) (Hyvärinen and Köster, 2007) is depicted in Figure 1. The input data I are multiple consecutive frames, so variable t is added to figures and equations. For each frame, the sum of squares pooling is computed over all filter responses in the same group. The group size is n=4 in this example, the four filter responses $\{a_{1k}, a_{2k}, a_{3k}, a_{4k}\}$ are pooled to single response s_k for each group. I_i is the current input frame, k is the group number, j is the filter number in each group, n is the group size (total number of filters in each group), i is the number of each input pixel, and m is the input image size (total number of pixels). ω_{ijk}, $j = 1...n$, are the filters, every n-tuple of ω_{ijk} corresponds to n filters in the same group. Each n-tuple of ω_{ijk} spans an independent subspace by a group consisting of n filters. As denoted in Equation (1), a_{jk} are the filter responses for all ω_{ijk} for each frame. s_k are the pooling results which combine n filter responses in the same group by sum of squares pooling. This pooling result indicates whether a specific spatial appearance exists in the current receptive field. It is invariant to small local movements. However, the location of the selected spatial appearance is completely disregarded. So the relative movement of the object between frames is not traceable.

$$\begin{aligned} a_{jk} &= \sum_{i=1}^{m} I_i \omega_{ijk}, \\ s_k &= \sqrt{\Big(\sum_{j=1}^{n} a_{jk}^2\Big)}. \end{aligned} \tag{1}$$

Next, we introduce the complex basis factorization which decouples object appearance and its location to two sets of separate coefficients known as amplitude σ_k and phase φ_k. The factorization process is depicted in Figure 2. The overall procedure is very similar to Independent Subspace Analysis. Complex factorization extends the existing subspace square pooling to the complex domain by the following two operations:

1. Orthogonal filter pairs: the size of each group is $n = 2$, the two filters in the same group are orthogonal. The orthogonal filter pairs ω_{ijk} in the same group are written in complex format

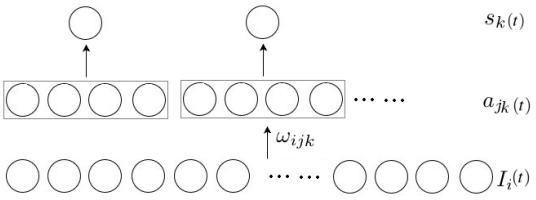

Figure 1: Independent Subspace Analysis (ISA) computes the sum of square pooling by computing the sum of squares over all filter responses in the same group $a_{1k}, a_{2k}, a_{3k}, a_{4k}$, the group size is $n = 4$ in this example.

with real and imaginary parts $\omega_{ijk} = [\omega_{ijk}^{j=real}, \omega_{ijk}^{j=img}]$. Meanwhile the pairwise filter responses in the same group are also in complex form $a_{jk} = [a_{jk}^{j=real}, a_{jk}^{j=img}]$.

2. Complex factorization: the complex pairwise filter responses in the same group $a_{jk} = [a_{jk}^{j=real}, a_{jk}^{j=img}]$ are factorized to polar form $s_k = [\sigma_k, \varphi_k]$ by $[a_{jk}^{j=real}, a_{jk}^{j=img}] = \sigma_k e^{j\varphi_k}$. σ_k is the factorized amplitude and φ_k is the factorized phase.

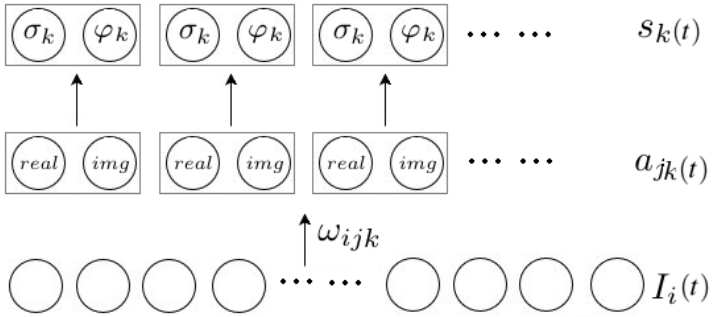

Figure 2: Amplitude and phase decomposition by complex basis factorization over over all filter responses in the same group $\{a_{1k}^{real}, a_{2k}^{img}\}$, the group size is $n = 2$ in this example.

