School of Electronic Engineering

GPGPU-accelerated Specialized Neural Models of Visual Phenomena

Project Portfolio

Niall Cullinane

Student Number: 18214952

August, 2020

MEng in Electronic and Computer Engineering

Supervised by Dr Noel Murphy

ACKNOWLEDGEMENT

I wish to thank Dr Noel Murphy for suggesting the topic and for his guidance and insightful input though out this project.

I would also like to thank Dr Julia Dietlmeier for her feedback during my presentation, my technical officer Robert Clare, Dr Conor McArdle for his help and advice with the GPGPU server and to Liam Meany for organising a remote connection to the server.

I am very grateful of the supports provided to me by the University's Disability Service during my time at DCU, in particular by Susan Madigan.

Finally, I am very appreciative of my family's patience and support throughout my return to studies.

NIALL CULLINANE

Dublin City University

August, 2020

CONTENTS

GPGPU-accelerated specialized neural models of visual phenomena

Niall Cullinane

***Abstract*—The human visual system is inherently more capable of processing occluded object when compared to computer vision techniques. It uses illusory contours to complete missing boundaries. LAMINART, a model developed by computational neuroscientist, mimics the formation of these contours. In this research, a computationally efficient implementation of LAMINART was made with the use of a GPGPU. The implementation was found to produce illusory contours. The model was found to have issues with stability and robustness.**

***Index Terms*—Bio-inspired computing, computational neuroscience, computer vision, biological neural networks, feedback, parallel processing.**

I. INTRODUCTION

OCCLUDED objects make a variety of tasks very challenging for computer vision. Areas such as image segmentation and figure-ground separation have numerous applications, many of which involve occlusion. Biological visual systems are much more capable of handling occlusion. The brain seems to be able to seamlessly fill in gaps in its visual field.

Neuroscientists have developed theories as to how the visual system process occluded objects with the use of visual illusions. Illusions involving illusory contours, such as the Kanizsa square shown in Fig. 1(a), take advantage of the same mechanisms that are fundamental to how the brain processes occluded objects. Some of these illusions have played an important role in the development of computational models of biological visual systems. They have also been used to test these models. It has been suggested that some of these computational models may be a valuable source of inspiration for the computer vision engineer [1].

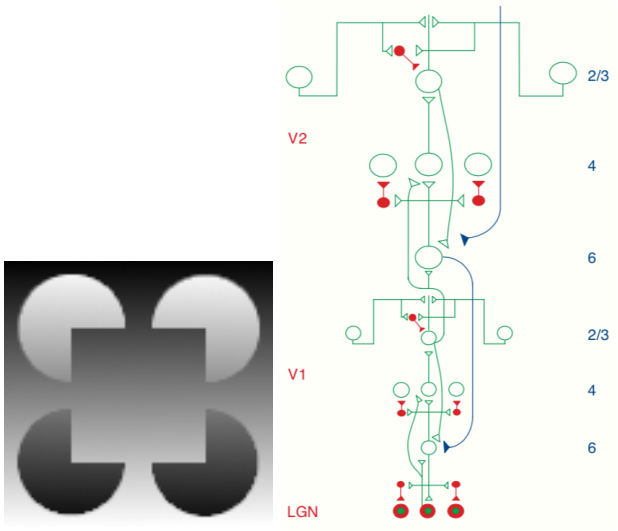
The LAMINART model [2] was developed, in part, to account for the occurrence of illusory contours in the mammalian visual system. The model uses several feedback pathways along with convolution and non-linear operators to mimic these illusory contours. There appear to be few implementations of the model outside the domain of computational neuroscience, although, within that domain, it has been found to work successfully [3], [4].

Several other computer vision techniques have been developed to produce illusory contours [5]. It would appear that all of these approaches are either computationaly demainding or not robust to noise.

The complex structure of LAMINART , in particular its feedback pathways and use of large convolution, may have previously made a model of this type infeasible for computer vision applications. The model is highly parallelisable however, making it suited for implementation on a general purpose graphics processing unit (GPGPU).

The goal of this research was to investigate if a computationally efficient and robust implementation of the LAMINART model could be developed to allow its future use in computer vision applications feasible.

N. Cullinane is with the School of Electronic Engineering, Dublin City University, Glasnevin, Dublin, Ireland.



(a) (b)

Figure 1. A Kanizsa square, shown in (a), is a visual illusion which tricks the brain into seeing four circles being occluded by a square. The brain generates an illusory contour between two boundaries of high contrast. (b) shows the feedforward, feedback and horizontal interaction of the LAMINART model. Inhibitory interactions are indicated in red.[6]

II. PRIOR WORK

A. LAMINART *model*

LAMINART [7] models the lower levels of a mammalian visual system. It was developed to mimic how those systems use illusory contours to undertake boundary completion. Cells are ordered in layers of arrays and are excited and inhibited by several feedforward, feedback and horizontal interactions, as shown in Fig. 1(b). These interactions work as on-center off-surround networks. The lower layers perform an oriented edge detection as an on-centre off-surround network excites areas of high contrast. Pairs of convolutions in higher layers work in tandem to inhibit the signal while it is fed back through the network at several points. Non-linear half-wave rectifier and threshold functions are used throughout the network. These attenuate low or negative inputs to a cell, balancing the feedforward and feedback signals. Where a feedback signal

is strong enough to meet this threshold the cell can output a signal without a direct stimulation at that point in the visual field. The output is an illusory signal. Feedback would be strong enough here due to neighbouring cells exciting that cell through horizontal connections.

The model was described by a system of ordinary differential equations (ODEs). Parameters were used to match experimental data on biological systems. Several of the kernels used for convolution were learned from data, using results from [8]. Values for these kernels were only provided graphically. The kernels only correspond to a model with two orientations.

B. Prior implementations of LAMINART

Few implementations of LAMINART appear to have been made outside of the domain of computational neuroscience. An earlier model called FACADE, a precursor to LAMINART , was evaluated and adapted for use with real-world images [9]. The model was found not to be very robust to noise. An imbalance of excitation and inhibition at the higher layers was also reported. Alterations to the shape of kernels improved robustness.

LAMINART has been used recently by neuroscientists investigating visual crowding [4], [3]. The model was implemented using a spiking neural network. The number of the model's orientations was increased to eight by [3] but the approach used to achieve this was unclear.

III. Technical description

The aim of this research was to investigate if part of LAMINART could be implemented efficiently to produce illusory contours. To make this investigation feasible, only the lower layers of the model – corresponding to the retina, the lateral geniculate nucleus (LGN) and primary visual cortex (V1) – were implemented. The higher layers of the model, corresponding to secondary visual cortex (V2), and the feedback from V2 to V1 were not included. The implementation was also restricted to take only monochromatic input images. An efficient implementation was designed by

- removing redundant operations in the equations,
- choosing efficent methods to implement these equations,
- choosing an appropriate method to solve the system,
- reducing allocations during solving and
- parallelisation.

A. Model's equations

The equations used by [7] to describe LAMINART were chosen. They consist of several first-order, time-dependent differential equations, each operating on an array. These arrays were connected with various convolution and non-linear operators as outlined in Fig. 2. The lower levels of arrays, v^+ and v^-, had the same dimensions as the input image, $i \times j$. A series of convolutions between $v^\pm$ and C resulted in a third dimension of orientations giving the higher arrays, C, x, y, m, z and s dimensions of $i \times j \times k$ where k, the number of orientations was two. The feedback from x to $v^\pm$ had its orientations summed before going through an on-centre off-surround network.

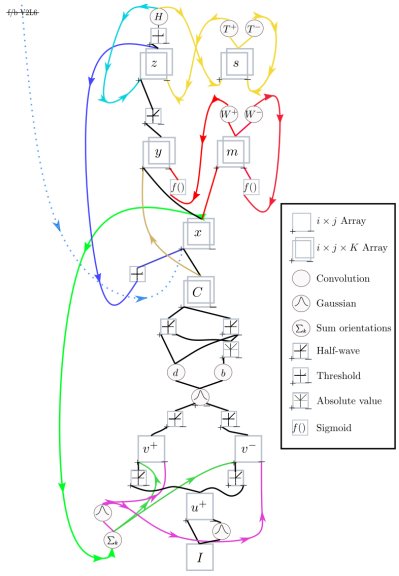


Figure 2. The version of LAMINART implemented. Feedback from V2, indicated with the dotted line, was omitted. The input at the bottom goes through on-centre off-surround networks. The red circuit inhibits y. The yellow circuit inhibits z. The green line shows the feedback from x to $v^\pm$. The light and dark blue lines show the other two feedback pathways.

Three sets of these equations were rearranged to reduce the number of convolutions to be performed.

- *$v^\pm$ to higher layers:* The equations describing the input from $v^\pm$ to the higher cells were rewritten, reducing the number of convolutions from four per orientation to two per orientation plus one.
- *Feedback from x to $v^\pm$:* The orientations were summed before going through an on-centre off-surround arrangement. The summing operation was moved so as to be performed only once for both the on-centre and off-surround feedback.
- *z and s:* Two pathways which feedback z, directly and through s, share a common convolution, H. This convolution step was moved so to be performed once for both pathways.

Details of the rearrangements made to the equations were outlined in [10].

The implementation of the model in [7] relied on three sets of self-organised kernels from [8]. These kernels were only learned for two orientations. Three of these kernels W^+, W^- and H were only presented graphically. In order to replicate these kernels, several oriented Gaussian filters were combined to match to the plots given. Plots of the kernels are shown in Fig. 3. The third self-organised kernel, T, was reported numerically. This was a single pixel, depth-wise kernel but was asymmetrical across the two orientations. It was decided to average the values from the two orientations. This would allow more efficient convolution. All other parameters used in the implementation matched [11].

B. Solving the system

The equations form a system of coupled, first-order, time dependent, non-linear, ODEs. The size and complexity of the

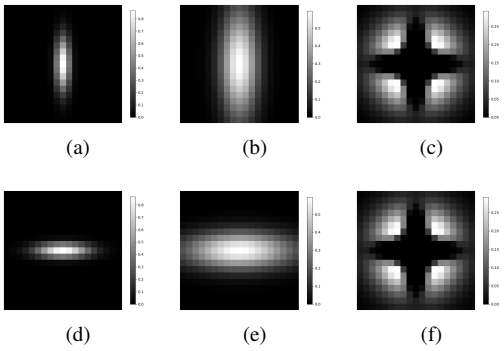


Figure 3. Plots of the kernels used for orientation $k = 1$ (top line) and $k = 2$ (bottom line). The kernels are H (a,d); W^+ for the same orientations (b,e) and W^+ for the opposite orientations (c,f). W^- has a similar shape but is stronger with a peak value ≈ 4.

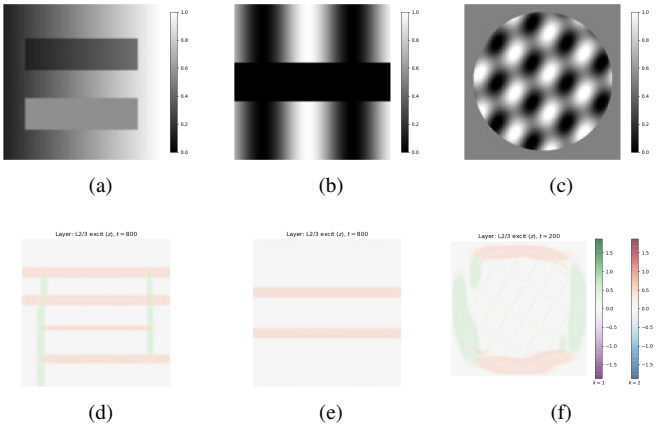


Figure 4. A selection of illusory test images (a), (b) and (c) [22], with activation of z for each image at $t = 800$ below. Illusory contours formed with the three images. Orientation $k = 1$ is shown in green, $k = 2$ is shown in red.

system means that a solution cannot be calculated analytically from its initial condition. Exact digital simulation methods have been popular in the computational neuroscience community and LAMINART has been implemented with these methods by [3], [4]. The focus of those implementations was towards an accurate model of the visual system. The focus of this paper is to develop an efficient design. A numerical approach was decided to be more suitable. ODE problems may be divided into two groups – non-stiff and stiff problems. The stiffness of a problem relates to the steepness of the curve and the step-size needed to solve it. Numerical methods also fall into two groups – explicit and implicit. Explicit methods are simpler and are faster for non-stiff problems. They can be very slow at solving stiff problems, and may fail completely. Implicit methods are more complex. They usually require a Jacobian to be calculated. They are more suited to stiff problems. The stiffness of a problem may change during solving. Adaptive methods have been developed to automatically switch to the most appropriate method during solving. Three explicit methods: *Tsit5* [12], *BS3* [13] and *Vern7* [14] along with two adaptive: implicit methods – *AutoTsit5 with Rosenbrock23* [12],[15] and *LSODA*[16] were tested. This selection was made as these methods were suited to this type of system. The model implemented in *Julia* [17] using the package *DifferentialEquations.jl* [18]. Convolutions were done using *NNlib.jl*[19]. Zero padding was used to avoid bordering artifacts. This was particularly important as the convolutions were performed at each step during solving. The functions representing the model's equations were designed to mutate existing variables rather then return values. This was to reduce allocations during solving in order to increase performance.

C. Parallelisation

The array structure of the system meant that it was highly parallelisable. An implementation could be run on a GPGPU with the use of Compute Unified Device Architecture (CUDA)[20]. The Julia package *CuArrays.jl* [21] was used to port over the model. This package gives high-level access to CUDA to perform operations on arrays on a GPGPU. This allowed for a cleaner porting of the model to a GPGPU compared to writing CUDA kernels directly. The package can automate several optimisation steps if the implementation is carefully designed. Both *DifferentialEquations.jl* and *NNlib.jl* integrate well with CuArrays.jl. A large selection of solvers available to *DifferentialEquations.jl* are able to be parallelised. *NNlib.jl's* `conv` function runs on a GPGPU automatically with *CuArrays.jl*. Where broadcast functions are used with a scalar-array and array-array operations, *CuArrays.jl* performs the operations in parallel. The package also tries to use efficient CUDA libraries such as CUBLAS where possible. A different approach attempted was to perform only the convolutions on the GPGPU using an image processing library, *JuliaImages.jl*. A bottleneck occurred as the array was written back to the central processing unit (CPU) after each convolution. *CuArrays.jl* avoided this issue as the array remained in the GPGPU's memory.

The model was implemented on a NVidia Tesla K40C using *Float32*. This data type allowed the highest number of operations per cycle on that GPGPU [20].

IV. TESTING AND RESULTS

A. Visual illusions

Several visual illusions were used to test the implementation. Test images were either generated or chosen from [22]. The output of z for a selection of illusions is shown in Fig. 4. A Kanizsa square is shown in Fig. 5 with the output from each of the model's layers after stabilisation was reached. The output from z from the same illusion is shown in Fig. 6(a)-Fig. 6(f) as it changed over time during solving. The activation of the highest value pixel in z at the end of solving is shown for all arrays with respect to time in Fig. 6(g).

B. Noise and real-world images

The implementation was tested with a real-world image, with increasing levels of Gaussian noise, as shown in Fig. 7. The model failed to solve with tests where $\sigma < 0.5$.

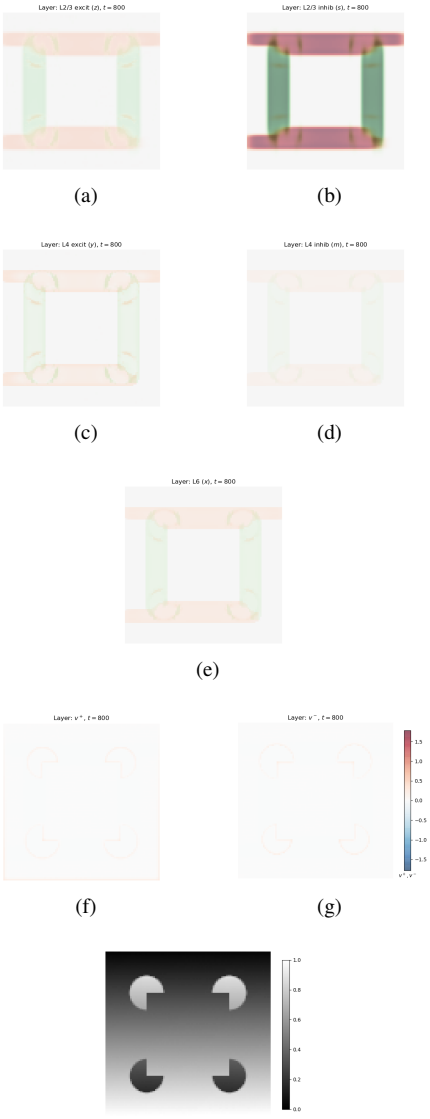


Figure 5. A Kanizsa square with opposing contrast. The input image is shown in (h). The activation is shown at $t = 800$ for (f):v^+, (g):v^-, (e):x, (c):y, (d):m, (a):z, (b):s. Orientation for the higher layers is shown in green for $k = 1$ and red for $k = 2$.

C. Parameter variation

Various combinations of parameters were investigated. These results are included in [23].

D. Benchmarking

Several solving methods were tested and benchmarked. The two implicit solvers tested, LSODA and *Rosenbrock23*, failed to solve before causing the system to run out of memory. Three explicit solvers were benchmarked, both on a CPU and GPGPU. The results are shown in Fig. 8(a). *BS3* was found to be the fastest. However, the solution lost stability as shown in Fig. 8(c). *Tsit5*, the second fastest, was chosen and maintained stability with the illusory tests.

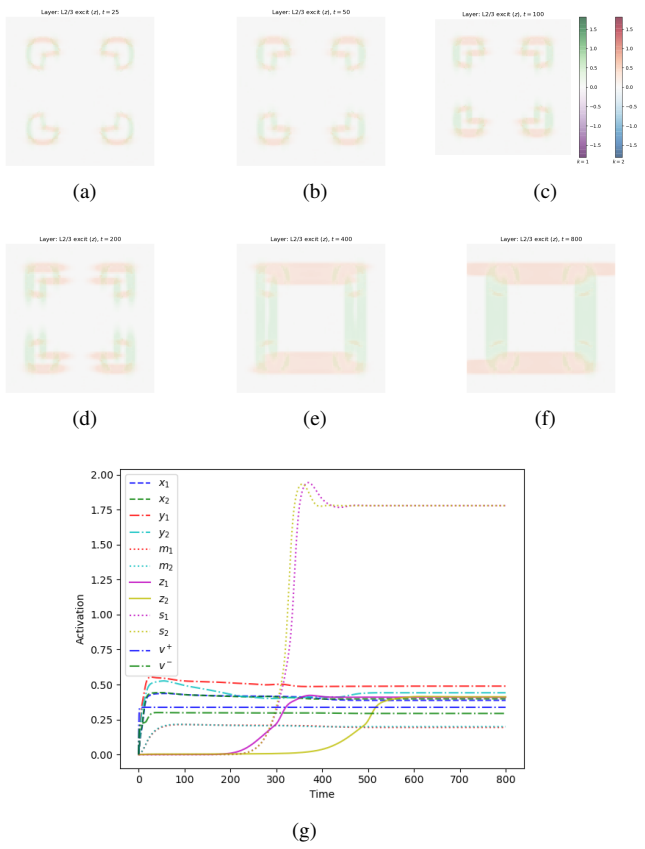


Figure 6. The activation of z the Kanizsa square at different steps during solving. The growth and spreading out of illusory contours is seen developing. (a):$t = 25$, (b):$t = 50$, (c):$t = 100$, (d):$t = 200$, (e):$t = 400$, (f):$t = 800$. The activation for each level is plotted over time in (g). The pixel with the highest value in z at $t = 800$ was used for all arrays.

