Learning V1 Simple Cells with Vector Representation of Local Content and Matrix Representation of Local Motion

Ruiqi Gao,[1*] **Jianwen Xie,** [2] **Siyuan Huang,** [3] **Yufan Ren,**[4] **Song-Chun Zhu,** [1,3] **Ying Nian Wu** [1]

[1] UCLA, [2] Baidu Research, [3] Beijing Institute for General Artificial Intelligence (BIGAI),
[4] École Polytechnique Fédérale de Lausanne (EPFL)
ruiqig@google.com, jianwen@ucla.edu, huangsiyuan@ucla.edu, yufan.ren@epfl.ch, sczhu@stat.ucla.edu, ywu@stat.ucla.edu

Abstract

This paper proposes a representational model for image pairs such as consecutive video frames that are related by local pixel displacements, in the hope that the model may shed light on motion perception in primary visual cortex (V1). The model couples the following two components: (1) the vector representations of local contents of images and (2) the matrix representations of local pixel displacements caused by the relative motions between the agent and the objects in the 3D scene. When the image frame undergoes changes due to local pixel displacements, the vectors are multiplied by the matrices that represent the local displacements. Thus the vector representation is equivariant as it varies according to the local displacements. Our experiments show that our model can learn Gabor-like filter pairs of quadrature phases. The profiles of the learned filters match those of simple cells in Macaque V1. Moreover, we demonstrate that the model can learn to infer local motions in either a supervised or unsupervised manner. With such a simple model, we achieve competitive results on optical flow estimation.

1 Introduction

Our understanding of the primary visual cortex or V1 (Hubel and Wiesel 1959) is still very limited (Olshausen and Field 2005). In particular, mathematical and representational models for V1 are still in short supply. Two prominent examples of such models are sparse coding (Olshausen and Field 1997) and independent component analysis (ICA) (Bell and Sejnowski 1997). Although such models may not provide detailed explanations at the level of neuronal dynamics, they help us understand the computational problems being solved by V1.

In this paper, we propose a model of this sort. It is a representational model of natural image pairs that are related by local pixel displacements. The image pairs can be consecutive frames of a video sequence, where the local pixel displacements are caused by the relative motions between the agent and the objects in the 3D environment. Perceiving such local motions can be crucial for inferring ego-motion, object motions, and 3D depth information.

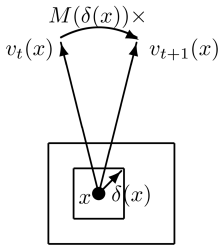


Figure 1: Scheme of representation. The image is illustrated by the big rectangle. A pixel is illustrated by a dot. The local image content is illustrated by a small square around it. The displacement of the pixel is illustrated by a short arrow, which is within the small square. The vector representation of the local image content is represented by a long vector, which is equivariant because it rotates as the image undergoes deformation due to the pixel displacements. The rotation is realized by a matrix representation of the local motion. See Section 3 for the detailed notation.

As is the case with existing models, we expect our model to explain only partial aspects of V1, including: (1) The receptive fields of V1 simple cells resemble Gabor filters (Daugman 1985). (2) Adjacent simple cells have quadrature phase relationship (Pollen and Ronner 1981; Emerson and Huang 1997). (3) The V1 cells are capable of perceiving local motions. While existing models can all explain (1), our model can also account for (2) and (3) naturally. Compared to models such as sparse coding and ICA, our model has a component that serves a direct purpose of perceiving local motions.

Our model consists of the following two components.

(1) Vector representation of local image content. The local content around each pixel is represented by a high dimensional vector. Each unit in the vector is obtained by a linear filter. These local filters or wavelets are assumed to form a normalized wavelet tight frame, i.e., the image can be reconstructed from the vectors using the linear filters as the basis functions.

(2) Matrix representation of local displacement. The change of the image from the current time frame to the next time frame is caused by the displacements of the pixels. Each possible displacement is represented by a matrix that operates on the vector. When the image changes according

[*]The author is now a Research Scientist at Google Research, Brain team.

to the displacements, the vector at each pixel is multiplied by the matrix that represents the local displacement, in other words, the vector at each pixel is rotated by the matrix representation of the displacement of this pixel. Thus the vector representation is equivariant as it varies according to the local displacements. See Fig. 1 for an illustration.

We train the model on image pairs where in each pair, the second image is a deformed version of the first image, and the deformation is either known or inferred in training. We learn the encoding matrices for vector representation and the matrix representation of local displacements from the training data.

Experiments show that our model learns V1-like units that can be well approximated by Gabor filters with quadrature phase relationship. The profiles of learned units match those of simples cells in Macaque V1. After learning the encoding matrices for vector representation and the matrix representations of the displacements, we can infer the displacement field using the learned model. Compared to popular optical flow estimation methods (Dosovitskiy et al. 2015; Ilg et al. 2017), which use complex deep neural networks to predict the optical flows, our model is much simpler and is based on explicit vector and matrix representations. We demonstrate comparable performance to these methods in terms of the inference of displacement fields.

In terms of biological interpretation, the vectors can be interpreted as activities of groups of neurons, and the matrices can be interpreted as synaptic connections. See Sections 5.1 and 5.2 for details.

2 Contributions and related work

This paper proposes a simple representational model that couples the vector representations of local image contents and matrix representations of local pixel displacements. The model explains certain aspects of V1 simple cells such as Gabor-like receptive fields and quadrature phase relationship, and adds to our understanding of V1 motion perception in terms of representation learning.

The following are two themes of related work.

(1) V1 models. Most well known models for V1 are concerned with statistical properties of natural images or video sequences. Examples include sparse coding model (Olshausen and Field 1997; Lewicki and Olshausen 1999; Olshausen 2003), independent component analysis (ICA) (Hyvärinen, Karhunen, and Oja 2004; Bell and Sejnowski 1997; van Hateren and Ruderman 1998), slowness criterion (Hyvärinen, Hurri, and Väyrynen 2003; Wiskott and Sejnowski 2002), and prediction (Singer et al. 2018). While these models are very compelling, they do not account for perceptual inference explicitly, which is the most important functionality of V1. On the other hand, our model is learned for the direct purpose of perceiving local motions caused by relative motion between the agent and the surrounding 3D environment. In fact, our model is complementary to the existing models for V1: similar to existing models, our work assumes a linear generative model for image frames, but our model adds a relational component with matrix representation that relates the consecutive image frames. Our model is also complementary to slowness criterion in that when the vectors are rotated by matrices, the norms of the vectors remain constant.

(2) Matrix representation. In representation learning, it is a common practice to encode the signals or states as vectors. However, it is a much less explored theme to represent the motions, actions or relations by matrices that act on the vectors. An early work in this theme is (Paccanaro and Hinton 2001), which learns matrices to represent relations. More recently, (Jayaraman and Grauman 2015) learns equivariant representation with matrix representation of ego-motion. (Zhu et al. 2021) learns generative models of posed images based on invariant representation of 3D scene with matrix representation of ego-motion (Gao et al. 2019, 2021) learn vector representation of self-position and matrix representation of self-motion in a representational model of grid cells. Our work constitutes a new development along this theme.