By complex factorization, the image content is decoupled to local amplitude and phase respectively. σ_k is the amplitude coefficient, and indicates whether or not a pattern exists in the receptive field. It is invariant to the object location since it combines the pairwise filter responses in each group by the sum of squares. φ_k is the phase coefficient, it indicates the location of the spatial pattern by the interpolation of the orthogonal filter pair in the same group. The amplitude is similar to ISA when the subspace dimension is $n = 2$. However, complex factorization explicitly represents and explores the phase variable in order to not only provide a sparse, locally invariant representation of image contents but also their location.

As an architecture for representing image and video, complex factorization represents not only the invariant part - the object in the image; but also the varying part - movement which causes the change in pixel domain. Separate amplitude and phase information was initially explored in (Olshausen et al., 2009; Cadieu and Olshausen, 2012) by stacking multiple layers of sparse coding. Here, we adopt a hybrid model consisting of sparse coding and CNN.

2.2 Movement estimation using phase gradient

The object location information is encoded in the phase coefficients. Once the inputs are multiple frames containing a moving object, the movement information is encoded in the phase gradient between consecutive frames. The phase gradient is denoted by Equation (2),

$$\varphi_k, (t) = \varphi_k(t) - \varphi_k(t-1) \tag{2}$$

To demonstrate the phase gradient over time, we first consider an example without feature learning. The filter pairs in each group are orthogonal Gabor filter pairs with different frequency and orientation. First, we generate 288 Gabor filter pairs; each pair has 90 degree phase shift to make them orthogonal. These 288 pairs consist of 24 orientations and 12 frequencies. As seen in Figure 3, they are shown in polar coordinates by their spatial frequency. Each dot represents an orthogonal Gabor filter pair (k=288, n=2). The radial angle is the Gabor filter orientation and the distance to origin is the Gabor filter frequency. E.g. the upper two examples show filter pairs with the same frequency but inverse orientation (reflected in the reversed phase of the 2nd component). The lower example shows a different orientation and higher frequency.

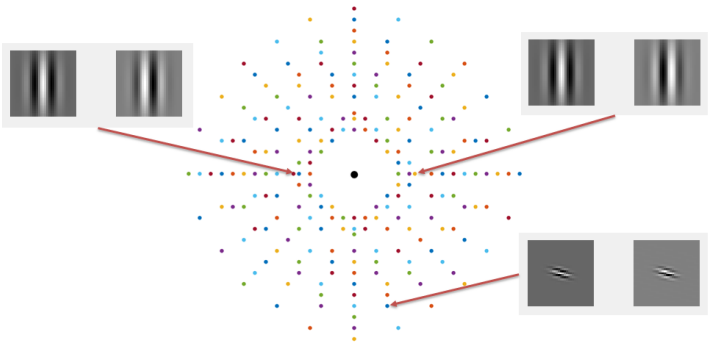

Figure 3: 288 orthogonal Gabor filter pairs in polar coordinates: 288 Gabor filter pairs plotted by their spatial frequency in polar coordinates. The black dot in the middle is the origin. Each color coded dot represent a Gabor filter pair (k=288, n=2), the radial angle is the Gabor filter orientation and the distance to origin is the Gabor filter scale. There are are 24 orientations and 12 scales. They make 288 orthogonal Gabor filter pairs.

An example of phase gradient over time is given in Figure 4. The input video sequence $I_i(t)$ is in the top row which consists of 15 frames. It is an edge moving from left to right, the disparity is 1 pixel to the next frame. In other words, the movement is [1,0] between every two consecutive frames. The factorized amplitude responses for each frame $\sigma_k(t)$ are shown in the middle row. Since the object in all 15 frames is the same (a vertical edge), in polar coordinates, the 15 factorized amplitude coefficients are very similar. The amplitude responses of the 9 frames from $t = 4$ to $t = 12$ are almost identical. The reason that we have this observation is: when the object is near the centre of the receptive field, all Gabor filters have responses; when the object is near the boundary, Gabor filters with high frequencies have no responses. The factorized phase gradient between every two consecutive frames $\phi\prime_k(t)$ is shown in the bottom row. Again, since the difference between frames is the same, [1,0], in polar coordinates, the 14 factorized phase gradients are very similar. For the same reason, the 8 factorized phase gradients between $t_4 - t_3$ and $t_{12} - t_{11}$ are almost identical.