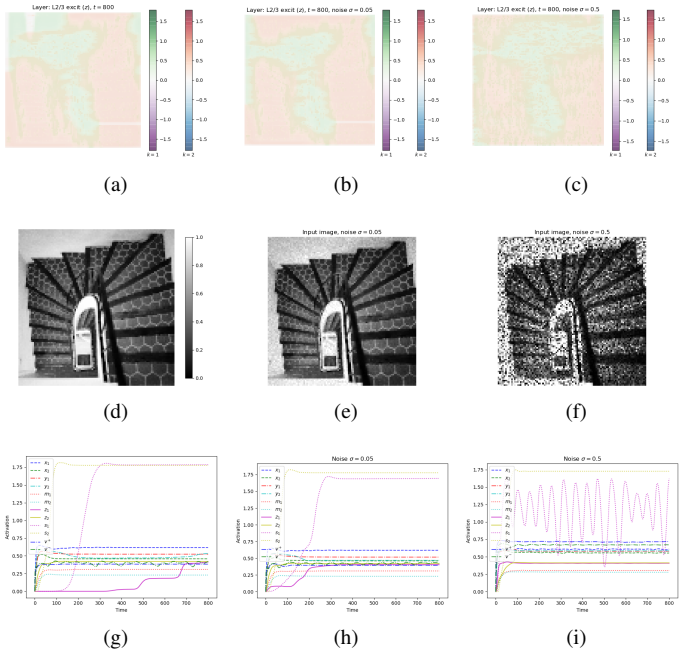
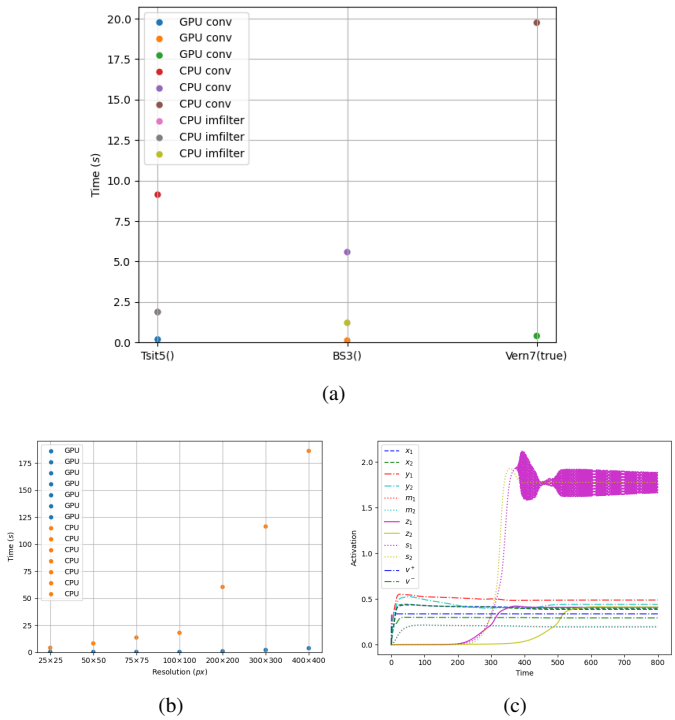


Figure 7. A real=world image with no additional noise on the left, Gaussian noise with $\sigma = 0.05$ in the middle and $\sigma = 0.5$ on the left. The system did not solve for tests where > 0.5. The activation of m appeared to become periodic (i).



(a)

(b) (c)

Figure 8. (a): Benchmarks for three explicit solvers for both GPU and CPU. This test was done for timespan $= [0, 800]$. (b): Benchmarks for the GPU and cpu with increasing sizes of input images. This test was done for timespan $= [0, 10]$. (c): Activation plot over time for *BS3*. The activation of s appeared to become unstable at $t \approx 100$.

Implementations on a CPU and GPGPU were tested with various sized input images. The results are shown in Fig. 8(b).

V. Analysis

The implementation was successful in producing illusory contours from a variety of visual illusions. It was able to do this for illusions involving opposite contrast polarities as seen in Fig. 4.

The contours spread out more than expected. This may have been due to the absence of feedback from V2. This may also have been due to the estimations of the W kernels which are needed to inhibit the circuit.

Variations in values for the kernels and other parameters either restricted the formation of illusory contours where feedback was not great enough, or caused the system to become unstable. When all feedback terms were either doubled, halved or removed, illusory contours were restricted. When H alone was doubled, illusory contours formed very quickly, before $t = 50$. The contours then spilled out across the whole array. When H alone was halved, feedback was not enough to form illusions. The feedback pathways, with inhibition from m and s, are complex. The self organised kernels W^+, W^-, T^+, T^- and H along with the threshold value γ and the parameters of the sigmoid function in y and m, balance the system out to form illusory contours. Only crude variations were made due to the time-frame of this research; further investigation into the effects of the kernels is warranted.

The limitations of implementing the model with only two orientations was seen in test images with curves or angled lines. For example, in Fig. 4(c) the model boxes the circle in.

The system of equations was found to form a non-stiff problem. Attempts at using implicit solvers failed. This was due to the large Jacobian needed for these methods. Not only was this step very slow, it also caused the system to run out of memory. Explicit methods were unnecessary however as the problem appeared to be non-stiff. Implicit methods were therefore adequate. Of the three implicit methods tested, *Tsit5* appeared to be most suitable for the system when the visual illusions were used as input. The faster *BS3* yielded an output close to *Tsit5* until it ran into stability issues. This was not surprising as *BS3* was designed for problems with a higher tolerance then those for *Tsit5*. The third implicit method tested, *Vern7*, was slower then *Tsit5*. Vern7 was designed for problems with a lower tolerance.

Issues occurred with *Tsit5* when the model was tested with real-world images with noise. The system ran out of memory as the solver decreased its step-size for images with high levels of noise. The use of another solver such as *Vern7* may avoid this issue. This was not investigated in this research. Alternatively, stopping criteria such as restricting the number of steps or changing error tolerance of *Tsit5* may help.

The reason for the change in step-size appeared to be due to m (the inhibitory part of L4) becoming periodic. Periodicity was also seen with two parameter variations: when the feedback to x from z was increased by setting $\phi = 4$ and when T^+ was doubled. It was not seen however when *all* feedback was doubled. It is unclear whether the periodicity originated in the system itself or whether it was an artifact of the solver. Further investigation could be done to determine this.

A lack of robustness to noise was reported with FACADE [9]. It would appear that LAMINART has similar issues. Approaches used by [9] included elongating some kernels. Adjustments to kernel shape here might also help.

Parallelisaion was effective in increasing the efficiency of the implementation. The benefits of parallelisation were found to increase when larger input images were used. The performance times on a CPU appeared to increase exponentially where as only a relatively slight increase occurred on a GPGPU.

An issue arose when the convolution function, `NNlib.conv`, was used with a sliced *CuArray* on the GPGPU. Rather than slicing the array with *Julia*'s `@view u[...]`, three arrays, x, m and s were copied to individual, pre-allocated arrays. This step could be improved upon.

Careful consideration had to be given to bordering artifacts near the edge of the image. There did not appear to be any issues when zero padding was used around the array.

VI. Conclusion

A version of LAMINART , up to V1, was implemented on a GPGPU, using numerical methods to solve the system of ODEs. Three sets of the model's equations were rearranged making several operations redundant. The model was found to form illusory contours when tested with visual illusory

images. These contours were able to form in illusions where contrast polarity switched across the boundary. Parallelisation was effective in increasing efficiency of the system. The benefits increased for larger input images when compared to a non-parallel implementation.

The model was found to have a low tolerance to noise and appeared to become unstable or periodic. A better strategy could be developed to handle this scenario.

Further investigation into variations to the models parameters could help gain a deeper understanding of the complex feedback pathways. The model could be expanded to include V2, either with the use of equations at equilibrium or a full implementation of the higher cortex.

This implementation relied on self-organised kernels from [8]. These kernels restricted the model to two orientations. Estimations of values for these kernels had to be made. Two approaches might be taken to develop kernels for higher orientations.

A better strategy to estimate the kernels could be developed. The kernels used between $v^{\pm}$ and C are similar to Gabor functions. $W^{\pm}$, $T^{\pm}$ and H are all Gaussian shaped. Efficient convolution might be possible if the kernels are defined by functions.

Another approach could be to use a network to learn these kernels. Either the methods used in [8] could be replicated or recent artificial neural network techniques could be applied. Recent developments which use differential equations and neural networks together [24] might be particularly suited to adapt the model. This approach might be effective in increasing the robustness of the model to noise.

The datatype of the arrays used was *Float32*. This was to achieve a high number of operations from the GPGPU used. This level of precision is not required by the model and the use of *Float16* with an appropriate GPGPU may be more efficient.

Existing computer vision techniques such as [5] would appear to be more robust to noise than LAMINART . The low noise tolerance of the current model reduces its suitability for computer vision applications. The model was found to be highly parallelisable. A balance between the model's kernels and other parameters is central to stability and robustness. Further research into the roles of these parameters may make future use of the model with real-world applicaitons feasible.

REFERENCES

[1] N. V. K. Medathati, H. Neumann, G. S. Masson, and P. Kornprobst, "Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision", *Computer Vision and Image Understanding*, vol. 150, pp. 1–30, Sep. 1, 2016. DOI: 10/f8w8cj.

[2] S. Grossberg and R. D. Raizada, "Contrast-sensitive perceptual grouping and object-based attention in the laminar circuits of primary visual cortex", *Vision Research*, vol. 40, no. 10-12, pp. 1413–1432, Jun. 2000. DOI: 10.1016/S0042-6989(99)00229-1.

[3] A. Doerig, A. Bornet, R. Rosenholtz, G. Francis, A. M. Clarke, and M. H. Herzog, "Beyond Bouma's window: How to explain global aspects of crowding?", *PLOS Computational Biology*, vol. 15, no. 5, e1006580, May 10, 2019. DOI: 10/ggp7nj.

[4] G. Francis, M. Manassi, and M. H. Herzog, "Neural dynamics of grouping and segmentation explain properties of visual crowding", *Psychological Review*, vol. 124, no. 4, pp. 483–504, Jul. 2017. DOI: 10/gbf8r8.

[5] L. Tan, W. Liu, L. Li, and Z. Pan, "A fast computational approach for illusory contour reconstruction", *Multimedia Tools and Applications*, vol. 78, no. 8, pp. 10 449–10 472, Apr. 1, 2019. DOI: 10/gg8fxh.

[6] S. Grossberg, "The complementary brain: Unifying brain dynamics and modularity", *Trends in Cognitive Sciences*, vol. 4, no. 6, pp. 233–246, Jun. 1, 2000. DOI: 10.1016/S1364-6613(00)01464-9.

[7] R. D. Raizada and S. Grossberg, "Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast", *Visual Cognition*, vol. 8, no. 3-5, p. 431, Jun. 2001. DOI: 10/d8phhc.

[8] S. Grossberg and J. R. Williamson, "A Neural Model of how Horizontal and Interlaminar Connections of Visual Cortex Develop into Adult Circuits that Carry Out Perceptual Grouping and Learning", *Cerebral Cortex*, vol. 11, no. 1, pp. 37–58, Jan. 1, 2001. DOI: 10/bn377j.

[9] D. Hu, Z. Zhou, and Z. Wang, "Processing real-world imagery with FACADE-based approaches", *Frontiers of Electrical and Electronic Engineering in China*, vol. 6, no. 1, pp. 120–136, Mar. 2011. DOI: 10/c8h7q5.

[10] N. Cullinane, "GPGPU-accelerated specialized neural models of visual phenomena", MEng Final Portfolio, Appendix D: Design & Implementation, Aug. 31, 2020.

[11] R. D. Raizada and S. Grossberg, "Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast", *Visual Cognition*, vol. 8, no. 3-5, p. 431, Jun. 2001. DOI: 10/d8phhc.

[12] C. Tsitouras, "Runge Kutta pairs of order 5(4) satisfying only the first column simplifying assumption", *Computers & Mathematics with Applications*, vol. 62, no. 2, pp. 770–775, Jul. 1, 2011. DOI: 10/d5p2zf.

[13] P. Bogacki and L. Shampine, "A 3 (2) pair of Runge Kutta formulas", *Applied Mathematics Letters*, vol. 2, no. 4, pp. 321–325, 1989. DOI: 10/cwcdkx.

[14] J. H. Verner, "Numerically optimal Runge Kutta pairs with interpolants", *Numerical Algorithms*, vol. 53, no. 2-3, pp. 383–396, Mar. 2010. DOI: 10/b669sf.

[15] L. F. Shampine and M. W. Reichelt, "The MATLAB ODE Suite", *SIAM Journal on Scientific Computing*, vol. 18, no. 1, pp. 1–22, Jan. 1997. DOI: 10/crztgb.

[16] L. Petzold, "Automatic Selection of Methods for Solving Stiff and Nonstiff Systems of Ordinary Differential Equations", *SIAM Journal on Scientific and Statistical Computing*, vol. 4, no. 1, pp. 136–148, Mar. 1983. DOI: 10/bk72v5.

[17] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing", *SIAM Review*, vol. 59, no. 1, pp. 65–98, 2017. DOI: 10/f9wkpj.

[18] C. Rackauckas and Q. Nie, "DifferentialEquations.jl, A Performant and Feature,Rich Ecosystem for Solving Differential Equations in Julia", *Journal of Open Research Software*, vol. 5, no. 1, p. 15, May 25, 2017. DOI: 10/ggfnfj.

[19] M. Innes, E. Saba, K. Fischer, D. Gandhi, M. C. Rudilosso, N. M. Joy, T. Karmali, A. Pal, and V. Shah, "Fashionable modelling with flux", *CoRR*, vol. abs/1811.01457, 2018. arXiv: 1811.01457.

[20] "CUDA C Programming Guide", NVIDIA, White Paper PG-02829-001_v8.0, Jun. 2017, p. 280.

[21] T. Besard, C. Foket, and B. De Sutter, "Effective Extensible Programming: Unleashing Julia on GPUs", *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 827–841, Apr. 2019. DOI: 10/gf7rfk.

[22] R. M. Williams and R. V. Yampolskiy, "Optical Illusions Images Dataset", Oct. 16, 2018. arXiv: 1810.00415.

[23] N. Cullinane, "GPGPU-accelerated specialized neural models of visual phenomena", MEng Final Portfolio, Appendix E: Testing and Results, Aug. 31, 2020.

[24] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, and A. Ramadhan, "Universal Differential Equations for Scientific Machine Learning", Jan. 13, 2020. arXiv: 2001.04385.

Appendix A

Literature review

REVISIONS

A submission of this review was not made in January.

At that time the focus of the project was on investigating whether the FACADE model was feasible. As the project developed, the model of interest changed to LAMINART , the focus of this paper.

A survey of GPGPU-accelerated specialized neural models of visual phenomena

Niall Cullinane

***Abstract*—Illusory contours are used in biological visual systems to process occluded objects. Investigation into the formation of these contours may be of benefit when developing computer vision systems robust to occlusion. LAMINART , a computational model of the mammalian visual system is capable of forming illusory contours. Its structure is highly parallelisable and may be suitable for implementation on a GPGPU. Parallelisation may make the model feasible for use in computer vision applications.**

***Index Terms*—Bio-inspired computing, computational neuroscience, computer vision, biological neural networks, feedback, parallel processing.**

A.I. INTRODUCTION

VISUAL illusions trick the mind, exposing some of the sketches under the paint as the brain constructs its representation of the world. Neuroscientists have used illusions to gain insight into some of the brain's processing mechanisms, testing where perception and reality fail to meet.

One form of illusion, such as those shown in Fig. A.1, show the occurrence of illusory contours. A Kanizsa Square, seen in Fig. A.1 (a), gives the illusion of a white square. At the top of the image, the brain sees two short horizontal contours of high contrast to the left and right. A connection is made between the short contours and the brain continues them out to form the top of a square. The brain perceives a white square, the corners of which are *occluding* four black circles. Illusory contours are agnostic to contrast polarity, as can be seen in Fig. A.1 (b,c).

These illusory contours are related to how the brain deals with occluded objects. Tasks involving occlusion remain a challenge for computer vision. Occlusion occurs in segmentation and figure-ground segregation tasks, both of which have a wide variety of applications. A comparison between a selection of biological models and computer vision techniques for segmentation was given in [2]. Computer vision approaches discussed there included artificial neural networks with decoder networks and attention as well as methods combining clustering and edge detection.

LAMINART [3] was developed, in part, to account for the occurrence of illusory contours in the mammalian brain. The development was based upon a large selection of psychophysical and neurophysiological experiments. LAMINART refined ideas from FACADE , a model which [2] described as promising for computer vision segmentation.

The aim of this research is to investigate the formation of illusory contours in LAMINART and determine whether a computationally efficient and robust implementation of the model can be developed to make its future use in computer vision applications feasible.

N. Cullinane is with the School of Electronic Engineering, Dublin City University, Glasnevin, Dublin, Ireland.

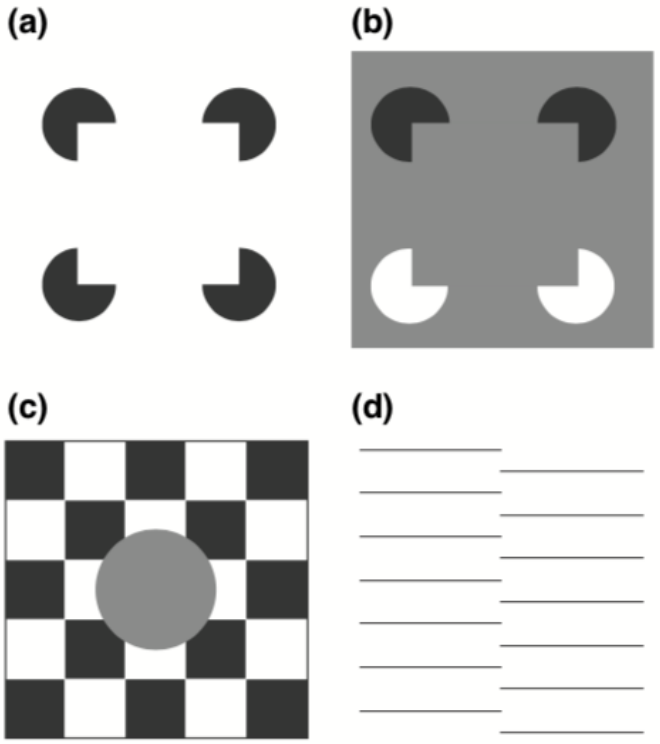


Figure A.1. Four visual illusions involving ilusory boundaries. [1]

A.II. REVIEW AND ANALYSIS OF PRIOR WORK

LAMINART [3] unified two previous ideas, Boundary Contour System/Feature Contour System (BCS/FCS), which FACADE is based upon, and Adaptive Resonance Theory (ART).

LAMINART was expanded into 3D LAMINART to model mammalian binocular vision. Ideas from ART and FACADE have been used in computer vision. Some attempts have been made to fully implement FACADE for computer vision. There appear to be no implementations of LAMINART by computer vision engineers. The model however, has been recently used by computational neuroscientists in modeling visual crowding.

A. Prior models to LAMINART

ART set out a series of rules relating to how feedback and feedforward signals interact in the brain [3]. The theory was developed to account for the brain's ability to learn without catastrophic forgetting, a problem called the stability-plasticity

dilemma. A feedforward signal activates a cell in a higher level. This then feeds back and excites and inhibits cells through on-centre off-surround interactions. Both the feedforward and feedback pathways have threshold functions. The theory's rules relate how a cell activates with these interactions in four scenarios. A cell can be activated if a threshold is met by a feedforward signal. It can also be activated if a threshold is met by a combination of feedback and feedforward signals. A cell can receive a signal though a feedback pathway which is not enough to activate it; it will however be more sensitive to feedforward signals. Lastly, a cell's feedforward signal can be inhibited by a feedback signal enough to supress activation.

ART was developed to account for higher brain mechanisms such as learning. It failed to account for the formation of illusory contours in the visual system.

BCS/FCS outlined a network with parallel pathways which complement each other [3]. Cells at boundaries are excited in the Boundary contour system (BCS). The Feature Contour System (FCS) retains surface information. This is combined with the BCS to fill in surfaces within the boundaries.

B. LAMINART *model*

LAMINART , first outlined in [3] and with mathematical details and results provided in [4], built a model of the lower levels of the visual system. The model included the retinal cells, the lateral geniculate nucleus (LGN), primary visual cortex (V1) and secondary visual cortex (V2). Illusions form, by definition, when cells have little or no bottom up signal. They were known, from previous experiments, to occur at a low level of the visual system. The model was developed to account for the formation of illusions at such low layers.

The model, shown in Fig. A.2, consisted of layers of cells that excite and inhibit each other though feedforward, feedback and horizontal interactions. These interactions may be broken down into several sub-circuits, each accounting for specific experimental data. Together, these interactions form a complex network.

The network was represented mathematically by a system on first-order, time-dependent, ordinary differential equations. The horizontal interactions were represented by convolution operations.

The output of the LGN goes through a difference of oriented Gaussians (DOOG). This results in the arrays of cells in V1 and V2 having a third dimension of orientations.

Several on-centre, off-surround pathways exist. These pathways work by a direct connection exciting while a Gaussian shaped kernel inhibits. Half-wave rectifiers and threshold functions add non-linearity to the network. This rectification means that cells can be in a sub-threshold state.

Illusory contours form in L2/3 . When a boundary is detected by the network, it continues this contour out through feedback. This is inhibited however if no matching boundary is detected along that orientation in that direction, as shown in Fig. A.3(a). If a matching boundary is detected, the inhibition is reduced, forming an illusory contour, as shown in Fig. A.3(b). The maximum gap which illusions can form in the network was governed by the size of kernels used in L2/3 . The size of the kernels used in L2/3 V2 are greater then in L2/3 V1.