3 Representational model

3.1 Vector representation

Let $\{\mathbf{I}(x), x \in D\}$ be an image observed at a certain instant, where $x = (x_1, x_2) \in D$ is the 2D coordinates of pixel. D is the image domain (e.g., 128×128). We represent the image $\mathbf{I}$ by vectors $\{v(x), x \in D_-\}$, where each $v(x)$ is a vector defined at pixel x, and D_- may consist of a sub-sampled set of pixels in D. $V = \{v(x), x \in D_-\}$ forms a vector representation of the whole image.

Conventionally, we assume the vector encoding is linear and convolutional. Specifically, let $\mathbf{I}[x]$ be a squared patch (e.g., 16×16) of $\mathbf{I}$ centered at x. We can flatten $\mathbf{I}[x]$ into a vector (e.g., 256 dimensional) and let

$$v(x) = W\mathbf{I}[x],\ x \in D_-, \tag{1}$$

be the linear encoder, where W is the encoding matrix that encodes $\mathbf{I}[x]$ into a vector $v(x)$. The rows of W are the linear filters and can be displayed as local image patches of the same size as the image patch $\mathbf{I}[x]$. We can further write $V = \mathbf{WI}$ if we treat $\mathbf{I}$ as a vector, and the rows of $\mathbf{W}$ are the translated versions of W.

3.2 Tight frame auto-encoder

We assume that $\mathbf{W}$ is an auto-encoding tight frame. Specifically, let $W(x)$ denote the translation of filter W to pixel x and zero-padding the pixels outside the filters, so that each row of $W(x)$ is of the same dimension as $\mathbf{I}$. We then assume

$$\mathbf{I} = \mathbf{W}^\top V = \sum_{x \in D_-} W^\top(x) v(x), \tag{2}$$

i.e., the linear filters for bottom-up encoding also serve as basis functions for top-down decoding. Both the encoder and decoder can be implemented by convolutional linear neural networks.

The tight frame assumption can be justified by the fact that under that assumption, for two images $\mathbf{I}$ and $\mathbf{J}$, we have $\langle \mathbf{WI}, \mathbf{WJ} \rangle = \mathbf{I}^\top \mathbf{W}^\top \mathbf{WJ} = \langle \mathbf{I}, \mathbf{J} \rangle$, i.e., the vector representations preserve the inner product. As a result, we also have $\|\mathbf{WI}\| = \|\mathbf{I}\|$ and $\|\mathbf{WJ}\| = \|\mathbf{J}\|$, so that the vector representations $\mathbf{WI}$ and $\mathbf{WJ}$ preserve the angle between the

images $\mathbf{I}$ and $\mathbf{J}$ and has the isometry property. That is, when the image $\mathbf{I}$ changes from $\mathbf{I}_t$ to $\mathbf{I}_{t+1}$, its vector representation V changes from V_t to V_{t+1}, and the angle between $\mathbf{I}_t$ and $\mathbf{I}_{t+1}$ is the same as the angle between V_t and V_{t+1}.

In this paper, we assume a tight frame auto-encoder for computational convenience. A more principled treatment is to treat the decoder as a top-down generative model, and treat the encoder as approximate inference. Sparsity constraint can be imposed on the top-down decoder model.

3.3 Matrix representation

Let $\mathbf{I}_t$ be the image at time frame t. Suppose the pixels of $\mathbf{I}_t$ undergo local displacements $(\delta(x), \forall x)$, where $\delta(x)$ is the displacement at pixel x. The image transforms from $\mathbf{I}_t$ to $\mathbf{I}_{t+1}$. We assume that $\delta(x)$ is within a squared range Δ (e.g., $[-6, 6] \times [-6, 6]$ pixels) that is inside the range of the local image patch $\mathbf{I}_t[x]$ (e.g., 16×16 pixels). We assume that the displacement field $(\delta(x), \forall x)$ is locally smooth, i.e., pixels within each local image patch undergoes similar displacements. Let $v_t(x)$ and $v_{t+1}(x)$ be the vector representations of $\mathbf{I}_t[x]$ and $\mathbf{I}_{t+1}[x]$ respectively. The transformation from $\mathbf{I}_t[x]$ to $\mathbf{I}_{t+1}[x]$ is illustrated by the following diagram:

$$\begin{array}{ccc} v_t(x) & \xrightarrow{M(\delta(x))\times} & v_{t+1}(x) \\ W \uparrow & \uparrow & \uparrow W \\ \mathbf{I}_t[x] & \xrightarrow{\delta(x)} & \mathbf{I}_{t+1}[x] \end{array} \tag{3}$$

Specifically, we assume that

$$v_{t+1}(x) = M(\delta(x))v_t(x), \ \forall x \in D_-. \tag{4}$$

That is, when $\mathbf{I}$ changes from $\mathbf{I}_t$ to $\mathbf{I}_{t+1}$, $v(x)$ undergoes a linear transformation, driven by a matrix $M(\delta(x))$, which depends on the local displacement $\delta(x)$. Thus $v(x)$ is an equivariant representation as it varies according to $\delta(x)$.

One motivation for modeling the transformation as a matrix representation operating on the vector representation of image patch comes from Fourier analysis. Specifically, an image patch $\mathbf{I}[x]$ can be expressed by the Fourier decomposition $\mathbf{I}[x] = \sum_k c_k e^{i\langle \omega_k, x\rangle}$. Assuming all the pixels in the patch are shifted by a constant displacement $\delta(x)$, then the shifted image patch becomes $\mathbf{I}(x - \delta(x)) = \sum_k c_k e^{-i\langle \omega_k, \delta(x)\rangle} e^{i\langle \omega_k, x\rangle}$. The change from the complex number c_k to $c_k e^{-i\langle \omega_k, dx\rangle}$ corresponds to rotating a 2D vector by a 2×2 matrix. However, we emphasize that our model does not assume Fourier basis or its localized version such as Gabor filters. The model figures it out with generic vector and matrix representations.

Parametrization. We consider two ways to parametrize the matrix representation of local displacement $M(\delta(x))$. First, we can discretize the displacement $\delta(x)$ into a finite set of possible values $\{\delta\}$, and we learn a separate $M(\delta)$ for each δ. Second, We can learn a parametric version of $M(\delta)$ as the second order Taylor expansion of a matrix-valued function of $\delta = (\delta_1, \delta_2)$,

$$M(\delta) = I + B_1\delta_1 + B_2\delta_2 + B_{11}\delta_1^2 + B_{22}\delta_2^2 + B_{12}\delta_1\delta_2. \tag{5}$$

In the above expansion, I is the identity matrix, and $B = (B_1, B_2, B_{11}, B_{22}, B_{12})$ are matrices of coefficients of the same dimensionality as $M(\delta)$.

A more careful treatment is to treat $M(\delta)$ as forming a matrix Lie group, and write $M(\delta)$ as an exponential map of generator matrices that span the matrix Lie algebra. By assuming skew-symmetric generator matrices, $M(\delta)$ will be automatically a rotation matrix. The exponential map can be approximated by Taylor expansion similar to 5.

Local mixing. If $\delta(x)$ is large, $v_{t+1}(x)$ may contain information from adjacent image patches of $\mathbf{I}_t$ in addition to $\mathbf{I}_t[x]$. To address this problem, we can generalize the motion model (Eq. (4)) to allow local mixing of encoded vectors. Specifically, let $\mathcal{S}$ be a local support centered at 0. We assume that

$$v_{t+1}(x) = \sum_{\mathrm{d}x \in \mathcal{S}} M(\delta(x), \mathrm{d}x) v_t(x + \mathrm{d}x) \tag{6}$$

In the learning algorithm, we discretize $\mathrm{d}x$ and learn a separate $M(\delta, \mathrm{d}x)$ for each $\mathrm{d}x$.