This example shows that when the object is not too far from the centre of the receptive field, 1) the amplitude coefficients result in an invariant representation of the object, 2) the phase gradient captures different types of movement.

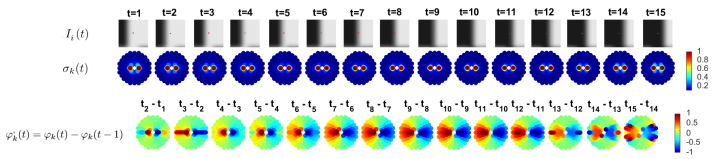

Figure 4: Factorization to amplitude and phase, example of moving edge sequence. $I_i(t)$: Input moving edge sequence consists of 15 frames. $\sigma_k(t)$: Factorized amplitude responses of all Gabor filter pairs for each fame in polar coordinates $\phi\prime_k(t)$: Difference of factorized phase responses of all Gabor filter pairs between every two consecutive fames in polar coordinates.

3 Learning amplitude and phase

In the previous section, we used Gabor filters to demonstrate how to get amplitude and phase from the input image by orthogonal filter pairs. Gabor filters are great example since they have a clean mathematical formulation. However, they are not the best descriptor for all datasets. It has been proven in most classification and recognition tasks, feature learning provides a better fit to natural data and yield better results than hand tuned features (Bengio et al., 2013). Here, we learn the orthogonal filter pairs from images in an unsupervised manner.

$$\begin{aligned} s_k(t) &= \sigma_k(t) e^{j\varphi_k(t)}, \\ a_{jk}^{j=real(t)} &= \sigma_k(t)\cos\varphi_k(t), \\ a_{jk}^{j=img(t)} &= \sigma_k(t)\sin\varphi_k(t), \\ I_i^R(t) &= \sum_k \sigma_k(t)[\cos\varphi_k(t)\omega_{ijk}^{j=real} + \sin\varphi_k(t)\omega_{ijk}^{j=img}] \end{aligned} \tag{3}$$

We use modified sparse coding to learn the orthogonal filter pairs. This is defined in Equation (3), where $\sigma_k(t)$ and $\varphi_k(t)$ are the factorized amplitude and phase, $a_{jk}(t)$ is the complex coefficient and $I_i^R(t)$ is the reconstruction. R indicates the real part is taken as the reconstructed image, for each frame, $I_i^R(t) = \sum_{kReal}[a_{jk}\omega_{ijk}]$. This generative procedure is depicted in Figure 5: Left. We choose sparse coding since it requires less training data compared to other learning models, eg, Autoencoder.

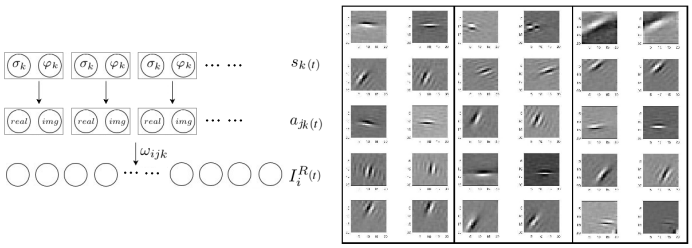

Figure 5: Left: Sparse coding model for learning the orthogonal filter pairs. Right: (Subset of) Learned orthogonal filter pairs.

The cost function is defined as Equation (4). There are three terms in the cost function: reconstruction error, spatial sparseness regularization, and temporal smoothness regularization, all three terms are with L^2 norm:

$$L(I,\sigma,\omega) = \sum k,t[I_i(t) - I_i^R(t)]^2 + \lambda \sum_{k,t}[\sigma_k(t)]^2 + \beta \sum_{k,t}[\sigma_k(t) - \sigma_k(t-1)]^2 \tag{4}$$

In the first term, I_i^R is the reconstruction which is generated by Equation (3). In the two regularization terms, the λ and β are scaling constants to determine the penalties corresponding to sparseness and temporal smoothness. The two regularization terms ensure the distribution of both the amplitude itself and its derivative between frames peak at zero and have high kurtosis. Both regularizations are on amplitude, there is no regularization on phase, which means the prior of phase is a uniform distribution. During learning, it drives the filters $\omega_{ijk}^{j=real}$ and $\omega_{ijk}^{j=img}$ to achieve a sparse and smooth representation on the amplitude. So far, the characteristic and statistics of phase are still not completely clear, more investigation is required. The amplitude and phase are solved by minimizing this cost function over gradient descent. The learning algorithm is very similar to sparse coding: first get a sample of amplitude and phase, then generate a reconstruction, finally update the weights. The gradient of amplitude, phase and weights are defined by Equation (Cadieu and Olshausen, 2012).