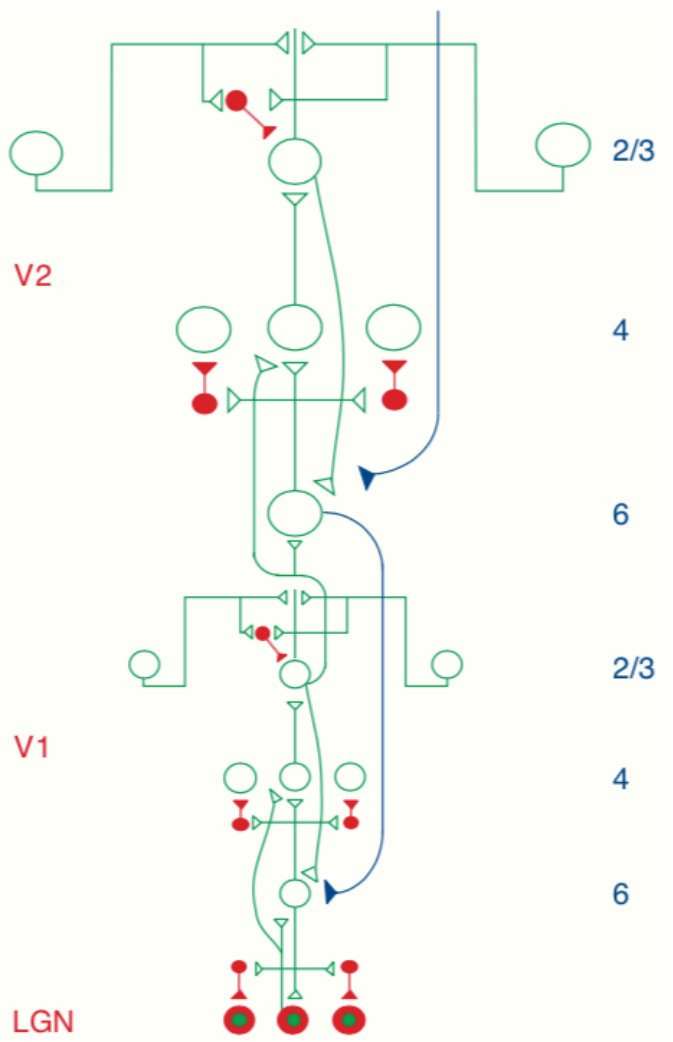


Figure A.2. The LAMINART model. The model is made up of feedforward, feedback and horzontal interactions. The solid red circles are inhibiting the signal. [1]

The model was refined in [4]. Sigmoid functions were added to parts of L4 . This led to weaker inhibition for areas with low contrast and stronger inhibition for areas with high contrast. This in effect normalized the whole output. Some parameters in L2/3 were reduced. Some simulations were run with additional inputs to L6 and L2/3 . This was in order to represent feedback from higher cortex to model attention.

Several kernels used in [4] were from results of [5]. In [5], the W kernels of L4 and the H and T kernels of L2/3 were learned using unstructured and structured inputs. The unstructured inputs simulated pre-natal learning whilst the structured inputs simulated post-natal learning. Several differences between the equations of this paper and of LAMINART were not discussed, for example the function F is defined differently. The kernels were only run at two orientations. Results for the H and W kernels were not reported numerically, they were only given graphically. The T kernels had different results for each orientation. Using randomly configured rectangles as structured input, it could be expected that the kernel would be the same

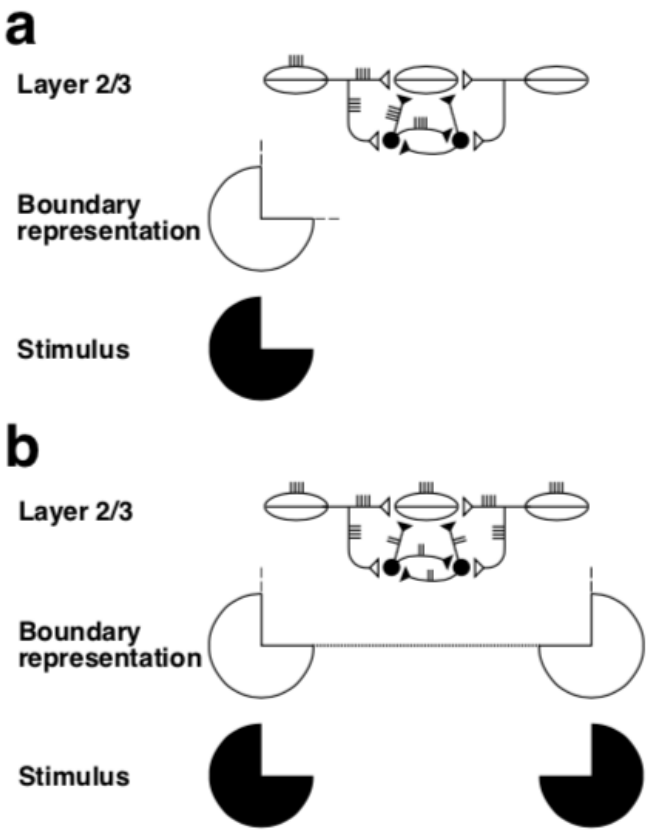


Figure A.3. Bipole circuit of L2/3 . Inhibition restricts the growth of an illusory contour in (a) . This inhibition is reduced in (b), allowing the formation of a contour. [3]

for both orientations. This discrepancy was not discussed. The use of a circular window to restrict learning was not clearly justified.

C. Recent implementations of LAMINART

Attempts at implementing LAMINART in computer vision appear to be rare. An investigation of FACADE with real-world images was made by [6]. The model was found not to be very robust, in particular for images with noise. An imbalance of excitation and inhibition at the bipolar cells was also reported. Oriented, anisotropic Gaussians were used to reduce the effect of noise. The bipole cell was also altered. Increases in robustness and efficiency were reported. Some dynamical equations were solved at equilibrium to calculate parts of the model. It is unclear how appropriate that may be. Although briefly discussed, it is unclear why the authors chose to implement FACADE over LAMINART .

LAMINART was implemented in the neuroscience domain by [7], with further work done by [8]. The LAMINART model was adapted to simulate visual crowding, a phenomena seen in neuroscience relating to decreased object recognition where there are more objects in the visual field. The model was run as a spiking neural network using exact digital simulation. LAMINART 's ability to simulate boundary grouping gave it an advantage over other methods for simulating visual crowding. The model was extended from four to eight orientations by [8]. It is unclear from the paper and code how the number of orientations was increased to eight. Specifically, it was unclear how the number of orientations of the model's self-organised kernels was increased.

A.III. RELATION OF PRIOR WORK TO PROJECT PROBLEM

LAMINART would appear to be a suitable model to investigate the formation of illusory contours. It is more refined than FACADE, as it incorporates ART. The implementations of LAMINART made by [7] and [8] were focused on developing an accurate model of a biological system to mimic certain phenomena. A spiking neural network using exact digital simulation was used. The focus of this research was to investigate if an *efficient* implementation may be made. A numerical approach, such as using Runge-Kutta type methods, may be more appropriate for developing an efficient implementation.

Variants of Runge-Kutta have been parallelised [9]. Operations involved in LAMINART's equations, in particular the convolution operations, are also parallelisable. The implementation of a parallelised version of LAMINART should yield gains in efficiency from a non-parallelised version. This should be achievable on a general purpose graphics processing unit (GPGPU) with the use of CUDA [10]. CUDA is a proprietary platform which allows access to NVidia GPGPUs.

The size of the system of ordinary differential equations (ODEs) may be challenging to implement. An implementation of the model up to V1 should be enough to investigate the formation of illusory contours. This partial implementation should greatly increase the feasibility of this research.

The dependence of the model on the self-organised kernels developed by [5] may also create some challenges. These kernels were only given graphically. A full implementation of the methods used there is outside the scope of this current work. The literature is unclear as to what approach [7] and [8] took. These self-organised kernels were only given for two orientations. [5]. A strategy would need to be developed for learning or estimating kernels in order to increase the number of orientations of the model.

A.IV. CONCLUSION

The LAMINART model was chosen to be implemented to investigate the formation of illusory contours. The model uses a network of feedback, feedforward and horizontal interactions with non-linear operators. These interactions allow the formation of illusory contours through on-centre off-surround circuits. The model is defined by a system of first-order ODEs. The structure of the model makes it highly parallelisable.

The horizontal interactions use self-organised kernels. Values for these kernels have only been reported graphically.

REFERENCES

[1] S. Grossberg, "The complementary brain: Unifying brain dynamics and modularity", *Trends in Cognitive Sciences*, vol. 4, no. 6, pp. 233–246, Jun. 1, 2000. DOI: 10.1016/S1364-6613(00)01464-9.

[2] N. V. K. Medathati, H. Neumann, G. S. Masson, and P. Kornprobst, "Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision", *Computer Vision and Image Understanding*, vol. 150, pp. 1–30, Sep. 1, 2016. DOI: 10/f8w8cj.

[3] S. Grossberg and R. D. Raizada, "Contrast-sensitive perceptual grouping and object-based attention in the laminar circuits of primary visual cortex", *Vision Research*, vol. 40, no. 10-12, pp. 1413–1432, Jun. 2000. DOI: 10.1016/S0042-6989(99)00229-1.

[4] R. D. Raizada and S. Grossberg, "Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast", *Visual Cognition*, vol. 8, no. 3-5, p. 431, Jun. 2001. DOI: 10/d8phhc.

[5] S. Grossberg and J. R. Williamson, "A Neural Model of how Horizontal and Interlaminar Connections of Visual Cortex Develop into Adult Circuits that Carry Out Perceptual Grouping and Learning", *Cerebral Cortex*, vol. 11, no. 1, pp. 37–58, Jan. 1, 2001. DOI: 10/bn377j.

[6] D. Hu, Z. Zhou, and Z. Wang, "Processing real-world imagery with FACADE-based approaches", *Frontiers of Electrical and Electronic Engineering in China*, vol. 6, no. 1, pp. 120–136, Mar. 2011. DOI: 10/c8h7q5.

[7] G. Francis, M. Manassi, and M. H. Herzog, "Neural dynamics of grouping and segmentation explain properties of visual crowding", *Psychological Review*, vol. 124, no. 4, pp. 483–504, Jul. 2017. DOI: 10/gbf8r8.

[8] A. Doerig, A. Bornet, R. Rosenholtz, G. Francis, A. M. Clarke, and M. H. Herzog, "Beyond Bouma's window: How to explain global aspects of crowding?", *PLOS Computational Biology*, vol. 15, no. 5, e1006580, May 10, 2019. DOI: 10/ggp7nj.

[9] K. Ahnert, D. Demidov, and M. Mulansky, "Solving ordinary differential equations on GPUs", in, Jan. 1, 2014, pp. 125–157. DOI: 10.1007/978-3-319-06548-9_7.

[10] "CUDA C Programming Guide", NVIDIA, White Paper PG-02829-001_v8.0, Jun. 2017, p. 280.

Appendix B

Project plan

Project Plan

Niall Cullinane

Sunday 4th October, 2020

Project Title: GPGPU-accelerated specialized neural models of visual phenomena
Supervisor: Dr Noel Murphy
Student Number: 18214952
Version: 1.2
Revisions:
v1.1 to v1.2: Replaced the term *illusory boundaries* with *illusory contours* in research question and design approach; formatted document to match portfolio.

B.I. Research Question

How are illusory contours formed in Grossberg's LAMINART model and can a computationally efficient and robust implementation of the model be developed to make its future use in computer vision applications feasible?

B.II. Project Scope

The project scope will comprise of

- the LAMINART model, a model of the human visual system developed in the theoretical neuroscience domain, from a computer vision engineering perspective;
- numerical methods appropriate for solving large systems of ordinary differential equations (ODEs), with a focus on *Euler-like* techniques;
- implementation of a partial version of the model for monochromatic, monocular, static images;
- use of *Julia*, a high-level, general-purpose language;
- use of existing high-level libraries where possible such as
  - *JuliaImages.jl* for image I/O, filtering,
  - *DifferienicalEquations.jl* for numerical methods;
- code optimisation techniques such as BLAS calls, through high-level libraries where possible;
- appropriate parallelisation techniques and GPGPU technologies such as CUDA through Julia's *CUArray.jl* library or similar and
- established computer vision techniques for boundary detection (to evaluate the LAMINART model against).

Mathematical methods outside the scope of the project include

- *exact integration*, currently popular in computational neuroscience, and
- *Bayesian* techniques for parameter optimisation.

B.III. Design Approach

The design phase of the project will comprise of

- the identification of tasks in computer vision which the model may be well suited to and are feasible to be investigated;
- identification of computer vision techniques with which the model can be evaluated against;
- the design of tests involving these tasks in order to evaluate the model's performance in terms of
  - computational efficacy and
  - robustness;
- optimisation of the model's equations by, for example, the elimination of redundant operations or by the use of modern filtering techniques from image processing;
- identification of appropriate
  - numerical methods,
  - libraries and data types,
  - methods for optimisation (such as BLAS, non-allocation, type-stability) and
  - parallelisation techniques;
- development of a design which can be ported for GPGPU processing as smoothly as possible;

The model will be implemented in the following stages:

- non-ODE functions/layers at equilibrium,
- ODE layers using CPU processing,
- porting of code for GPGPU processing.

Each of these stages will go through

- unit-testing,
- optimisation and
- benchmarking / profiling steps.

An optimised implementation of the model will be evaluated using the previously designed tests.

Results will be analysed to

- evaluate the effectiveness of various optimisation and parallelisation strategies;
- compare the computational efficiency and robustness of the model with traditional computer vision techniques;
- examine the formation and role of illusory contours and
- explore the role of various feedforward and feedback pathways within the model.

Good engineering and academic practices shall be maintained throughout including

- well-documented code,
- use of a VCS and
- clear notes on all decisions made.

B.IV. Confirmation of Remote Working Arrangements

The project will be able to be competed remotely.

- There is access to an adequate computer, internet connection and workspace;
- the faculty to have regular student/supervisor progress meetings online;
- the faculty to conduct interviews online;
- a VPN connection to the School allowing connection to the School's GPGPU server to run experiments remotely;
- support from technical officers in the School if required;
- remote access to other supports from the University remotely and
- remote access to journals through the University's Library although
  - access to some other Library resources is limited.

B.V. Detailed Timeline

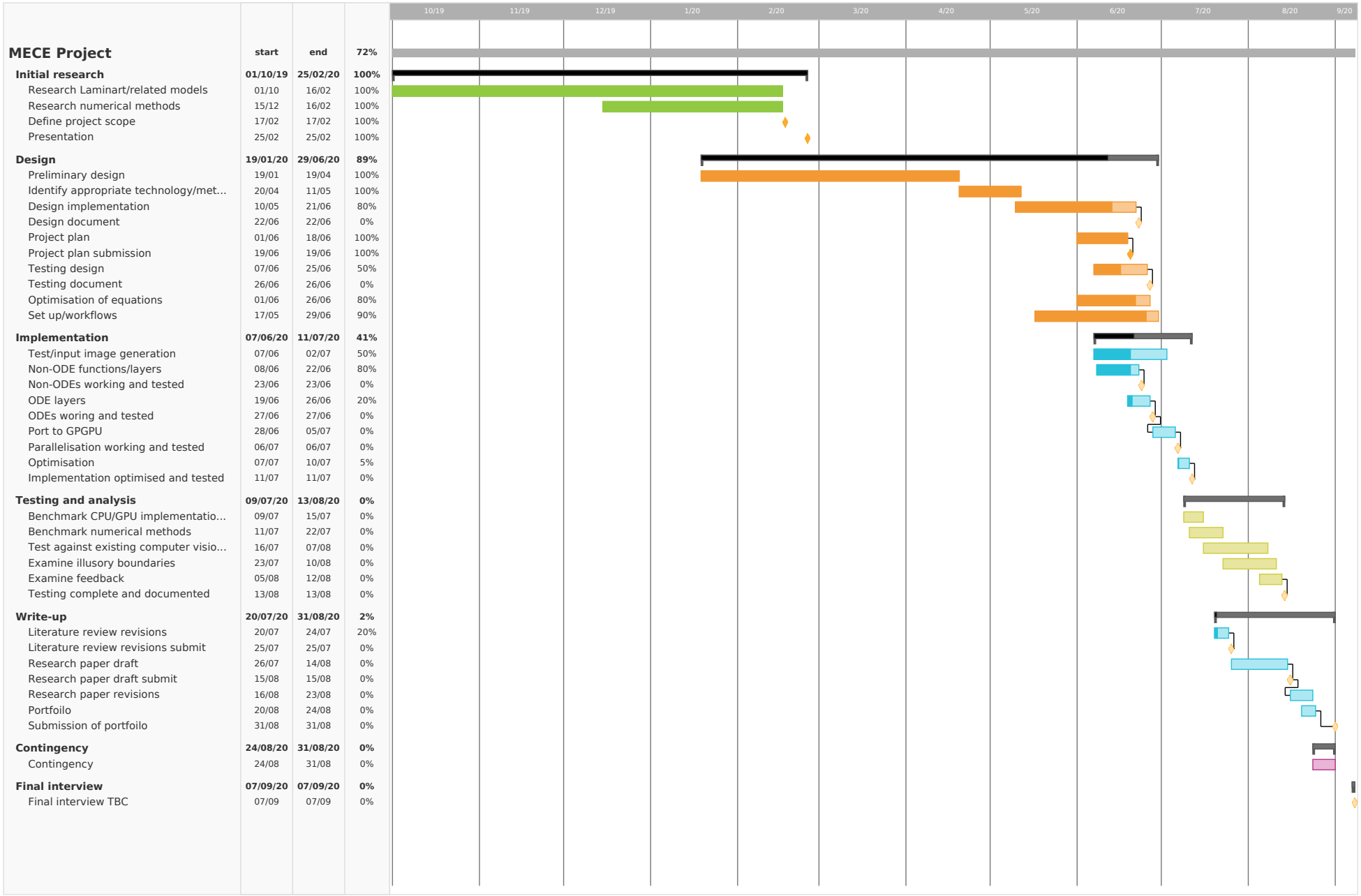
A detailed timeline is shown in the form of a Gantt chart in Fig. B.1.

B.VI. Success Criteria

The project will be considered successful if the following criteria are met:

- the formal requirements of the programme have been compleated:
  - a research seminar MCQ,
  - a presentation,
  - an interview following the presentation,
  - a project portfolio and
  - a final interview;
- a working implementation of a portion of the LAMINART model has been developed (within the scope of the project),
- evaluated through the use of appropriate tests and
- compared against existing techniques in computer vision.

A working, parallelised implementation of the model on a GPGPU would be considered a bonus to these criteria.



MECE Project	start	end	72%
Initial research	**01/10/19**	**25/02/20**	**100%**
Research Laminart/related models	01/10	16/02	100%
Research numerical methods	15/12	16/02	100%
Define project scope	17/02	17/02	100%
Presentation	25/02	25/02	100%
Design	**19/01/20**	**29/06/20**	**89%**
Preliminary design	19/01	19/04	100%
Identify appropriate technology/met...	20/04	11/05	100%
Design implementation	10/05	21/06	80%
Design document	22/06	22/06	0%
Project plan	01/06	18/06	100%
Project plan submission	19/06	19/06	100%
Testing design	07/06	25/06	50%
Testing document	26/06	26/06	0%
Optimisation of equations	01/06	26/06	80%
Set up/workflows	17/05	29/06	90%
Implementation	**07/06/20**	**11/07/20**	**41%**
Test/input image generation	07/06	02/07	50%
Non-ODE functions/layers	08/06	22/06	80%
Non-ODEs working and tested	23/06	23/06	0%
ODE layers	19/06	26/06	20%
ODEs woring and tested	27/06	27/06	0%
Port to GPGPU	28/06	05/07	0%
Parallelisation working and tested	06/07	06/07	0%
Optimisation	07/07	10/07	5%
Implementation optimised and tested	11/07	11/07	0%
Testing and analysis	**09/07/20**	**13/08/20**	**0%**
Benchmark CPU/GPU implementatio...	09/07	15/07	0%
Benchmark numerical methods	11/07	22/07	0%
Test against existing computer visio...	16/07	07/08	0%
Examine illusory boundaries	23/07	10/08	0%
Examine feedback	05/08	12/08	0%
Testing complete and documented	13/08	13/08	0%
Write-up	**20/07/20**	**31/08/20**	**2%**
Literature review revisions	20/07	24/07	20%
Literature review revisions submit	25/07	25/07	0%
Research paper draft	26/07	14/08	0%
Research paper draft submit	15/08	15/08	0%
Research paper revisions	16/08	23/08	0%
Portfoilo	20/08	24/08	0%
Submission of portfoilo	31/08	31/08	0%
Contingency	**24/08/20**	**31/08/20**	**0%**
Contingency	24/08	31/08	0%
Final interview	**07/09/20**	**07/09/20**	**0%**
Final interview TBC	07/09	07/09	0%

Figure B.1. Detailed timeline for the project shows as a Gantt chart.