3.4 Sub-vectors and disentangled rotations

The vector $v(x)$ can be high-dimensional. For computational efficiency, we further divide $v(x)$ into K sub-vectors, $v(x) = (v^{(k)}(x), k = 1, ..., K)$. Each sub-vector is obtained by an encoding sub-matrix $W^{(k)}$, i.e.,

$$v^{(k)}(x) = W^{(k)}\mathbf{I}[x], \ k = 1, ..., K, \tag{7}$$

where $W^{(k)}$ consists of the rows of W that correspond to $v^{(k)}$. In practice, we find that this assumption is necessary for the emergence of V1-like receptive field.

Correspondingly, the matrix representation

$$M(\delta) = \mathrm{diag}(M^{(k)}(\delta), k = 1, ..., K) \tag{8}$$

is block diagonal. Each sub-vector $v^{(k)}(x)$ is transformed by its own sub-matrix $M^{(k)}(\delta)$:

$$v_{t+1}^{(k)}(x) = M^{(k)}(\delta) v_t^{(k)}(x), \ k = 1, ..., K. \tag{9}$$

The linear transformations of the sub-vectors $v^{(k)}(x)$ can be considered as rotations. $v(x)$ is like a multi-arm clock, with each arm $v^{(k)}(x)$ rotated by $M^{(k)}(\delta(x))$. The rotations of $v^{(k)}(x)$ for different k and x are disentangled, meaning that the rotation of a sub-vector does not depend on other sub-vectors.

The assumption of sub-vectors and block-diagonal matrices is necessary for learning Gabor-like filters. For $v_{t+1}(x) = M(\delta(x))v_t(x)$, it is equivalent to $\tilde{v}_{t+1}(x) = \tilde{M}(\delta(x))\tilde{v}_t(x)$ if we let $\tilde{v}_t(x) = Pv_t(x)$, $\tilde{v}_{t+1}(x) = Pv_{t+1}(x)$, and $\tilde{M}(\delta(x)) = PM(\delta(x))P^{-1}$, for an invertible P. Assuming block-diagonal matrices helps eliminates such ambiguity. More formally, a matrix representation is irreducible if it cannot be further diagonalized into smaller block matrices. Assuming block-diagonal matrices helps to make the matrix representation close to irreducible.

4 Learning and inference

4.1 Supervised learning

The input data consist of the triplets $(\mathbf{I}_t, (\delta(x), x \in D_-), \mathbf{I}_{t+1})$, where $(\delta(x), x \in D_-)$ is the given displacement field. The unknown parameters to learn consist of matrices $(W, M(\delta), \delta \in \Delta)$, where Δ is the range of δ. In the case of parametric M, we learn the B matrices in the second order Taylor expansion (Eq. (5)). We assume that there are K sub-vectors so that M or B are block-diagonal matrices. We learn the model by optimizing a loss function defined as a weighted sum of two loss terms, based on the linear transformation model (Eq. (4)) and tight frame assumption (Eq. (2)) respectively:

(1) Linear transformation loss

$$L_1 = \sum_{k=1}^{K} \sum_{x \in D_-} \left\| W^{(k)} \mathbf{I}_{t+1}[x] - M^{(k)}(\delta(x)) W^{(k)} \mathbf{I}_t[x] \right\|^2. \quad (10)$$

For local mixing generalization, we substitute $M^{(k)}(\delta(x))$ by $\sum_{\mathrm{d}x \in \mathcal{S}} M^{(k)}(\delta(x), \mathrm{d}x)$.

(2) Tight frame auto-encoder loss

$$L_2 = \sum_{s \in t, t+1} \left\| \mathbf{I}_s - \mathbf{W}^\top \mathbf{W} \mathbf{I}_s \right\|^2. \quad (11)$$

4.2 Inference of motion

After learning $(W, M(\delta), \delta \in \Delta)$, given a testing pair $(\mathbf{I}_t, \mathbf{I}_{t+1})$, we can infer the pixel displacement field $(\delta(x), x \in D_-)$ by minimizing the linear transformation loss: $\delta(x) = \arg\max_{\delta \in \Delta} L_{1,x}(\delta)$, where

$$L_{1,x}(\delta) = \textstyle\sum_{k=1}^{K} \left\| W^{(k)} \mathbf{I}_{t+1}[x] - M^{(k)}(\delta) W^{(k)} \mathbf{I}_t[x] \right\|^2. \quad (12)$$

This algorithm is efficient in nature as it can be parallelized for all $x \in D_-$ and for all $\delta \in \Delta$.

If we use a parametric version of $(M(\delta), \delta \in \Delta)$ (Eq. (5)), we can minimize $\sum_x L_{1,x}(\delta)$ using gradient descent with δ initialized from random small values. To encourage the smoothness of the inferred displacement field, we add a penalty term $\|\nabla \delta(x)\|^2$ for this setting.

4.3 Unsupervised learning

We can easily adapt the learning of the model to an unsupervised manner, without knowing the pixel displacement field $(\delta(x), x \in D_-)$. Specifically, we can iterate the following two steps: (1) update model parameters by loss functions defined in Section 4.1; (2) infer the displacement field as described in 4.2. To eliminate the ambiguity of $M(\delta)$ with respect to δ, we add a regularization term $\|\mathbf{I}_{t+1} - \mathrm{warp}(\mathbf{I}_t, \delta)\|^2$ in the inference step, where $\mathrm{warp}(\cdot, \cdot)$ is a differentiable warping function, and we use the parametric version of $(M(\delta), \delta \in \Delta)$. To summarize, we infer the displacement field $(\delta(x), x \in D_-)$ by minimizing:

$$\sum\nolimits_x L_{1,x}(\delta) + \|\nabla \delta\|^2 + \|\mathbf{I}_{t+1} - \mathrm{warp}(\mathbf{I}_t, \delta)\|^2. \quad (13)$$

In practice, for each image pair at each iteration, we start the inference by running gradient descent on the inferred displacement field from the previous iteration.

5 Discussions about model

5.1 Biological interpretations of cells and synaptic connections

The learned $(W, M(\delta)), \delta)$ can be interpreted as synaptic connections. Specifically, for each block k, $W^{(k)}$ corresponds to one set of connection weights. Suppose $\delta \in \Delta$ is discretized, then for each δ, $M^{(k)}(\delta)$ corresponds to one set of connection weights, and $(M^{(k)}(\delta), \delta \in \Delta)$ corresponds to multiple sets of connection weights. For motion inference in a biological system, after computing $v_{t,x}^{(k)} = W^{(k)} \mathbf{I}_t[x]$, $M^{(k)}(\delta) v_{t,x}^{(k)}$ can be computed simultaneously for every $\delta \in \Delta$. Then $\delta(x)$ is inferred by max pooling according to Eq. (12).