The training data are multiple consecutive frames with object movement. The learned filter pairs are localized, oriented bandpass filters. A subset of the learned filter pairs is shown in Figure 5:

Right. Each two filters in the same group share similar location, orientation and frequency. They are orthogonal pairs, with a 90 degree phase shift between the two filters in the same group. This orthogonality is not enforced in learning, it appears to match the characteristic of the training data frames.

4 Experiment: Optical flow estimation by phase gradient

To validate the learned results, we could either test if the amplitude component stays invariant while the object makes small moves or test that movement has been encoded in the phase gradient. Our experiment focuses on testing the phase gradient since square pooling has been well studied and tested in the existing literature. We choose optical flow as the experiment for validation because not only is optical flow a standard way to represent object movement, but it also has several datasets with benchmarks.

Since 2010, there is a tendency to use deep learning based networks to solve the optical flow problem in an end to end manner. DeepFlow (Weinzaepfel et al., 2013) is one of the early works that applies sparse convolutions and max-pooling for optical flow estimation. It aggregates from fine to coarse as a pyramid process to deal with different disparities. However, it does not perform any learning, all parameters are hand tuned. Soon after, FlowNet (Fischer et al., 2015) made significant progress over DeepFlow. It applies a standard CNN architecture with 9 filter and 9 pooling layers to extract motion. On training, they use a modified version of the caffe (Jia et al., 2014) framework. In contrast, CNNFlow (Teney and Hebert, 2016) uses a much shallower network than FlowNet but yields a better average end point error (AEE) and average angular error (AAE). CNNFlow not only relies on extracting information by CNN, but also leverages the fundamental characteristics of vision, such as brightness invariance by normalization, phase invariance by max pooling and rotation invariance by enforcing through weight sharing. These tricks do not violate the CNN architecture; the normalization, pooling and weight sharing can be integrated with CNN training seamlessly.

In this experiment, We infer the optical flow from the factorized phase gradient which is denoted by Equation (2). Since the movement between consecutive frames is encoded by the factorized phase gradient, we attach a CNN to the factorized phase gradient. This CNN works as a decoder, the output is a dense optical flow (2D vector for every pixel). The flow chart is shown in Figure (6). Since the movement has been extracted by complex factorization, the CNN is only a decoder, hence, we make this CNN as simple as possible. It consists of only three layers, one hidden layer, one Softmax layer and one linear output layer:

$$
\begin{aligned}
\frac{dL}{d\sigma_k} &= -\sum_{k,t}\left[\cos\varphi_k(t)\omega_{ijk}^{j=real}+\sin\varphi_k(t)\omega_{ijk}^{j=img}\right]\left[I_i(t)-I_i^R(t)\right]+\lambda\sum_{k,t}\sigma_k(t)+\beta\sum_{k,t}\left[\sigma_k(t)-\sigma_k(t-1)\right]\\
\frac{dL}{d\varphi_k} &= -\sum_{k,t}\sigma_k(t)\left[-\sin\varphi_k(t)\omega_{ijk}^{j=real}+\cos\varphi_k(t)\omega_{ijk}^{j=img}\right]\left[I_i(t)-I_i^R(t)\right]\\
\frac{dL}{d\omega_{ijk}} &= -\sum_{k,t}\left(a_{jk}^{j=real}\left[I_i(t)-I_i^R(t)\right]+a_{jk}^{j=img}\left[I_i(t)-I_i^R(t)\right]\right)
\end{aligned}
\tag{5}
$$

$$
\begin{aligned}
x_j^1 &= \sum_k \varphi\prime_k,(t)\omega_{jk},\\
x_j^2 &= e^{x_j^1}/\sum_j e^{x_j^1},\\
x_i^3 &= \sum_j x_j^2\omega_{ij}.
\end{aligned}
\tag{6}
$$