Appendix C

Research log

Masters Project Research Log

Masters in Electronic and Computer Engineering 2019/2020

Student Name: Niall Cullinane
Student ID: 18214952
Project Title: GPGPU-accelerated specialized neural models of visual phenomena

Please read before making entries in this log

The purpose of this Project Research Log is to capture concise, focused summaries of research materials you read, as you progress through your project. The emphasis is to record (i) how the material you have read will determine or influence your project solution approach and (ii) your assessment of the key strengths and weaknesses of the solutions, methods, technologies, etc. proposed in the material you have read.

In the first stage of your project, the literature review, use the Log to capture this information for the key papers you have read (for example, the three most important papers of your 10 literature review references). As your project progresses into the design and implementation phases, you will need to continue to search the literature so you can review, revise and refine your initial thinking and the details of your approach to a project solution. Use this Research Log to capture your continued research reading and its influence on your project design and implementation.

Be selective about what you record in this log. Do not use it as an informal notebook while you are reading a new paper. Only make an entry after you have read a paper that you consider important to the development of your project solution. It is expected that, by the end of the project, you will have made **between 10 and 20 entries (20 maximum)**.

Your log will be shared with your supervisor for viewing throughout the project and you will submit the final version of the log for grading, at the end of the project implementation period. It will be assessed on the basis of how well you have used your analysis of the literature to inform your project design, implementation and the evaluation of your project results. The Research Log contributes **5%** to the overall project mark.

Note: All entries you make in this log must use the prescribed format. A new entry with the correct format is generated using the “Research Log” menu above (it may take a minute for this menu to appear). Do not make any other entries or change the format of the generated tables. You will maintain other notes as you progress through your project but they should not be recorded here. Do not exceed the maximum word counts indicated.

Statement of project problem / research question (maximum 200 words)
This statement should be periodically reviewed and updated, as necessary, as your project progresses and you gain further insight into the detailed project challenges, requirements and objectives as your project work moves from background reading, literature review, initial project design planning and detailed design and implementation. Initially, start by stating your current understanding of the project objectives. After each meeting with your supervisor, review and refine your project problem statement, as required.

How are illusory contours formed in Grossberg's LAMINART model and can a computationally efficient and robust implementation of the model be developed to make its future use in computer vision applications feasible?

How does the cerebral cortex work? Learning, attention, and grouping by the laminar circuits of visual cortex
S. Grossberg, ‘How does the cerebral cortex work? Learning, attention, and grouping by the laminar circuits of visual cortex’, *Spatial vision*, vol. 12, no. 2, pp. 163–185, 1999, doi: 10.1163/156856899x00102.
https://doi.org/10.1163/156856899x00102
Summary of paper (maximum 100 words)
An overview of subcircuits that make up the LAMINART model was given. The motivation that led to unifying two existing models: Adaptive Resonance Theory (ART) and Boundary Contour System (BCS) into LAMINART was discussed. Specifically, LAMINART was developed to account for perceptual groupings forming 1) pre-attentively with 2) no bottom up signal at a position. Four pre-attentive and four attentive subcircuits were presented.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The development of LAMINART from previous models was outlined. Perceptual groupings were discussed in relation to the model.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The roles eight subcircuits have in LAMINART were clearly outlined. The model was discussed in relation to ART and BCS. Perceptual grouping was also discussed. Mathematical details of the model were not given.
Log Entry Creation Date: Wed Aug 05 2020 12:34:16 GMT+0100 (BST)

Brightness perception, illusory contours, and corticogeniculate feedback
A. Gove, S. Grossberg, and E. Mingolla, 'Brightness perception, illusory contours, and corticogeniculate feedback', *Visual Neuroscience*, vol. 12, no. 6, pp. 1027–1052, Nov. 1995, doi: 10.1017/S0952523800006702.
https://doi.org/10.1017/S0952523800006702
Summary of paper (maximum 100 words)
The FACADE model of the visual cortex was presented. The model was developed to account for illusory contours seen with several visual illusions. It builds on BCS/FCS model by proposing feedback to LGN from V1. The occurrence of brightness buttons at line ends suggested that this feedback has an on-centre off-surround network. The feedback is inline with the rules proposed in ART. Results from simulations with several visual illusions were presented.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The model proposed was FACADE, a precursor to LAMINART. The role which V1-LGN feedback plays in the formation of illusory contours was thoroughly outlined. Several major changes were made between FACADE and LAMINART, for example the on centre feedback to LGN goes through Gaussian in FACADE but does not in LAMINART.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The reasons that lead to the suggestion of V1-LGN feedback were thoroughly discussed. The model was presented in detail mathematically. Results from simulations with several illusions were presented and discussed. The model differs extensively to LAMINART making direct comparisons difficult.
Log Entry Creation Date: Wed Aug 05 2020 12:36:09 GMT+0100 (BST)

Contrast-sensitive perceptual grouping and object-based attention in the laminar circuits of primary visual cortex

S. Grossberg and R. D. S. Raizada, 'Contrast-sensitive perceptual grouping and object-based attention in the laminar circuits of primary visual cortex', *Vision Research*, vol. 40, no. 10–12, pp. 1413–1432, Jun. 2000, doi: 10.1016/S0042-6989(99)00229-1.

https://doi.org/10.1016/S0042-6989(99)00229-1

Summary of paper (maximum 100 words)

The LAMINART model was proposed.
The model was developed to account for perceptual grouping occurring across areas where there was no direct input. Several feedforward, feedbackward and horizontal interactions were proposed which either excite and inhibit.
These interactions allow attention to feedback onto an area. This, in turn, can vary the contrast threshold for stimulation. Gabor images with various contrasts were used to test this. The use of illusions to test perceptual grouping was briefly discussed.

How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)

The LAMINART model was presented with mathematical details. The model's pathways were discussed in relation to perceptual grouping and illusory contours.

What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)

The neurophysiological and psychophysical results which led to the model's development were discussed. The model's architecture was clearly outlined and mathematical descriptions were given. A summary of roles of the model's pathways was given.
Some previously learnt, self-organizing kernels were not reported numerically, only graphically. These kernels were only given for two orientations.
Details of how some parameters were chosen was lacking.
Some dynamical equations were solved at equilibrium and used for parts of simulations. It is unclear how accurate this approach is.
No comparison was made between LAMINART and previous, related models.
The model was updated a year later by the same authors.

Log Entry Creation Date: Wed Aug 05 2020 12:36:12 GMT+0100 (BST)

Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast

R. D. S. Raizada and S. Grossberg, 'Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast', *Visual Cognition*, vol. 8, no. 3–5, p. 431, Jun. 2001, doi: 10/d8phhc.

https://doi.org/10/d8phhc

Summary of paper (maximum 100 words)

The LAMINART model was presented with several alterations.
Sigmoid functions were added to parts of L4. This led to weaker inhibition for areas with low contrast and stronger inhibition for areas with high contrast. This, in effect, normalized the whole output. Some parameters in L2/3 were reduced.
Some simulations were run with additional inputs to L6 and L2/3.

How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)

This version of LAMINART was chosen to be implemented due to the refinements made of the previous version.
Illusory contours were discussed in relation to LAMINART.
The additional attention inputs are outside the project's scope.

What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)

The model was clearly outlined and mathematical descriptions were given. Illusory contours were discussed.
Similar to the previous paper, some of the self-organizing kernels were not reported numerically. These kernels were, again, only given for two orientations.
Details of how some parameters were chosen was lacking. It was mentioned that connectivity was empirically determined but it is unclear how neurophysiological results were tied together to form the model.
Similar to the previous paper, some dynamical equations were solved at equilibrium and used for parts of simulations.
No comparison was made between LAMINART and previous, related models.

Log Entry Creation Date: Wed Aug 05 2020 12:36:14 GMT+0100 (BST)

A Neural Model of how Horizontal and Interlaminar Connections of Visual Cortex Develop into Adult Circuits that Carry Out Perceptual Grouping and Learning

S. Grossberg and J. R. Williamson, 'A Neural Model of how Horizontal and Interlaminar Connections of Visual Cortex Develop into Adult Circuits that Carry Out Perceptual Grouping and Learning', *Cerebral Cortex*, vol. 11, no. 1, pp. 37–58, Jan. 2001, doi: 10/bn377j.

https://doi.org/10/bn377j

Summary of paper (maximum 100 words)

The methods to determine the self-organising kernels used in the LAMINART model were outlined.
The W kernels of L4 and the H and T kernels of L2/3 were learned using unstructured and structured inputs. The unstructured inputs simulated pre-natal learning whilst the structured inputs simulated post-natal learning.
A discussion on the effects of a balanced and unbalanced network on the formation of illusory contours was given.

How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)

The LAMINART model uses results from this paper in several kernels in L4 and L2/3.
The importance of balance between excitation and inhibition in the model was discussed.

What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)

The paper outlines how several kernels and parameters were determined for the LAMINART model.
The relationship between these parameters, the stability of the network and the formation of illusory contours was outlined.
Several differences between the equations of this paper and of LAMINART were not discussed, for example the function F is defined differently.
The kernels were only run at two orientations. Results for the H and W kernels were not reported numerically, they were only given graphically.
The T kernels had different results for each orientation. Using randomly configured rectangles as structured input, it could be expected that the kernel would be the same for both orientations. This discrepancy was not discussed.
The use of a circular window to restrict learning was not clearly justified.

Log Entry Creation Date: Wed Aug 05 2020 12:36:17 GMT+0100 (BST)

Processing real-world imagery with FACADE-based approaches
D. Hu, Z. Zhou, and Z. Wang, 'Processing real-world imagery with FACADE-based approaches', *Frontiers of Electrical and Electronic Engineering in China*, vol. 6, no. 1, pp. 120–136, Mar. 2011, doi: 10/c8h7q5.
https://doi.org/10/c8h7q5
Summary of paper (maximum 100 words)
The FACADE model was implemented and tested with real-world images. The model was found not to be very robust , in particular for images with noise. An imbalance of excitation and inhibition at bipolar cells was also reported. Oriented, anisotropic Gaussians were used to reduce the effect of noise. The bipole cell was also altered. Increases in robustness and efficiency were reported.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The FADADE model is a precursor to LAMINART. Implementations of LAMINART or FACADE would appear to be rare. Although FACADE is related to LAMINART, several differences exist making direct comparisons difficult.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
A short survey of Grossberg's visual models was given. Although briefly discussed, it is unclear why the authors chose to implement FACADE over LAMINART. Some dynamical equations were solved at equilibrium to calculate parts of the model. It is unclear how appropriate that may be.
Log Entry Creation Date: Wed Aug 05 2020 12:36:19 GMT+0100 (BST)

The complementary brain: unifying brain dynamics and modularity
S. Grossberg, 'The complementary brain: unifying brain dynamics and modularity', *Trends in Cognitive Sciences*, vol. 4, no. 6, pp. 233–246, Jun. 2000, doi: 10.1016/S1364-6613(00)01464-9.
https://doi.org/10.1016/S1364-6613(00)01464-9
Summary of paper (maximum 100 words)
Several complementary processing streams thought to exist in the brain were discussed. These parallel streams have complementary strengths and weaknesses. Boundary completion / surface filling-in is one such complementary pair. This was discussed in relation to the FACADE model. Several pathways from LAMINART were outlined. The role of the on-centre off-surround circuit in V1L4 and the direct LGN-V1L4 pathways were discussed.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
Several pathways in the LAMINART model were discussed, in particular LGN-(L6)-L4 including the on-centre off-surround, and V2-V1 feedback. Boundary completion / surface filling-in processing pair was also outlined.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
A broad discussion of complementary processes in the brain was given with the boundary completion / surface filling-in given as one example. Some pathways in the LAMINART model were broadly discussed. No mathematical details of these pathways were given.
Log Entry Creation Date: Wed Aug 05 2020 12:36:21 GMT+0100 (BST)

Consciousness CLEARS the mind
S. Grossberg, ‘Consciousness CLEARS the mind’, *Neural Networks*, vol. 20, no. 9, pp. 1040–1053, Nov. 2007, doi: 10.1016/j.neunet.2007.09.014.
https://doi.org/10.1016/j.neunet.2007.09.014
Summary of paper (maximum 100 words)
The interaction between feedforward and feedback pathways in LAMINART was outlined and it was suggested that similar balanced feedforward/feedback interactions operate at all levels of the brain linking consciousness, learning, expectation, attention resonance and synchrony (CLEARS). Mechanisms for these interactions, previously proposed in ART, were outlined. In particular subthresholding and resonance were discussed.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
Several properties of LAMINART were discussed. Some potential benefits of LAMINART model over Bayesian models were outlined. The relationship between a balanced network and perceptual grouping was summarized, in particular regarding L2/3’s bipole cells.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The paper related the dynamics of LAMINART with perceptual grouping. A broad summary of the possible advantages the model may have over other approaches was given. A broad discussion of how other processes in the brain may share mechanisms seen in LAMINART was given. Details of this prediction were not presented.
Log Entry Creation Date: Wed Aug 05 2020 12:36:25 GMT+0100 (BST)

How visual illusions illuminate complementary brain processes: illusory depth from brightness and apparent motion of illusory contours
S. Grossberg, ‘How visual illusions illuminate complementary brain processes: illusory depth from brightness and apparent motion of illusory contours’, *Frontiers in Human Neuroscience*, vol. 8, Oct. 2014, doi: 10/ggcg9t.
https://doi.org/10/ggcg9t
Summary of paper (maximum 100 words)
The complementary, parallel processing streams in the visual cortex were discussed. Two complementary processes were outlined along with the illusions which lead to their proposal. The apparent brightness seen in Kanizsa squares lead to the development of boundary completion / surface filling-in complementary pair. Boundary completion acts inwardly, is orientated and is insensitive to direction of contrast whilst surface filling in has opposite of these three properties. The FACADE and 3D LAMINART models were briefly discussed in relation to this complementary pair. The apparent motion of illusory contours and the complementary form and motion streams were also discussed.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The concept of a complementary pair of processing properties is relevant to the LAMINART model which has boundary compleantion / surface filling-in. This process was related to occluded objects and illusory contours.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
Boundary compilation / surface filling-in was broadly outlined and discussed in relation to Kanizsa square and occluded objects. Only FACADE and 3D LAMINART were discussed, not LAMINART. Little detail was given as to where these properties occur in these models.
Log Entry Creation Date: Wed Aug 05 2020 12:36:28 GMT+0100 (BST)

Visual brain and visual perception: how does the cortex do perceptual grouping?
S. Grossberg, E. Mingolla, and W. D. Ross, 'Visual brain and visual perception: how does the cortex do perceptual grouping?', *Trends in Neurosciences*, vol. 20, no. 3, pp. 106–111, Mar. 1997, doi: 10/c597d8.
https://doi.org/10/c597d8
Summary of paper (maximum 100 words)
Mechanisms and pathways were proposed to account for perceptual grouping in the mammalian brain. Illusions such as Kanizsa square lead to suggest that some areas of cortex have a long-range bipole property. Illusory contours with a large spanned distance invoke a response in V2 but not in V1. This suggests that V2 has longer-range interactions than V1. Neurophysiological results suggest that V1L4 receives input from LGN through an on-centre off-surround network. This normalizes V1L4 relative to contrast. There is a feedback pathway L4 through L3,5 back to L6 which may allow completion/filling in of discontinuities.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The pathways outlined led to the development of FACADE model, a precursor to LAMINART. The paper gave a broad outline of these pathways, in particular the role these pathways played in forming illusory contours was discussed.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
A broad outline of the roles pathways play in forming illusory contours was given but no mathematical details were offered.
Log Entry Creation Date: Wed Aug 05 2020 13:01:50 GMT+0100 (BST)

Neural dynamics of grouping and segmentation explain properties of visual crowding
G. Francis, M. Manassi, and M. H. Herzog, ‘Neural dynamics of grouping and segmentation explain properties of visual crowding’, *Psychological Review*, vol. 124, no. 4, pp. 483–504, Jul. 2017, doi: 10/gbf8r8.
https://doi.org/10/gbf8r8
Summary of paper (maximum 100 words)
The LAMINART model was adapted to simulate visual crowding. Visual crowding is a phenomena seen in neuroscience relating to decreased object recognition where there are more objects in the visual field. The model was run as a spiking neural network using exact digital simulation. It was concluded that boundary grouping may have an important role in visual crowding. LAMINART’s ability to simulate boundary grouping gave it an advantage over other methods for simulating visual crowding.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
Although the motivation and approach here differ from the project’s research question, recent implements of LAMINART appear to be rare.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The simulation’s code was made open source. Only two orientations were used during simulation. The model was run on a distributed computer using exact digital simulation rather than numerical approximation.
Log Entry Creation Date: Mon Aug 17 2020 17:07:56 GMT+0100 (BST)

Beyond Bouma's window: How to explain global aspects of crowding?
A. Doerig, A. Bornet, R. Rosenholtz, G. Francis, A. M. Clarke, and M. H. Herzog, 'Beyond Bouma's window: How to explain global aspects of crowding?', *PLOS Computational Biology*, vol. 15, no. 5, p. e1006580, May 2019, doi: 10/ggp7nj.
https://doi.org/10/ggp7nj
Summary of paper (maximum 100 words)
Work on LAMINART and visual crowding done by Francas et al. was extended. The model was compared with several other methods to simulate visual crowding. The model was adapted to tie together objects linked with illusory contours. Similar to Francas et al.'s implemented, exact digital simulation was used. LAMINART performed best for the visual crowding task investigated.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
Previously, LAMINART was simulated with two orientations. Here, the number of orientations was increased to eight.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
It is unclear from the paper and code how the number of orientations was increased to eight. Specifically, it was unclear how the number of orientations of the model's self-organised kernels was increased.
Log Entry Creation Date: Wed Aug 05 2020 13:09:14 GMT+0100 (BST)

Exact digital simulation of time-invariant linear systems with applications to neuronal modeling
S. Rotter and M. Diesmann, ‘Exact digital simulation of time-invariant linear systems with applications to neuronal modeling’, *Biological Cybernetics*, vol. 81, no. 5–6, pp. 381–402, Nov. 1999, doi: 10/fbdqd7.
https://doi.org/10/fbdqd7
Summary of paper (maximum 100 words)
Exact digital simulation, method for solving linear time-invariant systems of ODEs representing neural networks, was presented. The method may also be used with some non-linear neural networks by treating threshold functions as spikes. The method was tested against several numerical methods. Issues with step-size and stiffness can be avoided with exact digital simulation, however it was found to be slightly less efficient than most of the numerical methods tested.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The method presented offers a different approach for solving neural models involving ODEs. This approach was used with LAMINART by Francis and Doerig. The appendix contains a summary of numerical methods for solving systems similar to LAMINART.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
Issues with step-size and stiffness can be avoided with exact digital simulation. These issues may be managed for numerical methods with the use of adaptive solvers. Some numerical methods are more efficient.
Log Entry Creation Date: Wed Aug 05 2020 13:09:16 GMT+0100 (BST)

DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia
C. Rackauckas and Q. Nie, 'DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia', *Journal of Open Research Software*, vol. 5, no. 1, Art. no. 1, May 2017, doi: 10/ggfnfj.
https://doi.org/10/ggfnfj
Summary of paper (maximum 100 words)
DifferentialEquations.jl, a Julia package for numerically solving differential equations, was outlined. The package may be used to solve several types of differential equations including ODEs. Some features discussed included an extensive suite of solvers, parallelism and symbolic calculation of Jacabians. The package was designed for different solvers to use a common API. This API, along with several solvers were discussed.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
LAMINART model consists of a system of ODEs. DifferentialEquations.jl appears to be a suitable choice. It has a large suite of solvers, integrates well with larger ecosystem and is possible to parallelise with much greater ease then MATLAB or C++ code.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The package has a comprehensive suite of solvers and integrates well with the Julia ecosystem. It benchmarks faster than Matlab while being more usable than C++. The ability to parallelise through the use of CuArrays makes parallelisation much easier. The package is open source and is well maintained and documented. It would appear that other Julia packages which may be required for the project are not as mature or well documented as their counterparts in other languages, for example JuliaImages for image processing and Flux for convolution.
Log Entry Creation Date: Wed Aug 05 2020 13:09:19 GMT+0100 (BST)