$v_{t,x}^{(k)}$ can be interpreted as activities of simple cells, and $\|v_{t,x}^{(k)}\|^2$ can be interpreted as activity of a complex cell. If $M^{(k)}(\delta)$ is close to a rotation matrix, then we have norm stability so that $\|v_{t,x}^{(k)}\| \approx \|v_{t+1,x}^{(k)}\|$, which is closely related to the slowness property (Hyvärinen, Hurri, and Väyrynen 2003; Wiskott and Sejnowski 2002).

5.2 Spatiotemporal filters and recurrent implementation

If we enforce norm stability or the orthogonality of $M^{(k)}(\delta)$, then minimizing $\|v_{t+1,x} - M(\delta) v_{t,x}\|^2$ over $\delta \in \Delta$ is equivalent to maximizing $\langle v_{t+1,x}, M(\delta) v_{t,x} \rangle$, which in turn is equivalent to maximizing $\|v_{t+1,x} + M(\delta) v_{t,x}\|^2$ so that $v_{t+1,x}$ and $M(\delta) v_{t,x}$ are aligned. This alignment criterion can be conveniently generalized to multiple consecutive frames, so that we can estimate the velocity at x by maximizing the m-step alignment score $\|u\|^2$, where

$$u = \sum_{i=0}^{m} M(\delta)^{m-i} v_{t+i,x} = \sum_{i=0}^{m} M(\delta)^{m-i} W \mathbf{I}_{t+i}[x] \quad (14)$$

consists of responses of spatiotemporal filters or "animated" filters $(M(\delta)^{m-i} W, i = 0, ..., m)$, and $\|u\|^2$ corresponds to the energy of motion δ in the motion energy model (Adelson and Bergen 1985) for direction selective cells. Thus our model is connected with the motion energy model. Moreover, our model enables a recurrent network for computing u by $u_i = v_{t+i,x} + M(\delta) u_{i-1}$ for $i = 0, ..., m$, with $u_{-1} = 0$, and $u = u_m$. This recurrent implementation is much more efficient and biologically plausible than the plain implementation of spatiotemporal filtering which requires memorizing all the $\mathbf{I}_{t+i}$ for $i = 0, ..., m$. See (Pachitariu and Sahani 2017) for a discussion of biological plausibility of recurrent implementation of spatiotemporal filtering in general.

The spatiotemporal filters can also serve as spatiotemporal basis functions for the top-down decoder model.

6 Experiments

The code, data and more results can be found at http://www.stat.ucla.edu/~ruiqigao/v1/main.html

We learn our model $(W, M(\delta), \delta \in \Delta)$ from image pairs $(\mathbf{I}_t, \mathbf{I}_{t+1})$ with its displacement field $(\delta(x))$ known or un-

known. The number of sub-vectors $K = 40$, and the number of units in each sub-vector $v^{(k)}(x)$ is 2. We use Adam (Kingma and Ba 2014) optimizer for updating the model. To demonstrate the efficacy of the proposed model, we conduct experiments on two new synthetic datasets (V1Deform and V1FlyingObjects) and two public datasets (MPI-Sintel and MUG Facial Expression). The motivation to generate the synthetic datasets is that we find existing datasets such as Flying Chairs (Dosovitskiy et al. 2015), FlyingThings3D (Mayer et al. 2016), and KITTI flow (Geiger, Lenz, and Urtasun 2012) contain image pairs with fairly large motions, which are unlikely consecutive frames perceived by V1. Thus we generate two synthetic datasets: V1Deform and V1FlyingObjects, which contains image pairs with only small local displacements and therefore better serve our purpose of studying motion perception in V1. See Fig. 4 for some examples from the synthetic datasets.

6.1 Datasets

In this subsection, we elaborate the generation process of the two new synthetic datasets, and introduce the public datasets we use in this work.

V1Deform. For this dataset, we consider random smooth deformations for natural images. Specifically, We obtain the training data by collecting static images for $(\mathbf{I}_t)$ and simulate the displacement field $(\delta(x))$. The simulated displacement field is then used to transform $\mathbf{I}_t$ to obtain $\mathbf{I}_{t+1}$. We retrieve natural images as $\mathbf{I}_t$ from MIT places365 dataset (Zhou et al. 2016). The images are scaled to 128×128. We sub-sample the pixels of images into a $m \times m$ grid ($m = 4$ in the experiments), and randomly generate displacements on the grid points, which serve as the control points for deformation. Then $\delta(x)$ for $x \in D$ can be obtained by spline interpolation of the displacements on the control points. We get $\mathbf{I}_{t+1}$ by warping $\mathbf{I}_t$ using $\delta(x)$ (Jaderberg et al. 2015). When generating a displacement $\delta = (\delta_1, \delta_2)$, both δ_1 and δ_2 are randomly sampled from a range of $[-6, +6]$. We synthesize $20,000$ pairs for training and $3,000$ pairs for testing.

V1FlyingObjects. For this dataset, we consider separating the displacement field into motions of the background and foreground, to jointly simulate the self-motion of the agent and the motion of the objects in the natural 3D scenes. To this end, we apply affine transformations to background images collected from MIT places365 (Zhou et al. 2016) and foreground objects from a public 2D object dataset COIL-100 (Nene et al. 1996). The background images are scaled to 128×128, and the foreground images are randomly rescaled. To generate motion, we randomly sample affine parameters of translation, rotation, and scaling for both the foreground and background images. The motions of the foreground objects are relative to the background images, which can be explained as the relative motion between the moving object and agent. We tune the distribution of the affine parameters to keep the range of the displacement fields within $[-6, +6]$, which is consistent with the V1Deform dataset. Together with the mask of the foreground object and the sampled transformation parameters, we render the image pair $(\mathbf{I}_t, \mathbf{I}_{t+1})$ and its displacement field $(\delta(x))$ for each pair of the background image and foreground image.

For the foreground objects, we obtain t he estimated masks from (tev 2006), resulting in 96 objects with 72 views per object available. We generate $14,411$ synthetic image pairs with their corresponding displacement fields and further split $12,411$ pairs for training and $2,000$ pairs for testing. Compared with previous optical flow dataset like Flying Chairs (Dosovitskiy et al. 2015) and scene flow dataset like FlyingThings3D (Mayer et al. 2016), the proposed V1FlyingObjects dataset has various foreground objects with more realistic texture and smoother displacement fields, which simulates more realistic environments.

We shall release the two synthetic datasets, which are suitable for studying local motions and perceptions. Besides, we also use two public datasets:

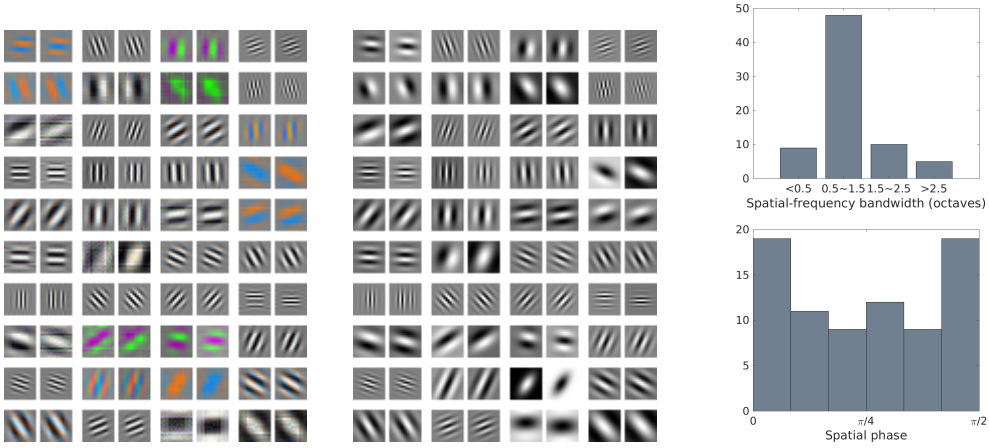
MPI-Sintel. MPI-Sintel (Butler et al. 2012; Wulff et al. 2012) is a public dataset designed for the evaluation of optical flow derived from rendered artificial scenes, with special attention to realistic image properties. Since MPI-Sintel is relatively small, which contains around a thousand image pairs, we use it only for testing the learned models in the inference of the displacement field. We use the final version of MPI-Sintel and resize each frame into size 128×128. We select frame pairs whose motions are within the range of $[-6, +6]$, resulting in 384 frame pairs in total.