The hidden layer maps the phase gradient to scores. Since each pixel associates a single movement (uni-modal), the Softmax layer takes the maximum score. Finally, the output layer maps them to

a 2D flow vector for every pixel. The mathematical formulation of those three layers is described in Equation (4). The training of this CNN is supervised learning, optical flow labels are required. Compared to the existing model FlowNet (Fischer et al., 2015) and CNNFlow (Teney and Hebert, 2016), this CNN is a lot shallower. It makes the training of this CNN much easier and requires less training data.

Note that the learning of amplitude and phase is unsupervised. This CNN is the only part that needs labeled data, so, overall our model requires significantly less labeled data than existing CNN based optical flow models.

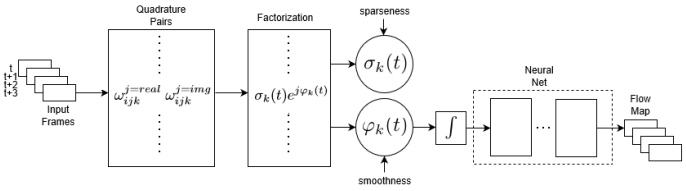

Figure 6: Flow chart of optical flow estimation by phase gradient.

The evaluation of all methods is done on a Xeon E5-1620 v3 at 3.5GHz with an Nvidia Tesla K40. The dataset is the Middlebury dataset (Baker et al., 2011) which was listed originally in 2002 and still remains one of the most important datasets published in the last decade. We use 6 randomly selected image pairs for training and another 8 randomly selected pairs for testing. Visual comparison results are shown in Figure 7. The top row are the ground truth, the middle row are the results of CNNFlow (Teney and Hebert, 2016) and the bottom row are the results of our algorithm - PhaseFlow. The figures indicate that both algorithms maintain the contour of the main object in the flow map well. All algorithms generate some artificial errors, such as on the fifth row bottom column. Phase Flow makes more errors in the green area than CNN Flow; in the sixth row middle column, CNN Flow makes more errors in the yellow area than Phase Flow. However, some details are lost in both algorithms, e.g the details in third column. The error measurements of optical flow are endpoint error and angular error. Endpoint error (in pixels) between two flow vectors (u_0, v_0) and (u_1, v_1) is their absolute error defined by $\sqrt{\left((u_0 - u_1)^2 + (v_0 - v_1)^2\right)}$. Angular error (in degrees) between two flows (u_0, v_0) and (u_1, v_1) is their relative measurement defined by the angle of $(u_0, v_0, 1)$ and $(u_1, v_1, 1)$ in three dimensions. Quantitive evaluation by AEE and AAE on Middlebury and Sintel datasets are shown in Table 1. The performance of Phase Flow is on par with the state of art algorithms on Middlebury dataset. The AEE results on Sintel and KITTI are somewhat weaker than the other algorithms. Phase gradient captures small motions accurately. However, AEE is dominated by large motions, which is currently a disadvantage of our algorithm, we are addressing this limitation in our current work.

Table 1: Average endpoint errors (in pixels) and average angular error (in degrees) for the three methods on testing set in three datasets.

	Deep Flow	FlowNetS with refinement	CNN Flow	Phase Flow
Middlebury Private AEE	0.42	0.47	0.58	0.69
Middlebury Private AAE	4.22	4.58	5.22	8.41
Sintel Private Clean AEE	4.44	4.56	9.46	10.49
Sintel Private Final AEE	7.76	7.21	10.14	14.92
KITTI 2012	5.78	9.14	9.92	14.23

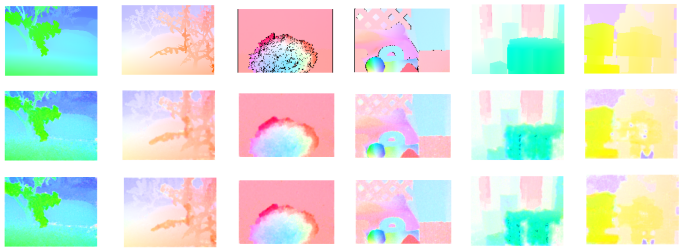

Figure 7: Examples of optical flow results on Middlebury dataset. Optical flow results are colour coded by the standard "Flow Field Color Coding" schema described (Baker et al., 2007). Top row: Ground truth; Middle row: CNNFlow (Teney and Hebert, 2016); Bottom row: PhaseFlow.