Automatic Selection of Methods for Solving Stiff and Nonstiff Systems of Ordinary Differential Equations
L. Petzold, ‘Automatic Selection of Methods for Solving Stiff and Nonstiff Systems of Ordinary Differential Equations’, *SIAM Journal on Scientific and Statistical Computing*, vol. 4, no. 1, pp. 136–148, Mar. 1983, doi: 10/bk72v5.
https://doi.org/10/bk72v5
Summary of paper (maximum 100 words)
LSODE, a method for automatically determining switching ODE solvers was outlined. LSODE determines whether a problem is non-stiff or stiff at the end of each step while solving. It is able to switch to more effective solver for that interval. The stiffness of a problem does not need to be estimated or determined before solving. It also allows for the use of faster non-stiff solvers where possible and switches to slower stiff solvers when required.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
LSODE would appear to be suitable for solving LAMINART’s system of ODEs. It has been used by Grossberg. DifferentialEquations.jl also suggested that it may be an appropriate choice for problems with >1000 ODEs.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
LSODE is a well established method for solving ODEs. It automatically detects whether a problem is stiff or non-stiff while solving and automatically switches to a more effective solver. It does this with little overhead. Parallelisation of the method is hard as it uses implicit solvers. DifferentialEquations.jl does not currently have a parallelised version.
Log Entry Creation Date: Wed Aug 05 2020 13:09:30 GMT+0100 (BST)

Solving ordinary differential equations on GPUs
K. Ahnert, D. Demidov, and M. Mulansky, ‘Solving ordinary differential equations on GPUs’, 2014, pp. 125–157. doi: 10.1007/978-3-319-06548-9_7.
https://doi.org/10.1007/978-3-319-06548-9_7
Summary of paper (maximum 100 words)
A method to parallelise Runge-Kutta type explicit ODE solvers to process on GPUs was presented. Vector-vector addition and scalar-vector multiplication are well suited for parallelisation. A Runge-Kutta method was developed with these operations parallelised. When benchmarked against solving on a CPU, the parallelised version performed magnitudes of time faster. The problem type and size varied the difference.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
Part of the project’s research question was to implement LAMINART on a GPU. The approach in the paper addresses how ODEs may be solved with the use of a GPU. The approach was implemented in DifferenicalEquations.jl’s OrdinaryDiffEq.jl solvers.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The parallelised version of the method performed magnitudes faster than solving on a CPU. The package’s modular design allows scalability and the use of both CUDA and OpenCL as backends. The method is not effective with implicit methods.
Log Entry Creation Date: Wed Aug 05 2020 13:09:31 GMT+0100 (BST)

Rapid software prototyping for heterogeneous and distributed platforms
T. Besard, V. Churavy, A. Edelman, and B. D. Sutter, ‘Rapid software prototyping for heterogeneous and distributed platforms’, *Advances in Engineering Software*, vol. 132, pp. 29–46, Jun. 2019, doi: 10/gf7rfm.
https://doi.org/10/gf7rfm
Summary of paper (maximum 100 words)
A collection of Julia packages to integrate with CUDA at various levels was presented. CuArrays.jl provides high-level access for performing operations on arrays on a GPU. CUDANative.jl provides access at a lower level, allowing kernels for GPU to be written in Julia. DistributedArrays.jl allows for processing on distributed systems and does not have the same level of maturity a s CuArrays.jl. The use of ‘dot syntax’ with CuArrays.jl allows for the broadcast of functions and the automatic use of appropriate CUDA libraries such as cuBLAS calls.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
CuArrays appear to offer an appropriate, high-level method to parallelise LAMINART’s ODE system. Some solvers from DifferentialEquations.jl as well as NNlib convolution integrate well CuArrays.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
CuArrays.jl integrates well with Julia’s ecosystem. The package makes it possible to parallelise code easily with the use of abstract arrays and ‘dot syntax’. Appropriate CUDA libraries such as cuBLAS may be used automatically.
Log Entry Creation Date: Wed Aug 05 2020 13:09:33 GMT+0100 (BST)

CUDA C Programming Guide
‘CUDA C Programming Guide’, NVIDIA, White Paper PG-02829-001_v8.0, Jun. 2017. [Online]. Available: https://docs.nvidia.com/cuda/archive/8.0/pdf/CUDA_C_Programming_Guide.pdf.
https://docs.nvidia.com/cuda/archive/8.0/pdf/CUDA_C_Programming_Guide.pdf
Summary of paper (maximum 100 words)
The use of CUDA to perform general purpose calculations on a GPU was presented. CUDA Developer’s Toolkit 8.0GA2, various CUDA libraries including cuBLAS and the use of CUDA on specific NVIDIA GPUs were documented comprehensively.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
CUDA is intended to be used to parallelise the LAMINART model. The School has an NVIDIA Tesla K40C GPU with CUDA Toolkit 8.0GA2. The documentation provides detailed information on how to achieve this.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
The School’s GPU has a compute capacity of 3.5, meaning that half-precision operations are not permitted. It was reported that the GPU can perform 192 single precision floating-point add, multiply, multiply-add operations per clock cycle per multiprocessor. This is opposed to 160 32-bit integer add operations per clock cycle per multiprocessor and 30 32-bit integer multiply operations per clock cycle per multiprocessor. (Section 5.4.1)
Log Entry Creation Date: Mon Aug 17 2020 12:22:11 GMT+0100 (BST)

Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision
N. V. K. Medathati, H. Neumann, G. S. Masson, and P. Kornprobst, 'Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision', *Computer Vision and Image Understanding*, vol. 150, pp. 1–30, Sep. 2016, doi: 10/f8w8cj.
https://doi.org/10/f8w8cj
Summary of paper (maximum 100 words)
A survey of computational models of biological vision relevant to computer vision was presented. Three computer vision tasks were discussed: image sensing, segmentation and optical flow. Biological models contrasted against computer vision methods for each of these tasks. Previously, for computer vision, segmentation involved either edge detection or clustering. These were than have combined. Recently, CNNs and segmentation networks have been successful at segmentation. The use of feedback was discussed.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The possible advantages FACADE may have over current computer vision methods for segmentation were discussed.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
An extensive survey of biological models of vision was presented for computer vision engineers. Several tasks which the human visual system can do but which computer vision finds challenging were identified. Approaches taken by neuroscientists to model were outlined. Large advancements have been made in computer vision in the four years since this paper, for example in recurrent networks.
Log Entry Creation Date: Tue Aug 18 2020 15:02:15 GMT+0100 (BST)

Appendix D

Design and implementation

D.I. Equations of the LAMINART Model

Looking at the appendix of [1], the following equations are used to describe the LAMINART model.

A. Retinal cells

Outputs from the retinal cells, (u_{ij}^{+}, u_{ij}^{-}), are given as

$$u_{ij}^{+} = I_{pq} - \sum_{pq} G_{pq}(i,j,\sigma) I_{pq} \tag{D.1}$$

$$u_{ij}^{-} = -I_{pq} + \sum_{pq} G_{pq}(i,j,\sigma) I_{pq} \tag{D.2}$$

where I_{pq} is the input and $G_{pq}(i,j,\sigma)$ is a two dimensional Gaussian:[1]

$$G_{pq}(i,j,\sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2\sigma^2}\left((p-i)^2 + (q-j)^2\right)\right) \tag{D.3}$$

Equation (D.2) may be rewritten in terms of (D.1):

$$u_{ij}^{-} = -u_{ij}^{+} \tag{D.4}$$

B. LGN layer

The LGN layer, with outputs v_{ij}^{+} and v_{ij}^{-}, is given by the differential equations

$$\frac{1}{\delta_v}\frac{\mathrm{d}}{\mathrm{d}t} v_{ij}^{\pm} = -v_{ij}^{\pm} + (1 - v_{ij}^{\pm})[u_{ij}^{\pm}]^{+}(1 + A_{ij}) - (1 + v_{ij}^{\pm}) B_{ij} \tag{D.5}$$

where A and B are feedback terms from L6 , δ_v is a parameter and $[x]^{+}$is a half-wave rectifier:

$$[x]^{+} = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

The feedback terms A and B are given as

$$A_{ij} = C_1 \sum_{k} x_{ijk} \tag{D.6}$$

$$B_{ij} = C_2 \sum_{pqk} Gpq(i,j,\sigma_1) x_{ijk} \tag{D.7}$$

where x is the output of L6 and C_1 and C_2 are scaler parameters.
B may be rewritten in terms of A:

$$B_{ij} = \frac{C_2}{C_1} \sum_{pq} Gpq(i,j,\sigma_1) A_{ij} \tag{D.8}$$

C. LGN *to* L6 / L4

The LGN 's on-centre (v_{pq}^{+}) and off-surround (v_{pq}^{-}) outputs go through the following before being used as the inputs into L6 and L4 (C_{ijk})

An oriented difference of offset Gaussians, $D_{pqij}^{(k)}$, is given as

$$D_{pqij}^{(k)} = G_{pq}(i - \delta\cos\theta, j - \delta\sin\theta, \sigma) - G_{pq}(i + \delta\cos\theta, j + \delta\sin\theta, \sigma) \tag{D.9}$$

and

$$\delta = \frac{\sigma}{2} \tag{D.10}$$

$$\theta = \frac{\pi(k-1)}{K} \tag{D.11}$$

[1] presuming that $(p-1)^2 + (q-i)^2$ is a typo in [1]

where K is total orientations, $k \in [1, 2K]$. A sketch is shown in Fig. D.1. The DOOG is used to get an "On subregion", R_{ijk} and an "Off subregion" L_{ijk}.

$$R_{ijk} = \sum_{pq} \left([v^{+}_{pq}]^{+} - [v^{-}_{pq}]^{+}\right)[D^{(k)}_{pqij}]^{+} \tag{D.12}$$

$$L_{ijk} = \sum_{pq} \left([v^{-}_{pq}]^{+} - [v^{+}_{pq}]^{+}\right)[-D^{(k)}_{pqij}]^{+} \tag{D.13}$$

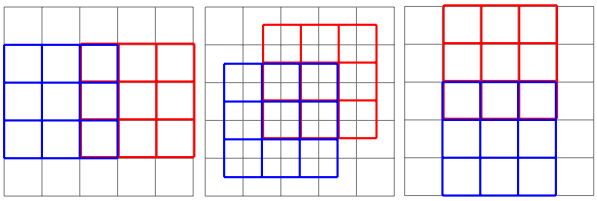
The subregions are then summed and their difference is subtracted. This is then half-wave rectified to get S_{ijk}:

$$S_{ijk} = \gamma[R_{ijk} + L_{ijk} - |R_{ijk} - L_{ijk}|]^{+} \tag{D.14}$$

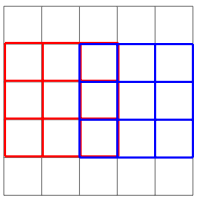
where γ is a parameter.
Figure D.3 shows $D(x, y\sigma)$ at four orientations. Figure D.4 shows $[D_{pqij}]^{+}$ and $-[-D_{pqij}]^{+}$ along with $[D_{pqij}]^{+} + [-D_{pqij}]^{+}$ and $[D_{pqij}]^{+} - [-D_{pqij}]^{+}$.

To simplify the model, opposite polarities along the same orientations are pooled to get C_{ijk} which is used as the inputs to L6 and L4 :

$$C_{ijk} = S_{ijk} + S_{ij(k+K)} \tag{D.15}$$



(a) 0° (b) 45° (c) 90°



(d) 180°

Figure D.1. Orientated Difference of Offset Gaussian filters. The filter equals the filtered red region less the filtered blue region with four orientations shown. Filter $(d) \equiv -$(Filter (a))

Putting (D.3) into (D.9) gets:

$$D^{(k)}_{pqij} = \frac{1}{2\pi\sigma^2}\left(\exp\left(-\frac{1}{2\sigma^2}\left(\big(p - (i - \delta\cos\theta)\big)^2 + \big(p - (i - \delta\sin\theta)\big)^2\right)\right) - \exp\left(-\frac{1}{2\sigma^2}\left(\big(q - (i + \delta\cos\theta)\big)^2 + \big(q - (i + \delta\sin\theta)\big)^2\right)\right)\right) \tag{D.16}$$

Expanding out the terms

$$\begin{aligned}(p - (i - \delta\cos\theta))^2 &= p^2 - 2pi + 2p\delta\cos\theta + i^2 - 2i\delta\cos\theta + \delta^2\cos^2\theta \\ &= (p - i)^2 + 2\delta\cos\theta(p - i) + \delta^2\cos^2\theta\end{aligned} \tag{D.17}$$

and

$$\begin{aligned}(p - (i + \delta\cos\theta))^2 &= p^2 - 2pi - 2p\delta\cos\theta + i^2 + 2i\delta\cos\theta + \delta^2\cos^2\theta \\ &= (p - i)^2 - 2\delta\cos\theta(p - i) + \delta^2\cos^2\theta\end{aligned} \tag{D.18}$$

and doing similar with sine terms gets

$$D_{pqij}^{(k)} = \frac{1}{2\pi\sigma^2}\Bigg(\exp\Bigg(-\frac{1}{2\sigma^2}\Big((p-i)^2+(q-j)^2+2\delta\big(\cos\theta(p-i)+\sin\theta(q-j)\big)+\delta^2\big(\cos^2\theta+\sin^2\theta\big)\Big)\Bigg)$$
$$-\exp\Bigg(-\frac{1}{2\sigma^2}\Big((p-i)^2+(q-j)^2-2\delta\big(\cos\theta(p-i)+\sin\theta(q-j)\big)+\delta^2(\cos^2\theta+\sin^2\theta)\Big)\Bigg)\Bigg)$$
$$= G_{pq}(i,j,\sigma)\exp\Bigg(-\frac{\delta^2}{2\sigma^2}\Bigg)\Bigg(\exp\Bigg(-\frac{1}{2\sigma}2\delta\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Bigg)$$
$$-\exp\Bigg(-\frac{1}{2\sigma}\Big(-2\delta\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)\Bigg)\Bigg) \tag{D.19}$$

Replacing δ with (D.10):

$$D_{pqij}^{(k)} = G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}\Bigg(\exp\Bigg(-\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Bigg)$$
$$-\exp\Bigg(\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta\,(q-j)\,\big)\Bigg)\Bigg) \tag{D.20}$$

Equation (D.12) and (D.13) may be rewritten as

$$R_{ijk} = \sum_{pq}\Big(\big[v_{pq}^{+}\big]^{+}-\big[v_{pq}^{-}\big]^{+}\Big)G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}\Big($$
$$\Big[\exp\Big(-\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)-\exp\Big(\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)\Big]^{+}\Big) \tag{D.21}$$

and

$$L_{ijk} = \sum_{pq}\Bigg(\big[v_{pq}^{+}\big]^{+}-\big[v_{pq}^{-}\big]^{+}\Big)G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}\Big($$
$$-\Big[\Big(\exp\Big(-\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)-\exp\Big(\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)\Big)\Big]^{+}\Bigg) \tag{D.22}$$

Defining $d_{pq}(i,j,\sigma,\theta)$ as

$$d_{pq}(i,j,\sigma,\theta) = \exp\Big(-\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big)-\exp\Big(\frac{1}{2\sigma}\big(\cos\theta(p-i)+\sin\theta(q-j)\big)\Big) \tag{D.23}$$

then

$$D_{pqij}^{(k)} = G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}d_{pq}(i,j,\sigma,\theta)$$

Given that

$$\Big[G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}\Big]^{+} = G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8} \tag{D.24}$$

then

$$\Big[\pm D_{pqij}^{(k)}\Big]^{+} = G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8}[\pm d_{pq}(i,j,\sigma,\theta)]^{+} \tag{D.25}$$

Defining $V_{pq}(i,j,\sigma)$ as

$$V_{pq}(i,j,\sigma) = \Big(\big[v_{pq}^{+}\big]^{+}-\big[v_{pq}^{-}\big]^{+}\Big)G_{pq}(i,j,\sigma)\mathrm{e}^{-1/8} \tag{D.26}$$

R_{ijk} and L_{ijk} may be simplified further:

$$R_{ijk} = \sum_{pq}V_{pq}(i,j,\sigma)[d_{pq}(i,j,\sigma,\theta)]^{+} \tag{D.27}$$

$$L_{ijk} = -\sum_{pq}V_{pq}(i,j,\sigma)[-d_{pq}(i,j,\sigma,\theta)]^{+} \tag{D.28}$$

Putting into (D.14):

$$S_{ijk} = \gamma\Bigg[\sum_{pq} V_{pq}(i,j,\sigma)\big([d_{pq}(i,j,\sigma,\theta)]^{+} - [-d_{pq}(i,j,\sigma,\theta)]^{+}\big) - \Bigg|\sum_{pq} V_{pq}(i,j,\sigma)\big([d_{pq}(i,j,\sigma,\theta)]^{+} + [-d_{pq}(i,j,\sigma,\theta)]^{+}\big)\Bigg|\Bigg]^{+} \tag{D.29}$$

Using (D.11), then:

$$\theta_{k+K} = \frac{\pi(k+K-1)}{K} = \theta_k + \pi \tag{D.30}$$

meaning

$$\begin{aligned} d_{pq}(i,j,\sigma,\theta_{(k+K)}) &= \exp\Big(-\frac{1}{2\sigma}\big(-\cos\theta_k(p-i) - \sin\theta_k(q-j)\big)\Big) \\ &\quad - \exp\Big(\frac{1}{2\sigma}\big(-\cos\theta_k(p-i) - \sin\theta_k(q-j)\big)\Big) \\ &= -d_{pq}(i,j,\sigma,\theta_k) \end{aligned} \tag{D.31}$$

It follows that

$$[\pm d_{pq}(i,j,\sigma,\theta_k)]^{+} = \big[\mp d_{pq}(i,j,\sigma,\theta_{(k+K)})\big]^{+} \tag{D.32}$$

Putting into (D.15)

$$\begin{aligned} C_{ijk} = \gamma\Bigg(&\Bigg[\sum_{pq} V_{pq}(i,j,\sigma)\big([d_{pq}(i,j,\sigma,\theta)]^{+} - [-d_{pq}(i,j,\sigma,\theta)]^{+}\big) \\ &- \Bigg|\sum_{pq}(V_{pq}(i,j,\sigma)\big([d_{pq}(i,j,\sigma,\theta)]^{+} + [-d_{pq}(i,j,\sigma,\theta)]^{+}\big)\Bigg|\Bigg]^{+} \\ &+ \Bigg[\sum_{pq} V_{pq}(i,j,\sigma)\big([-d_{pq}(i,j,\sigma,\theta)]^{+} - [d_{pq}(i,j,\sigma,\theta)]^{+}\big) \\ &- \Bigg|\sum_{pq}(V_{pq}(i,j,\sigma)\big([-d_{pq}(i,j,\sigma,\theta)]^{+} + [d_{pq}(i,j,\sigma,\theta)]^{+}\big)\Bigg|\Bigg]^{+}\Bigg) \end{aligned} \tag{D.33}$$

Since

$$\begin{aligned} d_{pq}(i,j,\sigma,\theta) = [d_{pq}(i,j,\sigma,\theta)]^{+} - [-d_{pq}(i,j,\sigma,\theta)]^{+} \\ \Bigg(= \exp\Big(-\frac{1}{2\sigma}\big(\cos\theta(p-i) + \sin\theta(q-j)\big)\Big) \\ - \exp\Big(\frac{1}{2\sigma}\big(\cos\theta(p-i) + \sin\theta\,(q-j)\,\big)\Big)\Big)\Bigg) \end{aligned} \tag{D.34}$$