MUG Facial Expression. MUG Facial Expression dataset (Aifanti, Papachristou, and Delopoulos 2010) records natural facial expression videos of 86 subjects sitting in front of one camera. This dataset has no ground truth of the displacement field, which we use for unsupervised learning. 200 videos with 30 frames are randomly selected for training, and anther 100 videos are sampled for testing.

6.2 Learned Gabor-like units with quadrature phase relationship

In this subsection, we show and analyze the learned units. The size of the filter is 16×16, with a sub-sampling rate of 8 pixels. Fig. 2(a) displays the learned units, i.e., rows of W, on V1Deform dataset. The units are learned with non-parametric $M(\delta)$, i.e., we learn a separate $M(\delta)$ for each displacement and $\delta(x)$ is discretized with an interval of 0.5. V1-like patterns emerge from the learned units. Moreover, within each sub-vector, the orientations and frequencies of learned units are similar, while the phases are different and approximately follow a quadrature relationship, consistent with the observation of biological V1 simple cells (Pollen and Ronner 1981; Emerson and Huang 1997). Similar patterns can be obtained by using a parametric version of $M(\delta)$. See Supplementary for more results of learned filters, including filters learned with different dimensions of sub-vectors using different datasets. It is worthwhile to mention that the dimension of sub-vectors is not constrained to be 2. V1-like patterns also merge when the dimension is 4 or 6.

To further analyze the spatial profiles of the learned units, we fit every unit by a two dimensional Gabor function (Jones and Palmer 1987): $h(x', y') = A \exp(-(x'/\sqrt{2}\sigma_{x'})^2 - (y'/\sqrt{2}\sigma_{y'})) \cos(2\pi f x' + \phi)$, where (x', y') is obtained by translating and rotating the original coordinate system (x_0, y_0): $x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta, y' = -(x -$



(a) Learned units (b) Fitted Gabors (c) Frequency and phase

Figure 2: Learned results on V1Deform dataset. (a) Learned units. Each block shows two learned units within the same sub-vector. (b) Fitted Gabor patterns. (c) Distributions of spatial-frequency bandwidth (in octaves) and spatial phase ϕ.

$x_0)\sin\theta + (y - y_0)\cos\theta$. The fitted Gabor patterns are shown in Fig. 2(b), with the average fitting r^2 equal to 0.96 (std $= 0.04$). The average spatial-frequency bandwidth is 1.13 octaves, with range of 0.12 to 4.67. Fig. 2(c) shows the distribution of the spatial-frequency bandwidth, where the majority falls within range of 0.5 to 2.5. The characteristics are reasonably similar to those of simple-cell receptive fields in the cat (Issa, Trepel, and Stryker 2000) (weighted mean 1.32 octaves, range of 0.5 to 2.5) and the macaque monkey (Foster et al. 1985) (median 1.4 octaves, range of 0.4 to 2.6). To analyze the distribution of the spatial phase ϕ, we follow the method in (Ringach 2002) to transform the parameter ϕ into an effective range of 0 to $\pi/2$, and plot the histogram of the transformed ϕ in Fig. 2(c). The strong bimodal with phases clustering near 0 and $\pi/2$ is consistent with those of the macaque monkey (Ringach 2002).

In the above experiment, we fix the size of the convolutional filters (16×16 pixels). A more reasonable model is to have different sizes of convolutional filters, with small size filters capturing high-frequency content and large size filters capturing low-frequency content. For fixed-size filters, they should only account for the image content within a frequency band. To this end, we smooth every image by two Gaussian smoothing kernels (kernel size 8, $\sigma = 1, 4$), and take the difference between the two smoothed images as the input image of the model. The effect of the two smoothing kernels is similar to a bandpass filter so that the input images are constrained within a certain range of frequencies. The learned filters on V1Deform dataset are shown in Fig 3(a). We also fit every unit by a two dimensional Gabor function, resulting in an average fitting $r^2 = 0.83$ (std $= 0.12$). Following the analysis of (Ringach 2002; Rehn and Sommer 2007), a scatter plot of $n_x = \sigma_x f$ versus $n_y = \sigma_y f$ is constructed in Fig. 3(b) based on the fitted parameters, where n_x and n_y represent the width and length of the Gabor envelopes measured in periods of the cosine waves. Compared to the sparse coding model (a.k.a. Sparsenet) (Olshausen and Field 1996, 1997), the units learned by our model have more similar structure to the receptive fields of simples cells of Macaque monkey.

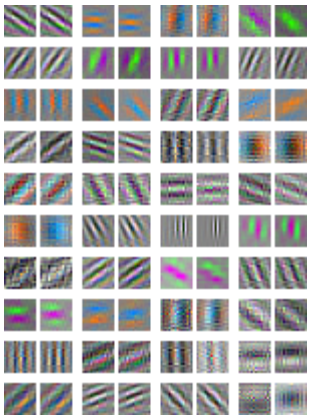
We also quantitatively compare the learned units within each sub-vector in Fig. 3(c). Within each sub-vector, the frequency f and orientation θ of the paired units tend to be the same. More importantly, most of the paired units differ in phase ϕ by approximately $\pi/2$, consistent with the quadrature phase relationship between adjacent simple cells (Pollen and Ronner 1981; Emerson and Huang 1997).

6.3 Inference of displacement field

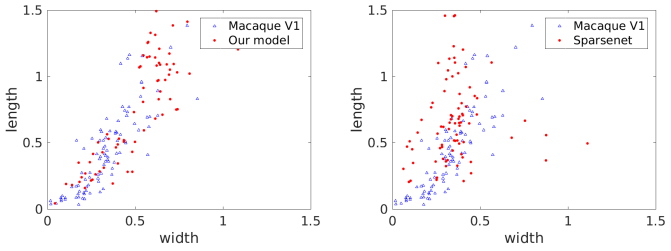
We then apply the learned representations to inferring the displacement field $(\delta(x))$ between pairs of frames $(\mathbf{I}_t, \mathbf{I}_{t+1})$. To get valid image patches for inference, we leave out those displacements at image border (8 pixels at each side).