5 Conclusion

In this paper, we proposed an algorithm to learn complex factorization which decouples the object appearance and location to amplitude and phase with the use of orthogonal filter pairs. The complex factorization maintains selectivity on frequency and orientation while achieving local phase invariance by computing the amplitude component which implicitly pools over phase. This process naturally separates the factors to independent abstract spaces which establishes a foundation for an unified framework to accomplish the complementary appearance-based and movement-based tasks integrally. However, the amplitude component is invariant only to local orientation and spatial frequency structure. It has limited capacity for handling 3D variations such as projective distortion. Orthogonality of the filter pairs is not enforced in training. This phenomenon is observed in the learned filter pairs, but further research is required to validate othogonality and provide a concrete proof. As a proof of concept, optical flow is extracted from the phase gradient between frames. From the results, we show that small movements and disparities are encoded in the phase gradient successfully. To extend to larger disparity, one proposed solution is build a coarse to fine process to extract the movements at different scales by a pyramid strategy. This is a commonly used mechanism in computer vision. Another approach is to improve pooling by feature pairs with not only 90 degree phase shift, but also location shift where the small local movements are modelled by phase shift and large global movements are modelled by location shift. In this case, since the group size is greater than 2, the factorization will be on on quaternions rather than complex numbers, hence learning will be a considerable challenge. As a hybrid model, the complex factorization and CNNs are trained separately. It's difficult to enforce orthogonality directly in regular CNN since it could violate the existing CNN training protocol. Further research is required to develop a unified framework that allows optimization and fine tuning over the entire model.

5.1 References

Adelson, E.H., and Bergen, J.R. (1985) Spatiotemporal energy models for the perception of motion. JOSA A, 2(2):284–299

Baker, S., Roth, S., Scharstein, D., Black, M.J., Lewis, J.P., and Szeliski, R. (2007) A database and evaluation methodology for optical flow. In: 2007 IEEE 11th International Conference on Computer Vision, pages 1–8

Baker, S., Scharstein, D., Lewis, J.P., Roth, Stefan, Black, Michael, J., and Szeliski, Richard. (2011) A database and evaluation methodology for optical flow. International Journal of Computer Vision, 92(1):1–31

Bengio, Y., Courville, A., and Vincent, P. (2013) Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8):1798–1828

Cadieu, C.F., and Olshausen, B.A. (2012) Learning intermediate-level representations of form and motion from natural movies. Neural computation, 24(4):827–866

Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., van der Smagt, P., Cremers, D. and Brox, T. (2015) Flownet: Learning optical flow with convolutional networks. arXiv preprint arXiv:1504.06852

Heeger, D.J. (1987) Model for the extraction of image flow. JOSA A, 4(8):1455–1471

Heeger, D.J. (1988) Optical flow using spatiotemporal filters. International Journal of Computer Vision, 1(4):279–302

Hyvärinen, A., and Hoyer, P. (2000) Emergence of phase-and shift-invariant features by decomposition of natural images into independent feature subspaces. Neural computation, 12(7):1705–1720

Hyvärinen, A., and Köster, U. (2007) Complex cell pooling and the statistics of natural images. Network: Computation in Neural Systems, 18(2):81–100

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014) Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093

Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks. In: F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, pages 2278–2324

Olshausen, B.A., Cadieu, C.F., and Warland, D.K. (2009) Learning real and complex overcomplete representations from the statistics of natural images. In: SPIE Optical Engineering + Applications, pages 74460S–74460S. International Society for Optics and Photonics

Simonyan, K., and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556

Teney, D., and Hebert., M. (2016) Learning to extract motion from videos in convolutional neural networks. arXiv preprint arXiv:1601.07532

Weinzaepfel, P., Revaud, J., Harchaoui, Z. and Schmid, C. (2013) Deepflow: Large displacement optical flow with deep matching. In: Proceedings of the IEEE International Conference on Computer Vision, pages 1385–1392