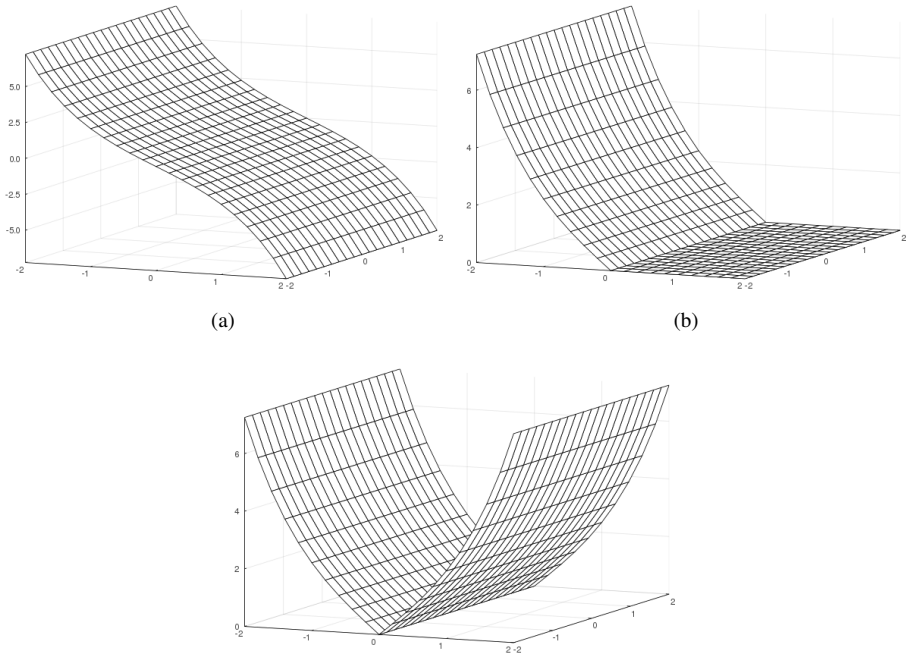
and by defining $b_{pq}(i,j,\sigma,\theta)$ as

$$\begin{aligned} b_{pq}(i,j,\sigma,\theta) = [d_{pq}(i,j,\sigma,\theta)]^{+} + [-d_{pq}(i,j,\sigma,\theta)]^{+} \\ \Bigg(= \exp\Big(-\frac{1}{2\sigma}\big(\cos\theta(p-i) + \sin\theta(q-j)\big)\Big) \\ + \exp\Big(\frac{1}{2\sigma}\big(\cos\theta(p-i) + \sin\theta\,(q-j)\,\big)\Big)\Big)\Bigg) \end{aligned} \tag{D.35}$$

and Q_{ijk} and P_{ijk} as

$$Q_{ijk} = \sum_{pq} V_{pq}(i,j,\sigma) d_{pq}(i,j,\sigma,\theta) \tag{D.36}$$

$$P_{ijk} = \sum_{pq} V_{pq}(i,j,\sigma) b_{pq}(i,j,\sigma,\theta) \tag{D.37}$$



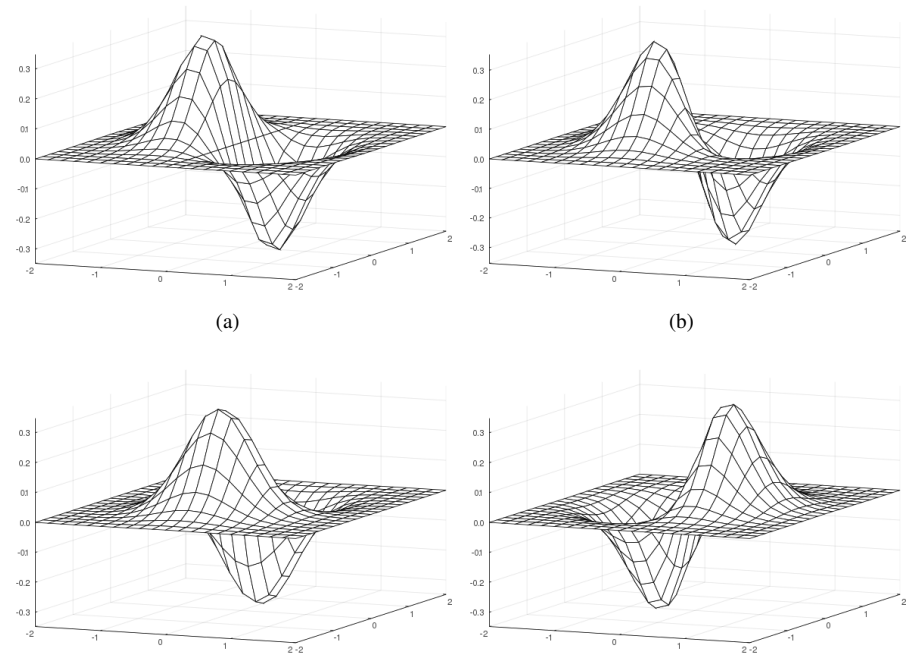
(a) (b)

(c)

Figure D.2. (a), $d(x, y)$, (b) $[d(x, y)]^+$ and (c) $b(x, y)$, all with $\sigma = 0.5$ and at the same orientation.

(D.33) can be rewritten as

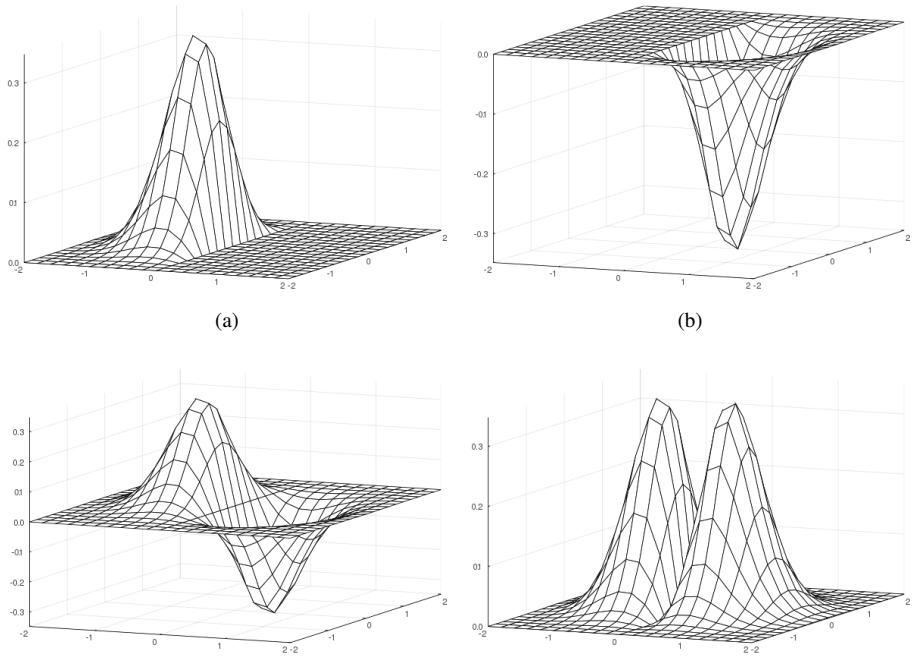
$$C_{ijk} = \gamma\Big([Q_{ijk} - |P_{ijk}|]^+ + [-Q_{ijk} - |P_{ijk}|]^+\Big) \tag{D.38}$$



(a) (b)

(c) (d)

Figure D.3. $D(x, y, \sigma = 0.2)$ and total orientations, $K = 12$ at orientations (a) $k = 1$ (0°), (b) $k = 4$ (45°), (c) $k = 7$ (90°), and (d) $k = 13$ (180°).



(a) (b)

(c) (d)

Figure D.4. (a) $[D(x,y)]^+$, (b) $-[-D(x,y)]^+$, (c) $[D(x,y)]^+ + [-D(x,y)]^+$ and (d) $[D(x,y)]^+ - [-D(x,y)]^+$, all with $\sigma = 0.5$ and at the same orientation.

D. L6 cells

Layer 6, with output x_{ijk} is given by the differential equation

$$\frac{1}{\delta_c}\frac{\mathrm{d}}{\mathrm{d}t}x_{ijk} = -x_{ijk} + (1 - x_{ijk})\big(\alpha C_{ijk} + \phi F(z_{ijk}, \Gamma) + V_{21}x_{ijk}^{V2} + att\big) \tag{D.39}$$

where x_{ijk}^{V2} is the output of V2 L6 , $att.$ is a two dimensional Gaussian used in some simulations involving attention and δ_c, α, ϕ and Γ are scalar parameters.
$F()$ is a thresholding signal function:

$$\begin{aligned} F(x,\Gamma) &= max(x - \Gamma, 0) \\ &= \begin{cases} x - \Gamma, & x > \Gamma \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{D.40}$$

In [2] the thresholding function is given by

$$F(x,\Gamma) = \begin{cases} x, & x > \Gamma \\ 0, & \text{otherwise} \end{cases} \tag{D.41}$$

The output of (D.40) is $x - \Gamma$ when the input is above the threshold but this version avoids the discontinuity which occurs in (D.41) if the input goes past the threshold. Equation (D.40) was chosen.

E. L4 cells

Layer 4 is given by two differential equations. The output of the layer, y_{ijk} is given by

$$\frac{1}{\delta_c}\frac{\mathrm{d}}{\mathrm{d}t}y_{ijk} = -y_{ijk} + (1 - y_{ijk})\big(C_{ijk} + \eta^+ x_{ijk}\big) - \big(y_{ijk} + 1\big)f\big(\sum_{pqr} W^+_{pqrijk} m_{pqr}\big) \tag{D.42}$$

where the inhibitory term m is given by the second differential equation

$$\frac{1}{\delta_m}\frac{\mathrm{d}}{\mathrm{d}t}m_{ijk} = -m_{ijk} + \eta^- x_{ijk} - m_{ijk} f\big(\sum_{pqr} W^-_{pqrijk} m_{pqr}\big) \tag{D.43}$$

The function $f()$ is a Sigmund given as

$$f(x) = \mu \frac{x^n}{v^n + x^n} \tag{D.44}$$

δ_m, η^+, η^-, μ and n are scalar parameters.
W^+_{pqrijk} is a self-organised kernel. The learning process is described in [2]. The values used by [1] were from [2].
W^-_{pqrijk} is a linearly scaled version of W^+_{pqrijk}.
The $\eta^- x_{ijk}$ term of m is quadratic in [2].

F. L2/3 cells

$$\frac{1}{\delta_z}\frac{\mathrm{d}}{\mathrm{d}t} z_{ijk} = -z_{ijk} + (1 - z_{ijk})\big(\gamma[y_{ijk}]^+ + \sum_{pqr}\big(H_{pqrijk}F(z_{pqr},\Gamma) + a^{23}_{\text{excit}} att - (z_{ijk} + \phi)T^+_{rk}s_{ijr}\big)\big) \tag{D.45}$$

$$\frac{1}{\delta_s}\frac{\mathrm{d}}{\mathrm{d}t} s_{ijk} = -s_{ijk} + \sum_{pqr}\big(H_{pqrijk}F(z_{pqr},\Gamma) + a^{23}_{\text{inhib}} att - s_{ijk}T^-_{rk}s_{ijr}\big) \tag{D.46}$$

δ_z, δ_s, γ, a^{23}_{excit} and a^{23}_{inhib} are scalar parameters.
H is a narrow, long kernel. T is a
$\sum_{pqr} H_{pqrijk}F(z_{pqr},\Gamma)$ is common to both (D.45) and (D.46) allowing the equations to be rewritten as

$$\frac{1}{\delta_z}\frac{\mathrm{d}}{\mathrm{d}t} z_{ijk} = -z_{ijk} + (1 - z_{ijk})\big(\gamma[y_{ijk}]^+ + h_{ijk} + \sum_{pqr} + a^{23}_{\text{excit}} att - (z_{ijk} + \phi)T^+_{rk}s_{ijr}\big) \tag{D.47}$$

and

$$\frac{1}{\delta_s}\frac{\mathrm{d}}{\mathrm{d}t} s_{ijk} = -s_{ijk} + h_{ijk} + \sum_{pqr}\big(a^{23}_{\text{inhib}} att - s_{ijk}T^-_{rk}s_{ijr}\big) \tag{D.48}$$

where

$$h_{ijk} = \sum_{pqr} H_{pqrijk}F(z_{pqr},\Gamma) \tag{D.49}$$

G. V2 cortex

Layers 6, 4 and 2/3 of V2 are identical to V1 except for three modifications.
V2 L6 is given by

$$\frac{1}{\delta_c}\frac{\mathrm{d}}{\mathrm{d}t} x^{V2}_{ijk} = -x^{V2}_{ijk} + (1 - x^{V2}_{ijk})\big(V^6_{12}F(z_{ijk},\Gamma) + \phi F(z^{V2}_{ijk},\Gamma)\big) \tag{D.50}$$

The excitory part of V2 L4 is given by

$$\frac{1}{\delta_c}\frac{\mathrm{d}}{\mathrm{d}t} y^{V2}_{ijk} = -y^{V2}_{ijk} + (1 - y^{V2}_{ijk})\big(V^4_{12}F(z_{ijk},\Gamma) + \eta^+ x^{V2}_{ijk}\big) - \big(y^{V2}_{ijk} + 1\big) f\big(\sum_{pqr} W^+_{pqrijk} m^{V2}_{pqr}\big) \tag{D.51}$$

where V^6_{12} and V^4_{12} are scalar parameters.
V2 L2/3 uses the kernel H^{V2} instead of H. H^{V2} has a longer range then H.

D.II. Implementation

A. Equations

1) Retinal layer: A Gaussian of the input was subtracted from the input to find u^+_{ij}. The off surround output $u^-_{ij} = -u^+_{ij}$.

Algorithm D.1 Retina cells

function FUNC_I_U($I, kern\_\sigma_1$) ▷ $I \in \mathbb{R}^{(i \times j)}$
    $u^+ - I * kern\_\sigma_1$
    return u^+
end function

2) LGN layer: Both dv^+ and dv^- used the same function, outlined in Algorithm D.2. The feedback from L6 has been outlined in Algorithm D.3.

Algorithm D.2 LGN

function FUNC_DV($v^{\pm}, u, x\_lgn, \delta_v, C_1, C_2, kern\_\sigma_1$) ▷ $v^{\pm}, u, x\_lgn \in \mathbb{R}^{(i \times j)}$
    $dv^{\pm} \leftarrow \delta_v \times (-v + ((1 - v) \times \text{relu.}(u) \times (1 + C_1 \times x\_lgn)) - ((1 + v) \times C_2 \times (x\_lgn * kern\_\sigma_1)))$ ▷ relu. is elementwise relu

    return $dv^{\pm}$
end function

Algorithm D.3 Feedback from L6 → LGN

function FUNC_X_LGN(x) ▷ $x \in \mathbb{R}^{(i \times j \times K)}$
    $x_lgn \leftarrow$CREATE(Array(zeros)$^{(i \times j \times K)}$)
    for k in $[1, K]$ **do**
        $x\_lgn \leftarrow x\_lgn + x[:, :, k]$
    end for
    return x_lgn
end function

3) LGN → L6 / L4: An implementation is outlined in Algorithm D.4 and D.5.
The size of the kernels, l, was left at $4\sigma + 1$, the default for a Gaussian filter in Julia's image filtering library. The affect of increasing the kernel size was investigated.
Four one dimensional kernels were first created. The first pair, $kern\_d\_h\_p$ and $kern\_d\_v\_p$, corresponded to the positive term of (D.23) whilst the second pair, $kern\_d\_h\_m$ and $kern\_d\_v\_m$, corresponded to the negative term. A pair of two dimensional arrays were created by broadcast multiplying each pair of kernels respectively. Finally, $kern\_Q$ and $kern\_P$ were calculated by subtracting and summing the two dimensional kernels respectively. These kernels were then used in Algorithm D.5 to calculate C at K orientations.

Algorithm D.4 LGN → L6 / L4 kernels

function FUNC_KERN_C(σ, θ, l) ▷ Ensure l is odd
    if l not provided **then**
        $l = 4\sigma + 1$ ▷ default l for Gaussian in JuliaImaging
    end if
    for w in $[(-l - 1/2), (l - 1/2)]$ **do**
        $x \leftarrow$ ABS(w)
        $kern\_d\_p\_h_w \leftarrow \exp\{-x \cos\theta / 2\sigma\}$
        $kern\_d\_p\_v_w \leftarrow \exp\{-x \sin\theta / 2\sigma\}$
        $kern\_d\_m\_h_w \leftarrow \exp\{x \cos\theta / 2\sigma\}$
        $kern\_d\_m\_v_w \leftarrow \exp\{x \sin\theta / 2\sigma\}$
    end for
    $kern\_d\_p \leftarrow kern\_d\_p\_h(kern\_d\_p\_v)^T$
    $kern\_d\_m \leftarrow kern\_d\_m\_h(kern\_d\_m\_v)^T$
    $kern\_Q \leftarrow kern\_d\_p - kern\_d\_m$
    $kern\_P \leftarrow kern\_d\_p + kern\_d\_m$
    return $kern\_Q, kern\_P$
end function

Algorithm D.5 LGN → L6 / L4

function FUNC_V_C($v^+, v^-, \sigma, K, \gamma$) ▷ $v^+, v^- \in \mathbb{R}^{(i \times j)}$
  $V \leftarrow \exp\{-1/s\}(\text{relu}(v^+) - \text{relu}(v^-)) * G(\sigma)$
  $A \leftarrow$CREATE(Array$^{(i \times j \times K)}$)
  $B \leftarrow$COPY(A)
  for k in $[1, K]$ **do**
    $\theta \leftarrow \pi(k-1)/K$
    $A_k \leftarrow V * kern\_Q(\sigma, \theta)$
    $B_k \leftarrow \text{abs}(V * kern\_P(\sigma, \theta))$
    $C_k \leftarrow \gamma(\text{relu}(Q_k - P_k) + \text{relu}(-Q_k - P_k))$
  end for
  return C
end function

The Julia library *JuliaImages* was used to load in a test image. This was then converted to an array of type *Float64*. Parameters were those used in [1] which where $\sigma_1 = 1$ (for the retinal layer), $\sigma_2 = 0.5$ (for the LGN layer)[2], and $\gamma = 10$. Twelve orientations were looked at. JuliaImages' function `imfilter` was first used to correlate the image with the Gaussian and the custom kernels. Bordering was dealt with using the function's option `"circular"`. Zero padding was used in later implementations to test against NNlib.conv. This was due to a restriction of bordering options with NNlib.

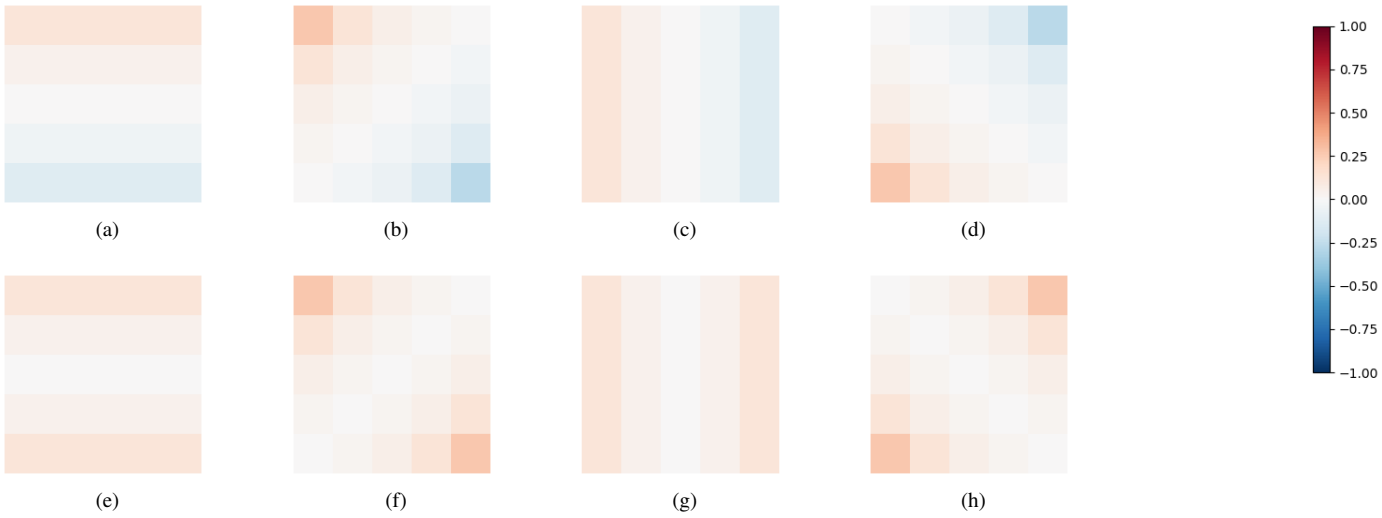


Figure D.5. Plots of *kern_Q* along the top row and *kern_P* along bottom row at orientations 0°, 45°, 90° and 135°.

4) Layer 6: The function used for dx was outlined in Algorithm D.6.

Algorithm D.6 Layer 6, dx

function FUNC_DX_V1($x, C, z, x^{V2}, \delta_c, a, \phi, \Gamma, v^{21}, att$) ▷ $x, C, z, x^{V2}, att \in \mathbb{R}^{(i \times j \times K)}$
  $dx \leftarrow \delta_c \times (-x + ((1-x) \times ((\alpha \times C) + (\phi \times \text{MAX.}(z - \Gamma, 0)) + (v^{21} \times x^{V2}) + att)))$
  ▷ MAX. operates elementwise on z
  ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
  return dx
end function

5) Layer 4: The functions used for dy and dm have been outlined in Algorithm D.7 and Algorithm D.7 respectively. The sigmoid function, f, used in these functions has been shown in Algorithm D.9.