We use non-parametric $M(\delta)$ and the local mixing motion model (Eq. (6)), where the local support $\mathcal{S}$ is in a range of $[-4, +4]$, and $\mathrm{d}x$ is taken with a sub-sampling rate of 2. After obtaining the inferred displacement field $(\delta(x))$ by the learned model, we also train a CNN model with ResNet blocks (He et al. 2016) to refine the inferred displacement field. In training this CNN, the input is the inferred displacement field, and the output is the ground truth displacement field, with least-squares regression loss. See Supplementary for the architecture details of the CNN. For biological interpretation, this refinement CNN is to approximate the processing in visual areas V2-V6 that integrates and refines the motion perception in V1 (Gazzaniga, Ivry, and Mangun 2002; Lyon and Kaas 2002; Moran and Desimone 1985; Born and Bradley 2005; Allman and Kass 1975). We learn the models from the training sets of V1Deform and V1FlyingObjects datasets respectively, and test on the corresponding test sets. We also test the model learned from V1FlyingObjects on MPI-Sintel Final, whose frames are resized to 128×128.

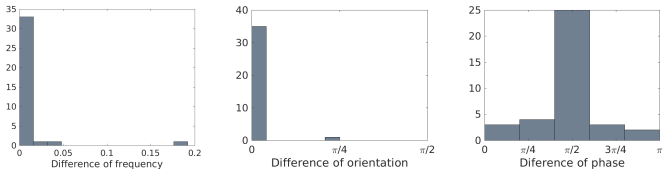
Table 1 summarizes the average endpoint error (AEE) of the inferred results. We compare with several baseline



(a) Learned units



(b) Gabor envelope shapes of the learned units



(c) Profile of paired units within each sub-vector

Figure 3: Learned results on band-pass image pairs from V1Deform. (a) Learned units. Each block shows two learned units within the same sub-vector. (b) Distribution of the Gabor envelope shapes in the width and length 2D-plane. (c) Difference of frequency f, orientation θ and phase ϕ of paired units within each sub-vector.

methods, including FlowNet 2.0 and its variants (Dosovitskiy et al. 2015; Ilg et al. 2017). For baseline methods, we test the performance using two models: one retrained on our datasets ('trained') and the released model by the original authors which are pre-trained on large-scale datasets ('pre-trained'). Note that for MPI-Sintel, a pre-trained baseline model gives better performance compared to the one trained on V1FlyingObjects, probably because these methods train deep and complicated neural networks with large amount of parameters to predict optical flows in supervised manners, which may require large scale data to fit and transfer to different domains. On the other hand, our model can be treated as a simple one-layer auto-encoder network, accompanied by weight matrices representing motions. As shown in Table 1, our model has about 88 times fewer parameters than FlowNet 2.0 and 21 times fewer parameters than the light FlowNet2-C model. We achieve competitive performance compared to these baseline methods. Fig. 4 displays several examples of the inferred displacement field. Inferred results from the FlowNet 2.0 models are shown as a qualitative comparison. For each dataset, we show the result of FlowNet 2.0 model with lower AEE between the pre-trained and trained ones.

6.4 Unsupervised learning

We further perform unsupervised learning of the proposed model, i.e., without knowing the displacement field of training pairs. For unsupervised learning, we scale the images to size 64×64. The size of the filters is 8×8, with a sub-sampling rate of 4 pixels. Displacements at the image border (4 pixels at each side) are left out. We train the model on MUG Facial Expression and V1FlyingObjects datasets. Fig. 5 shows some examples of inferred displacement fields on the testing set of MUG Facial Expression. The inference results are reasonable, which capture the motions around eyes, eyebrows, chin, or mouth. For the model trained on V1FlyingObjects, we test on the testing set of V1FlyingObjects and MPI-Sintel. Table 2 summarizes the quantitative results. We include comparisons with several baseline methods for unsupervised optical flow estimation: Unsup (Jason, Harley, and Derpanis 2016) and UnFlow (Meister, Hur, and Roth 2018) and its variants, which are also trained on V1FlyingObjects. The proposed model achieves better performance compared to baseline methods. See Supplementary for qualitative comparisons and more inference results.

6.5 Multi-step frame animation

The learned model is also capable of multi-step frame animation. Specifically, given the starting frame $\mathbf{I}_0(x)$ and a sequence of displacement fields $\{\delta_1(x), ..., \delta_T(x), \forall x\}$, we can animate the subsequent multiple frames $\{\mathbf{I}_1(x), ..., \mathbf{I}_T(x)\}$ using the learned model. We use the model with local mixing. We introduce a re-encoding process when performing multi-step animation. At time t, after we get the next animated frame $\mathbf{I}_{t+1}$, we take it as the observed frame at time $t+1$, and re-encode it to obtain the latent vector v_{t+1} at time $t+1$. Fig. 6 displays two examples of animation for 6 steps, learned with non-parametric version of M on V1Deform and V1FlyingObjects. The animated frames match the ground truth frames well. See Supplementary for more results.

6.6 Frame interpolation

Inspired by the animation and inference results, we show that our model can also perform frame interpolation, by combining the animation and inference together. Specifically, given a pair of starting frame $\mathbf{I}_0$ and end frame $\mathbf{I}_T$, we want to derive a sequence of frames $(\mathbf{I}_0, \mathbf{I}_1, ..., \mathbf{I}_{T-1}, \mathbf{I}_T)$ that changes smoothly. Let $v_0(x) = W\mathbf{I}_0[x]$ and $v_T(x) = W\mathbf{I}_T[x]$ for each $x \in D$. At time step $t+1$, like the inference, we can infer displacement field $\delta_{t+1}(x)$ by steepest

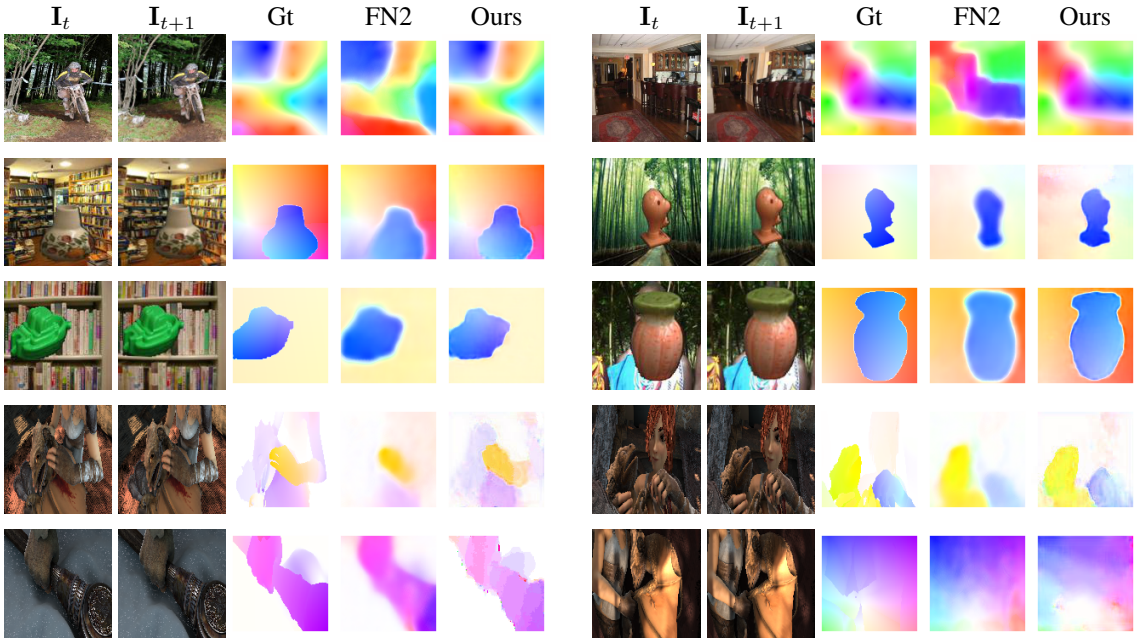


Figure 4: Examples of inferred displacement field on V1Deform, V1FlyingObjects and MPI-Sintel. For each block, from left to right are $\mathbf{I}_t$, $\mathbf{I}_{t+1}$, ground truth displacement field and inferred displacement field by FlowNet 2.0 model and our learned model respectively. For each dataset, we show the result of FlowNet 2.0 model with lower AEE between the pre-trained and trained ones. The displacement fields are color coded (Liu, Yuen, and Torralba 2010). See Supplementary for the color code.