[2]presuming that $\sigma^2 = 0.5$ is a typo in [1, p. 462]

Algorithm D.7 Layer 4, dy

function FUNC_DY($y, C, x, m, \delta_c, \eta^+, \mu, \nu, n, kern\_W^+$) ▷ $y, C, x, m \in \mathbb{R}^{(i \times j \times K)}, kern\_W^+ \in \mathbb{R}^{(l_W \times l_w \times K \times K)}$
  $dy \leftarrow \delta_c \times (-y + ((1-y) \times (C + (\eta^+ \times x))) - ((1+y) \times \text{FUNC\_F}(m \times \text{FUNC\_FILTER\_W}(m, kern\_W^+), \mu, \nu, n)))$
  ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
  return dy
end function

Algorithm D.8 Layer 4, dm

function FUNC_DM($m, x\delta_m, \eta^-, \mu, \nu, n, kern\_W^-$) ▷ $y, C, x, m \in \mathbb{R}^{(i \times j \times K)}, kern\_W^+ \in \mathbb{R}^{(l_W \times l_w \times K \times K)}$
  $dm \leftarrow \delta_m \times (-m + (\eta^- \times x) - (m \times \text{FUNC\_F}(\text{FUNC\_FILTER\_W}(m, kern\_W^-), \mu, \nu, n)))$
  ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
  return dm
end function

Algorithm D.9 Layer 4 sigmoid function f

function FUNC_F(x, μ, ν, n) ▷ $x \in \mathbb{R}^{(i \times j \times K)}$
  $f\_x \leftarrow (\mu \times x^n)/(\nu^n + x^n)$
  ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
  return f_x
end function

6) Layer 2/3: The functions used for dz and ds have been outlined in Algorithm D.10 and Algorithm D.11 respectively. The function, H, used on the feedback from dy has been shown in Algorithm D.12.

Algorithm D.10 Layer 2/3, dz

function FUNC_DZ(z, y, H_z, s, δ_z) ▷ $z, y, H\_z, s \in \mathbb{R}^{(i \times j \times K)}$
  $\delta_z \times (-z + ((1-z) \times ((l \times \text{MAX.}(y, 0)) + H\_z + (a_{ex}^{23} \times att))) - ((z + \phi) \times (s * kern\_T^+)))$
  return dz
end function

Algorithm D.11 Layer 2/3, ds

function FUNC_DS(s, z, H_z, δ_s) ▷ $s, z, H\_z \in \mathbb{R}^{(i \times j \times K)}$
  $ds \leftarrow \delta_s \times (-s + H_z + (a_{in}^{23} \times att) - (s \times (s * kern\_T^-)))$
  return ds
end function

Algorithm D.12 Layer 2/3, intralaminar feedback H

function FUNC_H_Z(z, Γ) ▷ $z \in \mathbb{R}^{(i \times j \times K)}$
  $H_z \leftarrow$CREATE(Array$^{(i \times j \times K)}$)
  for $k \in [1 : K]$ **do**
    $H\_z[:, :, k] \leftarrow (\text{MAX.}(z[:, :, k] - \Gamma, 0)) * kern\_H[:, :, k]$
  end for
  return H_z
end function

B. Allocation

The equations were first implemented functions returning a value. These functions were called on each iteration of the solver. Allocations to memory were reduced by rewriting the functions to mutate an existing variable and return nothing. This was particularly for important for parallelisation as a bottleneck might form writing to and from the general purpose graphics processing unit (GPGPU).

Algorithm D.7, for example, was rearranged as shown in Algorithm D.13.

Algorithm D.13 Layer 4, dy, non-allocating

function FUNC_DY($dy, y, C, x, m, \delta_c, \eta^+, \mu, \nu, n, kern\_W^+$) ▷ $dy, y, C, x, m \in \mathbb{R}^{(i \times j \times K)}, kern\_W^+ \in \mathbb{R}^{(l_W \times l_w \times K \times K)}$
    $dy \leftarrow \delta_c \times (-y + ((1-y) \times (C + (\eta^+ \times x))) - ((1+y) \times \text{FUNC\_F}(m \times \text{FUNC\_FILTER\_W}(m, kern\_W^+), \mu, \nu, n)))$
                ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
    return *nothing*
end function

Allocation was reduced further by replacing functions, used internally by each of the equation functions, with non-allocating versions. Allocation for variables needed temporally were moved outside the equation functions. For example, Algorithm D.13 was rearranged as Algorithm D.14.

Algorithm D.14 Layer 4, dy, non-allocating, with temporary variable

function FUNC_DY($dy, y, C, x, m, \delta_c, \eta^+, \mu, \nu, n, kern\_W^+, dyTemp$) ▷
$dy, y, C, x, m \in \mathbb{R}^{(i \times j \times K)}, kern\_W^+, dyTemp \in \mathbb{R}^{(l_W \times l_w \times K \times K)}$
    $dyTemp \leftarrow \text{FUNC\_FILTER\_W}(m, kern\_W^+)$
    $dyTemp \leftarrow \text{FUNC\_F}(dyTemp, \mu, \nu, n)$
    $dy \leftarrow \delta_c \times (-y + ((1-y) \times (C + (\eta^+ \times x))) - ((1+y) \times dyTemp))$
                ▷ for operations with scalar and arrays: scalar is broadcast and operates elementwise on array
    return *nothing*
end function

D.III. Numerical methods

The equations outlined in Section D.I form a system of time-dependent, first order, ordinary differential equations. The use of thresholding and half-wave functions adds non-linearity to the system. The number of equations in the system is dependent on the dimensions of the input image, the number of orientations, and the number of layers modeled dynamically. A model of the LGN and V1 only, with k orientations and having a monochromatic input image of size $i \times j$ will have a total of $(5k+2) \times (i \times j)$. A model including V2 nearly doubles this number to $(10k+2) \times (i \times j)$. A model taking a small input image of 100×100, excluding V2, and with only with two orientations results in a large system, made up of have 120 000 equations. The structure of the system however, with convolution operators linking these equations, may be highly parallelisable, increasing the feasibility of solving such a system efficiently.

The number of orientations to be modeled has been restricted to two, as discussed in Section D.I. Further sensibly restrictions to the scope of the project may be to only model up to V1 and to use small, monochromatic input images.

A. Differential equations in Julia

The Julia package DifferentialEquations.jl allows an extensive range of solvers, integrates well with other Julia packages and is well documented. Importantly, DifferentialEquations.jl allows for easy parallelisation on GPU with the use of CuArrays.jl.

The first approach taken to implement the model with DifferentialEquations was through the package ModelingToolkit.jl. ModelingToolkit.jl works as an Intermediate Representation, generating a function for DiffenentialEquations.jl. It can also symbolically optimise the function, which was the main motivation for taking this approach.

It was found, however, to be unsuitable for a model with the size and structure of LAMINART. It appeared to remove the array structure used with the the convolution operators, attempting to output a symbolic form of each equation independently. This step was not only very slow, it also made the system less parallelisable.

The next approach taken was to implement the model directly with DifferentialEquations.jl. A problem is first defined with the function `ODEProblem(f,u0,tspan,p)` where `f` is a function containing the model's equations, `u0` is an array of initial conditions, `tspan` is the timespan and `p` contains parameters to be passed to the function. The function `solve(prob)` then solves this problem.

The function `f` can be made non-allocating to increase performance. It is specified to take in arrays `du` and `u` along with `p` and `tspan`. As it is non-allocating, it should return nothing. Both `du` and `u` have dimensions matching `u0` and are generated during solving.

Listing D.1 contains the function used with `ODEProblem`. As the model had layers with and without orientations, it was decided to make the initial values array, `u0`, three dimensional with size `[1:i, 1:j, 1:5k+2]`. Orientations from all layers were stacked atop of each other. To allow for cleaner indexing, it was decided to place the layers without orientations, `v_p` and `v_m`, after the layers with orientations. Lines 13-19 and 21-27 of Listing D.1 show the indexing used to access `u` and `du` respectively. The macro `@view` was used to create subarrays of `u` and `du`. A subarray translates indexing to allow fast and easy slicing without performing a copy.

```
mutable struct LamFunction_imfil_cpu{T::AbstractArray} <: Function
    x_lgn::T
    C::T
    H_z::T
    H_z_temp::T
    v_C_temp1::T
    v_C_temp2::T
    v_C_tempA::T
    W_temp::T
end

function (ff::LamFunction_imfil_cpu)(du, u, p, t)
    x = @view u[:, :, 1:p.K]
    y = @view u[:, :, p.K+1:2*p.K]
    m = @view u[:, :, 2*p.K+1:3*p.K]
    z = @view u[:, :, 3*p.K+1:4*p.K]
    s = @view u[:, :, 4*p.K+1:5*p.K]
    v_p = @view u[:, :, 5*p.K+1]
    v_m = @view u[:, :, 5*p.K+2]

    dx = @view du[:, :, 1:p.K]
    dy = @view du[:, :, p.K+1:2*p.K]
    dm = @view du[:, :, 2*p.K+1:3*p.K]
    dz = @view du[:, :, 3*p.K+1:4*p.K]
    ds = @view du[:, :, 4*p.K+1:5*p.K]
    dv_p = @view du[:, :, 5*p.K+1]
    dv_m = @view du[:, :, 5*p.K+2]

    LaminartEqImfilter.fun_x_lgn!(ff.x_lgn, x, p)
    LaminartEqImfilter.fun_v_C!(ff.C, v_p, v_m, ff.v_C_temp1, ff.v_C_temp2, ff.v_C_tempA, p)
    LaminartEqImfilter.fun_H_z!(ff.H_z, z, ff.H_z_temp, p)
    LaminartEqImfilter.fun_dv!(dv_p, v_p, p.r, ff.x_lgn, p)
    LaminartEqImfilter.fun_dv!(dv_m, v_m, .-p.r, ff.x_lgn, p)
    LaminartEqImfilter.fun_dx_v1!(dx, x, ff.C, z, p.x_V2, p)
    LaminartEqImfilter.fun_dy!(dy, y, ff.C, x, m, ff.W_temp, p)
    LaminartEqImfilter.fun_dm!(dm, m, x, ff.W_temp, p)
    LaminartEqImfilter.fun_dz!(dz, z, y, ff.H_z, s, p)
    LaminartEqImfilter.fun_ds!(ds, s, ff.H_z, p)
    return nothing
end
```

Listing D.1: The function and struct used to input the model to ODEProblem.

A mutable struct was made to hold the non-state arrays `x_lgn`, `C` and `H_z`. The function inherits this struct. This was to reduce allocation during solving. Temporary arrays were also placed in the struct and passed to functions to further reduce allocation. The function then runs the models equation functions which modify `du`.

The output of `prob = ODEProblem(f,u0,tspan,p)` was passed to `sol = solve(prob)` to run the model.

Options for `solve()` include specifying a solver algorithm. If no solver is specified, the package looks at the size and type of `u0` and tries to select the an appropriate solver. This was first used. Various solvers were later benchmarked. Error tolerances and the model's stiffness may be specifed to adjust how a solver is suggested.

D.IV. Parallelisation

The size of the model and the type of operations it involves would suggest that large performance gains could be made through parallelisation. Parallelisation may be done on a GPGPU with the use of Compute Unified Device Architecture (CUDA) [3]. CUDA, a proprietary platform from NVidia , allows access to NVidia GPGPUs for general purpose programming.

A. Hardware, native arithmetic instructions and data types

The model was run on an NVidia Tesla K40 with CUDA Toolkit 8.0 GA2. This GPGPU has a compute capacity of 3.5. The compute capacity specifies the core architecture of the GPGPU. The native arithmetic instructions throughput for the compute capacity is specified in [3]. The NVidia Tesla K40 can perform Float32, Float64 and Int32 operations nativly. Float16 operations may not be done nativily on this GPGPU. As outlined in Table D.1, Float32 has the highest throughput for the NVidia Tesla K40.

Table D.1
THROUGHPUT OF NATIVE ARITHMETIC INSTRUCTIONS FOR NVIDIA TESLA K40 [3]

Data type	Number of multiply, multiply-add operations per clock cycle per multiprocessor
Float16	n/a
Float32	192
Float64	64
Int32	32

REFERENCES

[1] R. D. Raizada and S. Grossberg, "Context-sensitive binding by the laminar circuits of V1 and V2: A unified model of perceptual grouping, attention, and orientation contrast", *Visual Cognition*, vol. 8, no. 3-5, p. 431, Jun. 2001. DOI: 10/d8phhc.

[2] S. Grossberg and J. R. Williamson, "A Neural Model of how Horizontal and Interlaminar Connections of Visual Cortex Develop into Adult Circuits that Carry Out Perceptual Grouping and Learning", *Cerebral Cortex*, vol. 11, no. 1, pp. 37–58, Jan. 1, 2001. DOI: 10/bn377j.

[3] "CUDA C Programming Guide", NVIDIA, White Paper PG-02829-001_v8.0, Jun. 2017, p. 280.

Appendix E

Testing and results

E.I. Benchmarking

A. Methods for convolution

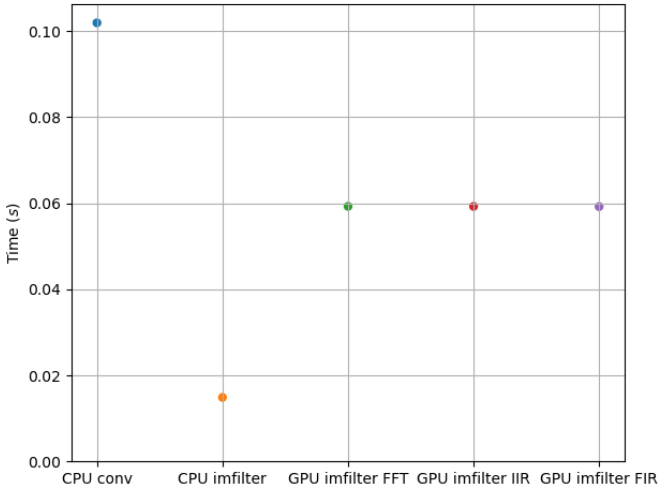


Figure E.1.

B. Kernel size

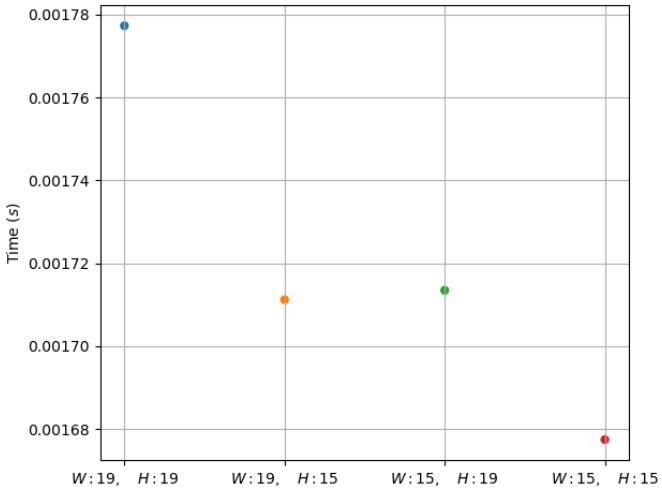
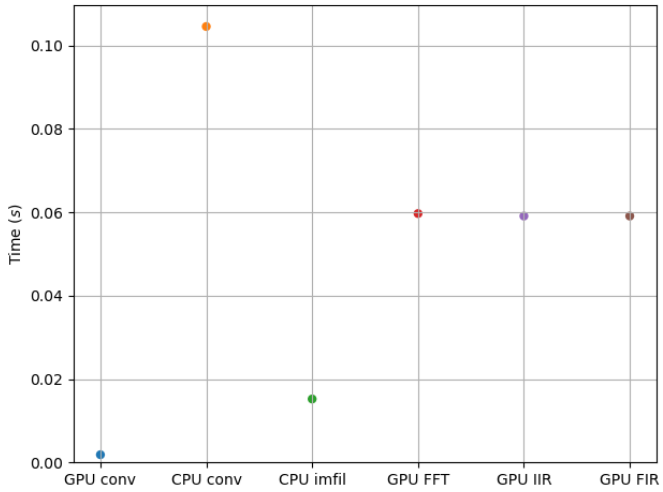


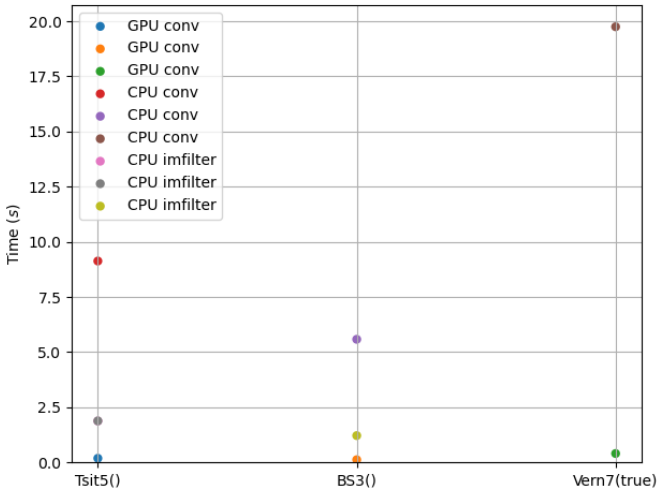
Figure E.2.

E.II. Noise

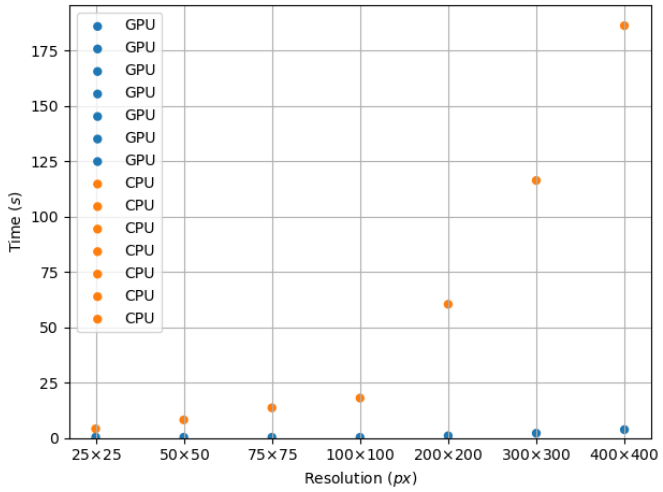
E.III. Testing



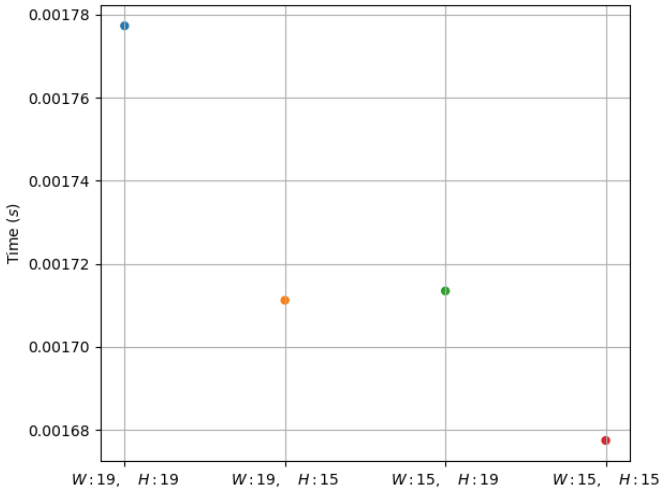
(a)



(b)

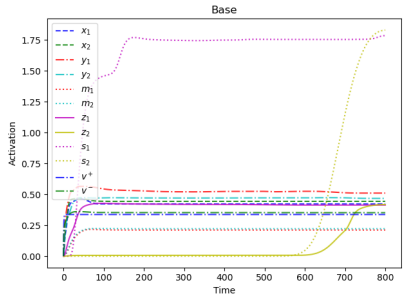


(c)

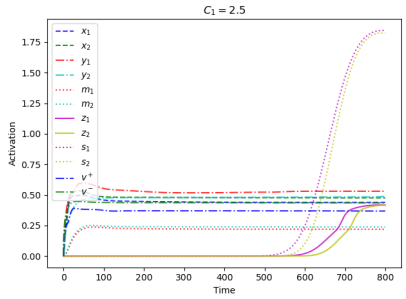


(d)

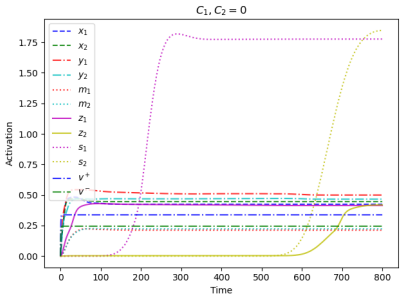
Figure E.3. Effect of noise on model with Kanizsa square.