Table 1: Average endpoint error of the inferred displacement and number of parameters. Abbreviation FN2 refers to FlowNet 2.0. 'Ours-ref' indicates that the results are post-processed by the refinement CNN.

	V1Deform		V1FlyingObjects		MPI-Sintel		# params (M)
	pre-trained	trained	pre-trained	trained	pre-trained	trained	
FN2-C	1.324	1.130	0.852	1.034	0.363	0.524	39.18
FN2-S	1.316	0.213	0.865	0.261	0.410	0.422	38.68
FN2-CS	0.713	0.264	0.362	0.243	0.266	0.346	77.87
FN2-CSS	0.629	0.301	0.299	0.303	0.234	0.450	116.57
FN2	0.686	0.205	0.285	0.265	0.146	0.278	162.52
Ours	-	0.258	-	0.442	-	0.337	1.82
Ours-ref	-	**0.156**	-	**0.202**	-	**0.140**	1.84

descent:

$$\hat{v}_{t+1}(x,\delta) = \sum\nolimits_{\mathrm{d}x\in\mathcal{S}} M(\delta,\mathrm{d}x)v_t(x+\mathrm{d}x), \tag{15}$$

$$\delta_{t+1}(x) = \arg\min_{\delta\in\Delta} \sum_{k=1}^{K} \left\| v_T^{(k)} - \hat{v}_{t+1}^{(k)}(x,\delta) \right\|^2. \tag{16}$$

Like the animation, we get the animated frame $\mathbf{I}_{t+1}$ by decoding $\hat{v}_{t+1}(x,\delta_{t+1}(x))$, and then re-encode it to obtain the latent vector $v_{t+1}(x)$.

The algorithm stops when $\mathbf{I}_t$ is close enough to $\mathbf{I}_T$ (mean pixel error < 10). Fig. 7 shows four examples, learned with non-parametric M on V1Deform and V1FlyingObjects. For 96.0% of the testing pairs, the algorithm can accomplish the frame interpolation within 10 steps. With this algorithm, we are also able to infer displacements larger than the acceptable range of δ by accumulating the displacements along the interpolation steps. See Supplementary for more results.

6.7 Ablation study

We perform ablation studies to analyze the effect of two components of the proposed model: (1) dimensionality of sub-vectors and (2) sub-sampling rate. Besides comparing the average endpoint error (AEE) of motion inference, we also test if the learned model can make accurate multi-step animation of image frames given the sequence of displacement fields. Per pixel mean squared error (MSE) of the predicted next five image frames is reported. Table 3 summarizes the results learned from V1Deform dataset. The dimensionality of sub-vectors controls the complexity of the motion matrices, which is set to a minimum of 2 in the experiments. As the dimensionality of sub-vectors increases, the error rates of the two tasks decrease first and then increase. On the other hand, sub-sampling rate can be changed to make the adjacent image patches connect with each other

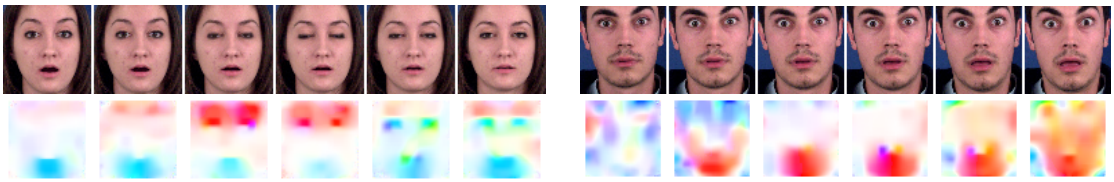


Figure 5: Examples of inferred displacement fields by unsupervised learning on MUG Facial Expression dataset. Within each block, the top row shows the observed image frames, while the bottom row shows the inferred color-coded displacement fields (Liu, Yuen, and Torralba 2010). See Supplementary for the color code.

Table 2: Average endpoint error of the inferred displacement in unsupervised learning.

	Unsup	UnFlow-C	UnFlow-CS	UnFlow-CSS	Ours
V1FlyingObjects (train)	0.379	0.336	0.374	0.347	**0.245**
V1FlyingObjects (test)	0.811	0.399	0.394	0.453	**0.316**
MPI-Sintel	0.440	0.198	0.248	0.202	**0.101**

Table 3: Ablation study measured by average endpoint error (AEE) of motion inference and mean squared error (MSE) of multi-step animation, learned from V1Deform dataset.

	Sub-vector dimension				
	2	4	6	8	12
AEE	0.258	0.246	0.238	0.241	0.0.246
MSE	8.122	7.986	7.125	7. 586	8.017

	Sub-sampling rate		
	4	8	16
AEE	0.383	0.258	0.312
MSE	11.139	8.122	10.293

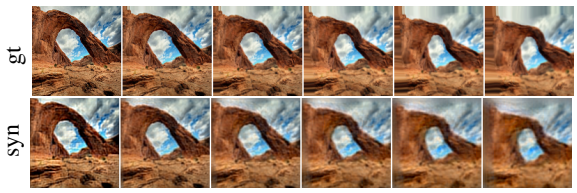


Figure 6: Example of multi-step animation. For each block, the first row shows the ground truth frame sequences, while the second row shows the animated frame sequences.

more loosely or tightly. As shown in Table 3, a sub-sampling rate of 8, which is half of the filter size, leads to the optimal performance.

7 Conclusion

This paper proposes a simple representational model that couples vector representations of local image contents and matrix representations of local motions, so that the vector representations are equivariant. Unlike existing models for V1 that focus on statistical properties of natural images or videos, our model serves a direct purpose of perception of local motions. Our model learns Gabor-like units with quadrature phases. We also give biological interpretations of the learned model and connect it to the spatiotemporal energy model. It is our hope that our model adds to our understanding of motion perception in V1.

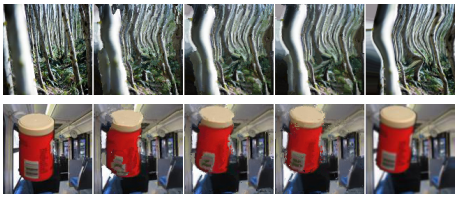


Figure 7: Examples of frame interpolation, learned with non-parametric M. For each block, the first frame and last frame are given, while the frames between them are interpolated frames.

Our motion model can be integrated with the sparse coding model. For sparse coding, we can keep the top-down decoder of the tight frame auto-encoder for each image frame, and impose sparsity on the number of sub-vectors that are active. For motion, we then assume that each sub-vectors is transformed by a sub-matrix that represents the local displacement.

This paper assumes linear decoder for image frames and matrix representation of local motion. We can generalize the linear decoder to a neural network and generalize the matrix representation to non-linear transformation modeled by non-linear recurrent network.

In our future work, we shall study the inference of ego-motion, object motions and 3D depth information by generalizing our model based on equivariant vector representations and their transformations. We shall also apply our model to stereo in binocular vision.

Acknowledgement

The work is supported by NSF DMS-2015577, ONR MURI project N00014-16-1-2007, DARPA XAI project N66001-17-2-4029, and XSEDE grant ASC170063. We thank Prof. Tai Sing Lee for sharing his knowledge and insights on V1.

References

2006. GroundTruth100-for-COIL : 100 labelled images for object recognition tests.

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

Aifanti, N.; Papachristou, C.; and Delopoulos, A. 2010. The MUG facial expression database. In *11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10*, 1–4. IEEE.

Allman, J.; and Kass, J. 1975. The dorsomedial cortical visual area: a third tier area in the occipital lobe of the owl monkey (Aotus trivirgatus). *Brain research*, 100(3): 473–487.

Bell, A. J.; and Sejnowski, T. J. 1997. The independent components of natural scenes are edge filters. *Vision research*, 37(23): 3327–3338.

Born, R. T.; and Bradley, D. C. 2005. Structure and function of visual area MT. *Annu. Rev. Neurosci.*, 28: 157–189.

Butler, D. J.; Wulff, J.; Stanley, G. B.; and Black, M. J. 2012. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), ed., *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, 611–625. Springer-Verlag.

Daugman, J. G. 1985. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7): 1160–1169.

Dosovitskiy, A.; Fischer, P.; Ilg, E.; Häusser, P.; Hazırbaş, C.; Golkov, V.; v.d. Smagt, P.; Cremers, D.; and Brox, T. 2015. FlowNet: Learning Optical Flow with Convolutional Networks. In *IEEE International Conference on Computer Vision (ICCV)*.

Emerson, R. C.; and Huang, M. C. 1997. Quadrature subunits in directionally selective simple cells: counterphase and drifting grating responses. *Visual neuroscience*, 14(2): 373–385.

Foster, K.; Gaska, J. P.; Nagler, M.; and Pollen, D. 1985. Spatial and temporal frequency selectivity of neurones in visual cortical areas V1 and V2 of the macaque monkey. *The Journal of physiology*, 365(1): 331–363.

Gao, R.; Xie, J.; Wei, X.-X.; Zhu, S.-C.; and Wu, Y. N. 2021. On path integration of grid cells: group representation and isotropic scaling. In *Neural Information Processing Systems*.

Gao, R.; Xie, J.; Zhu, S.-C.; and Wu, Y. N. 2019. Learning grid cells as vector representation of self-position coupled with matrix representation of self-motion. In *International Conference on Learning Representations*.

Gazzaniga, M. S.; Ivry, R.; and Mangun, G. 2002. Cognitive Neuroscience, New York: W. W.

Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *European conference on computer vision*, 630–645. Springer.

Hubel, D. H.; and Wiesel, T. N. 1959. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3): 574–591.

Hyvärinen, A.; Hurri, J.; and Väyrynen, J. 2003. Bubbles: a unifying framework for low-level statistical properties of natural image sequences. *JOSA A*, 20(7): 1237–1252.

Hyvärinen, A.; Karhunen, J.; and Oja, E. 2004. *Independent component analysis*, volume 46. John Wiley & Sons.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Issa, N. P.; Trepel, C.; and Stryker, M. P. 2000. Spatial frequency maps in cat visual cortex. *Journal of Neuroscience*, 20(22): 8504–8514.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In *Advances in neural information processing systems*, 2017–2025.

Jason, J. Y.; Harley, A. W.; and Derpanis, K. G. 2016. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *European Conference on Computer Vision*, 3–10. Springer.

Jayaraman, D.; and Grauman, K. 2015. Learning image representations tied to ego-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, 1413–1421.

Jones, J. P.; and Palmer, L. A. 1987. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of neurophysiology*, 58(6): 1233–1258.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lewicki, M. S.; and Olshausen, B. A. 1999. Probabilistic framework for the adaptation and comparison of image codes. *JOSA A*, 16(7): 1587–1601.

Liu, C.; Yuen, J.; and Torralba, A. 2010. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5): 978–994.

Lyon, D. C.; and Kaas, J. H. 2002. Evidence for a modified V3 with dorsal and ventral halves in macaque monkeys. *Neuron*, 33(3): 453–461.

Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; and Brox, T. 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4040–4048.

Meister, S.; Hur, J.; and Roth, S. 2018. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Moran, J.; and Desimone, R. 1985. Selective attention gates visual processing in the extrastriate cortex. *Science*, 229(4715): 782–784.

Nene, S. A.; Nayar, S. K.; Murase, H.; et al. 1996. Columbia object image library (coil-20).

Olshausen, B. A. 2003. Learning sparse, overcomplete representations of time-varying natural images. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 1, I–41. IEEE.

Olshausen, B. A.; and Field, D. J. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583): 607.

Olshausen, B. A.; and Field, D. J. 1997. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23): 3311–3325.

Olshausen, B. A.; and Field, D. J. 2005. How close are we to understanding V1? *Neural computation*, 17(8): 1665–1699.

Paccanaro, A.; and Hinton, G. E. 2001. Learning distributed representations of concepts using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering*, 13(2): 232–244.

Pachitariu, M.; and Sahani, M. 2017. Visual motion computation in recurrent neural networks. *bioRxiv*, 099101.

Pollen, D. A.; and Ronner, S. F. 1981. Phase relationships between adjacent simple cells in the visual cortex. *Science*, 212(4501): 1409–1411.

Rehn, M.; and Sommer, F. T. 2007. A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields. *Journal of computational neuroscience*, 22(2): 135–146.

Ringach, D. L. 2002. Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *Journal of neurophysiology*, 88(1): 455–463.

Singer, Y.; Teramoto, Y.; Willmore, B. D.; Schnupp, J. W.; King, A. J.; and Harper, N. S. 2018. Sensory cortex is optimized for prediction of future input. *Elife*, 7: e31557.

van Hateren, J. H.; and Ruderman, D. L. 1998. Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proceedings of the Royal Society of London B: Biological Sciences*, 265(1412): 2315–2320.

Wiskott, L.; and Sejnowski, T. J. 2002. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4): 715–770.

Wulff, J.; Butler, D. J.; Stanley, G. B.; and Black, M. J. 2012. Lessons and insights from creating a synthetic optical flow benchmark. In A. Fusiello et al. (Eds.), ed., *ECCV Workshop on Unsolved Problems in Optical Flow and Stereo Estimation*, Part II, LNCS 7584, 168–177. Springer-Verlag.

Zhou, B.; Khosla, A.; Lapedriza, A.; Torralba, A.; and Oliva, A. 2016. Places: An image database for deep scene understanding. *arXiv preprint arXiv:1610.02055*.

Zhu, Y.; Gao, R.; Huang, S.; Zhu, S.-C.; and Wu, Y. N. 2021. Learning Neural Representation of Camera Pose with Matrix Representation of Pose Shift via View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9959–9968.