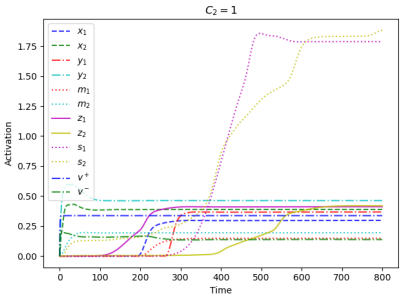
(a)



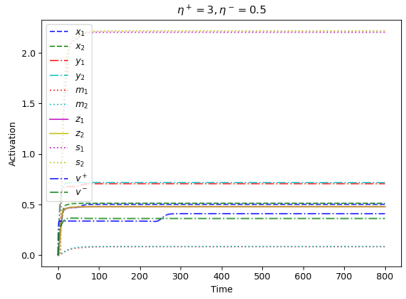
(b)



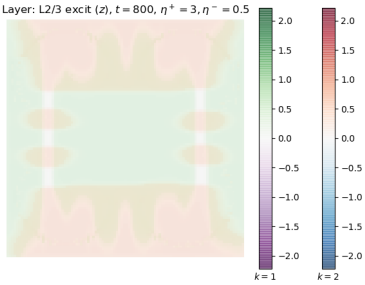
(c)



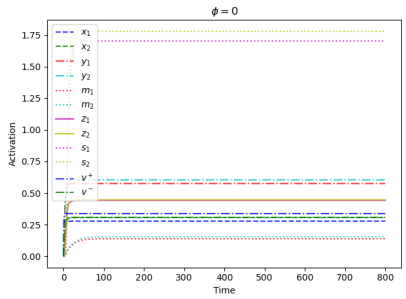
(d)



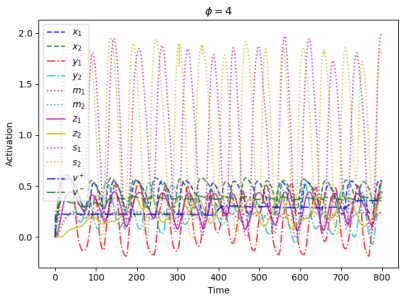
(e)



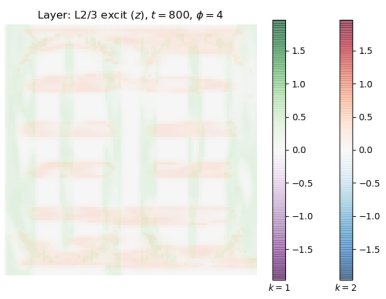
(f)



(g)

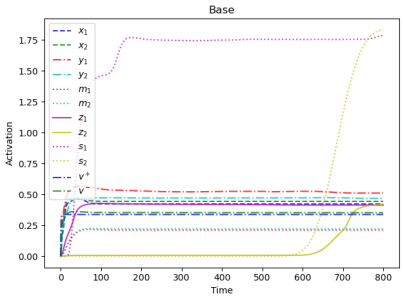


(h)

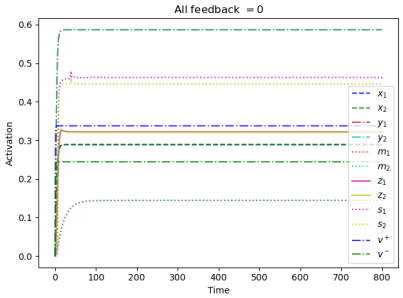


(i)

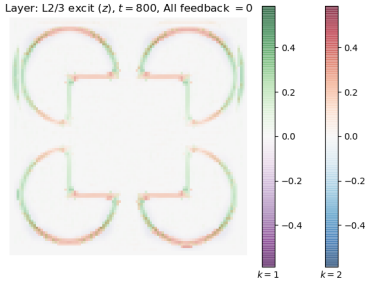
Figure E.4. Variations in model parameters



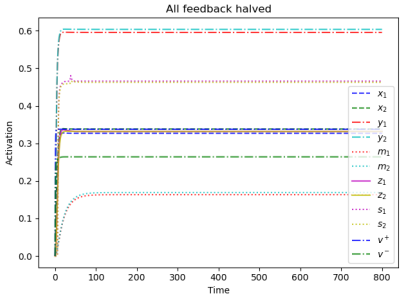
(a)



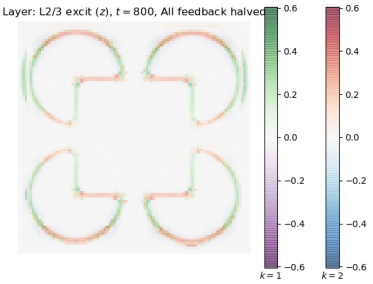
(b)



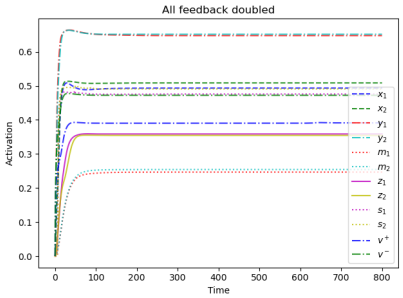
(c)



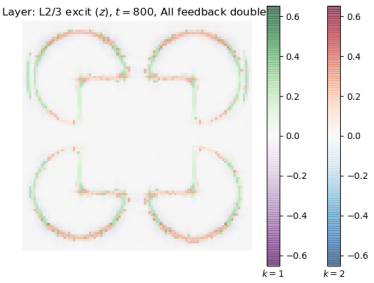
(d)



(e)

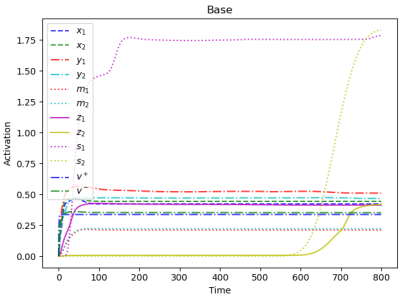


(f)

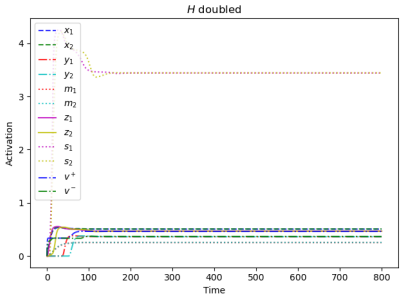


(g)

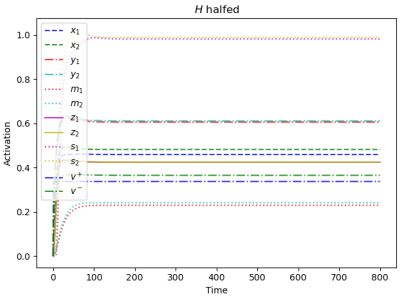
Figure E.5. Variations in all three feedback terms.



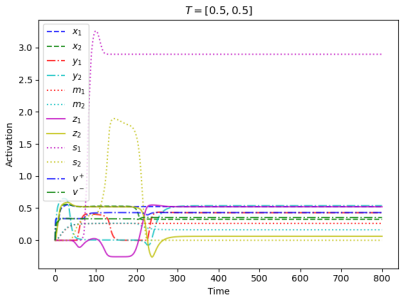
(a)



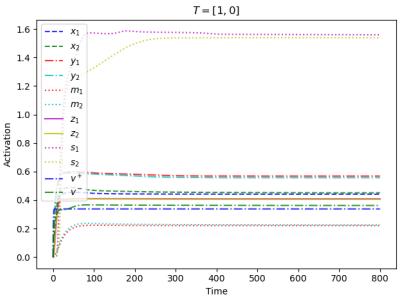
(b)



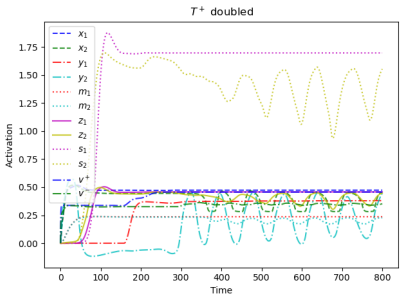
(c)



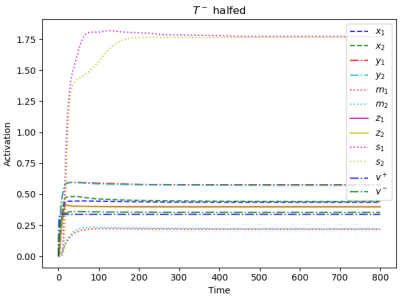
(d)



(e)



(f)



(g)

Figure E.6. Variations in kernel strength.

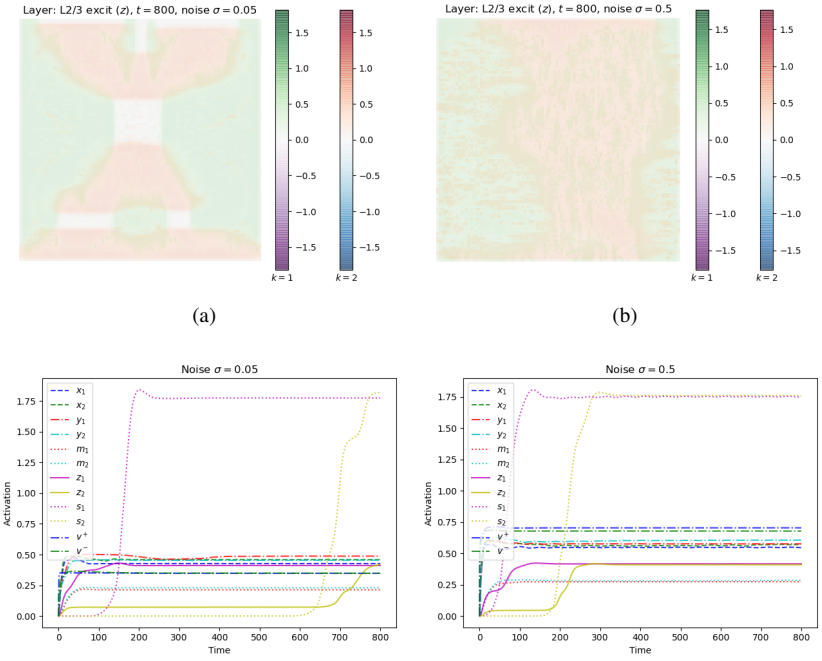


Figure E.7. Effect of noise on model with Kanizsa square.

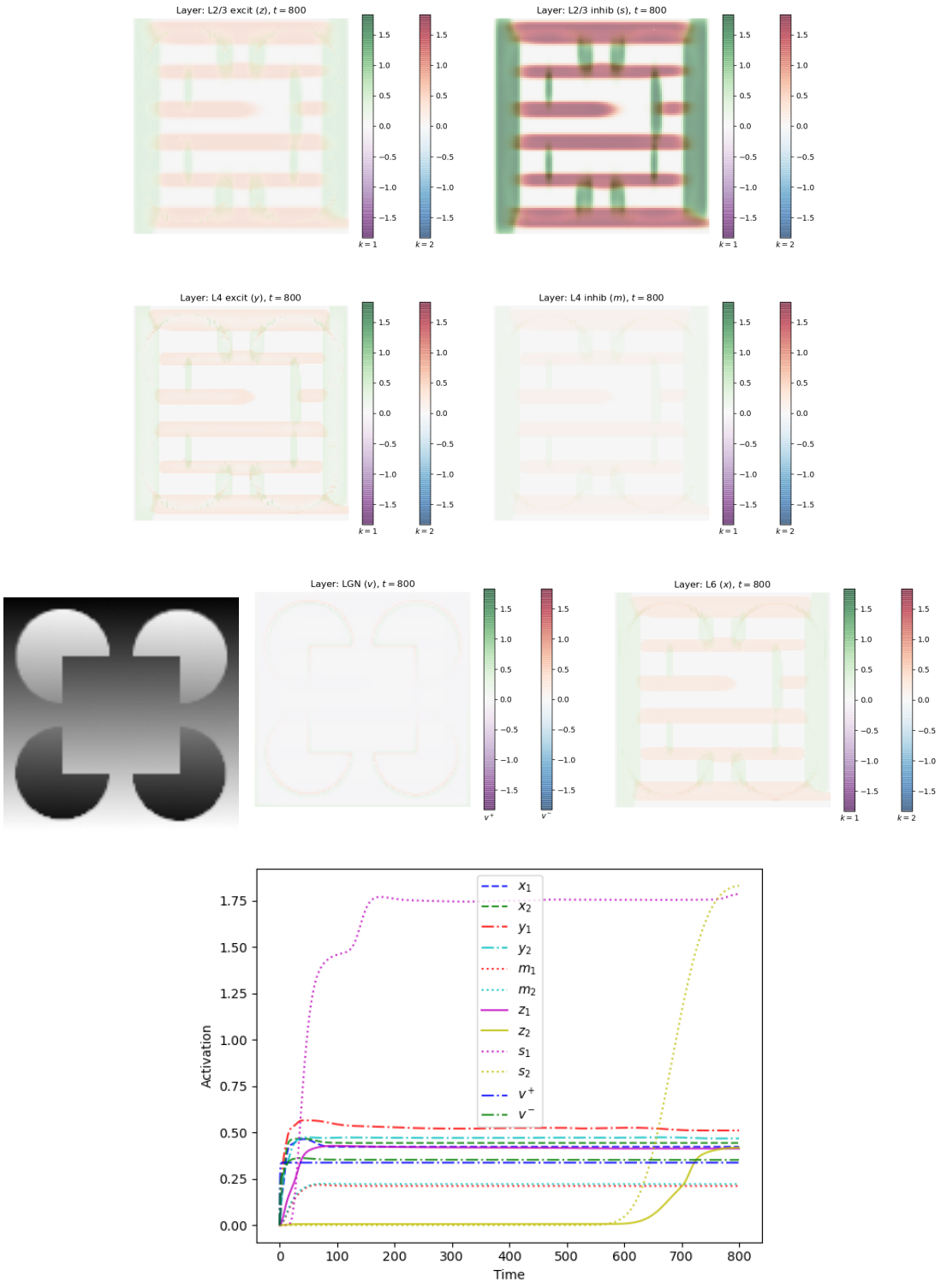

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.8.

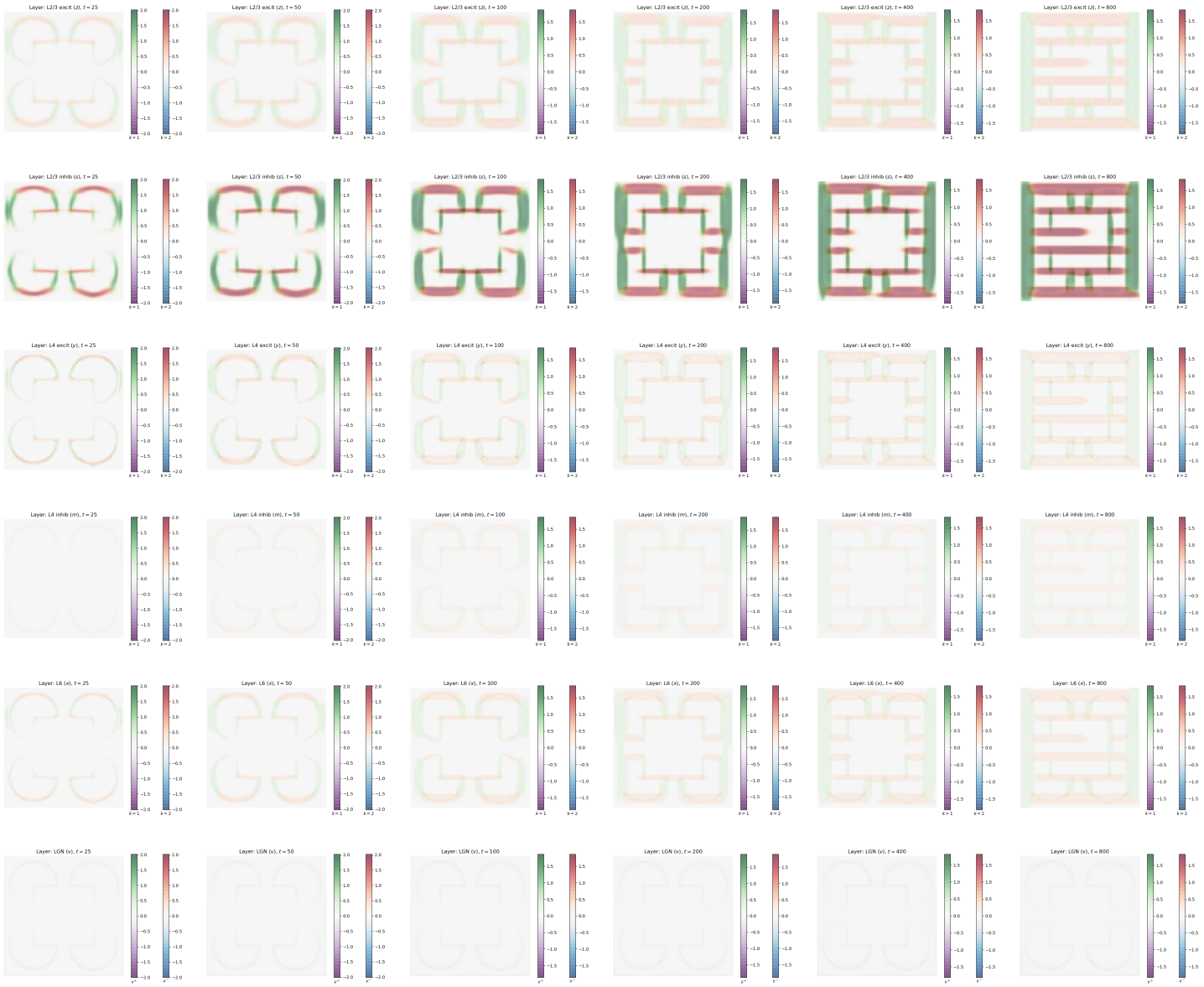

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.9.

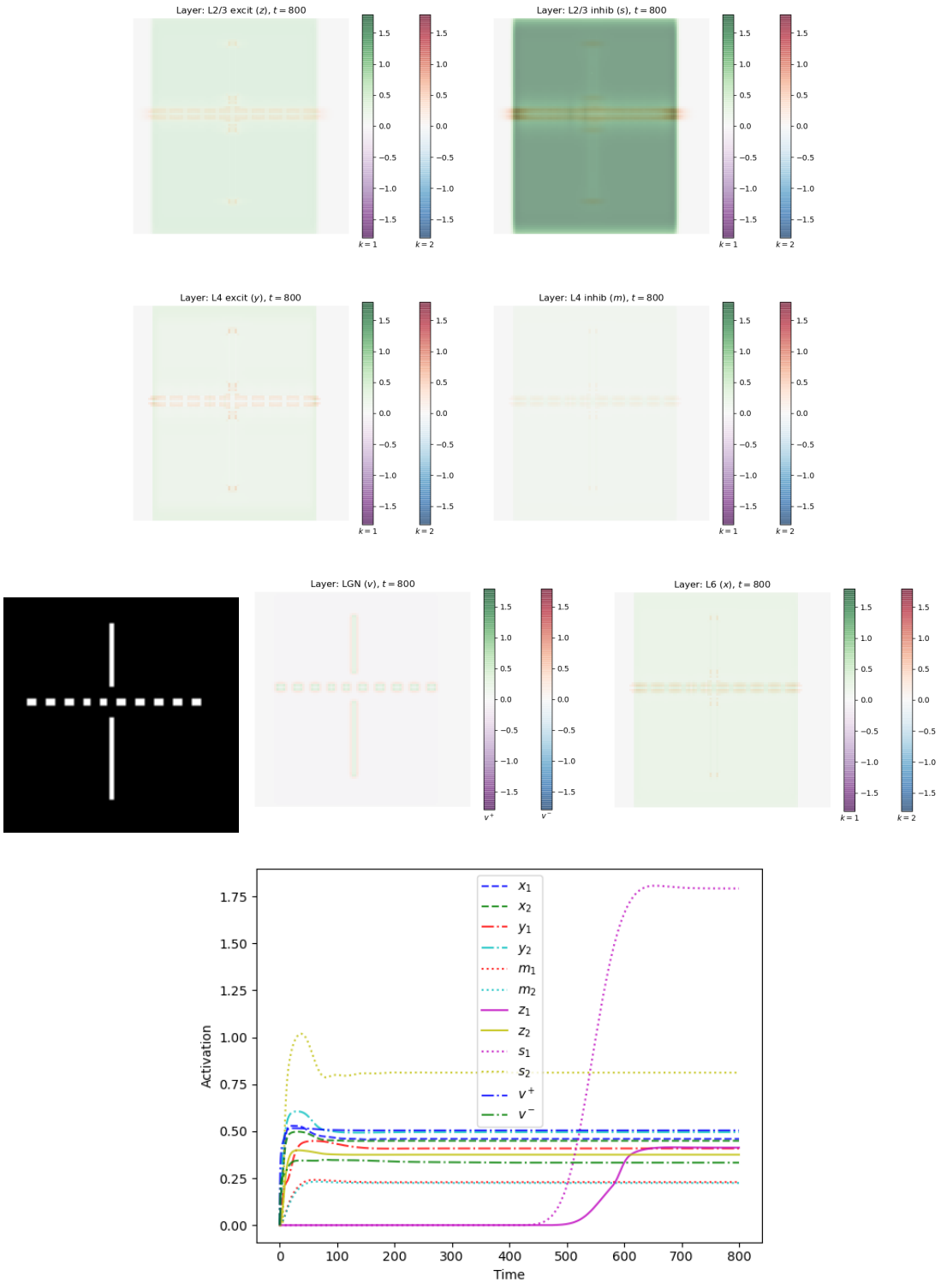

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.10.

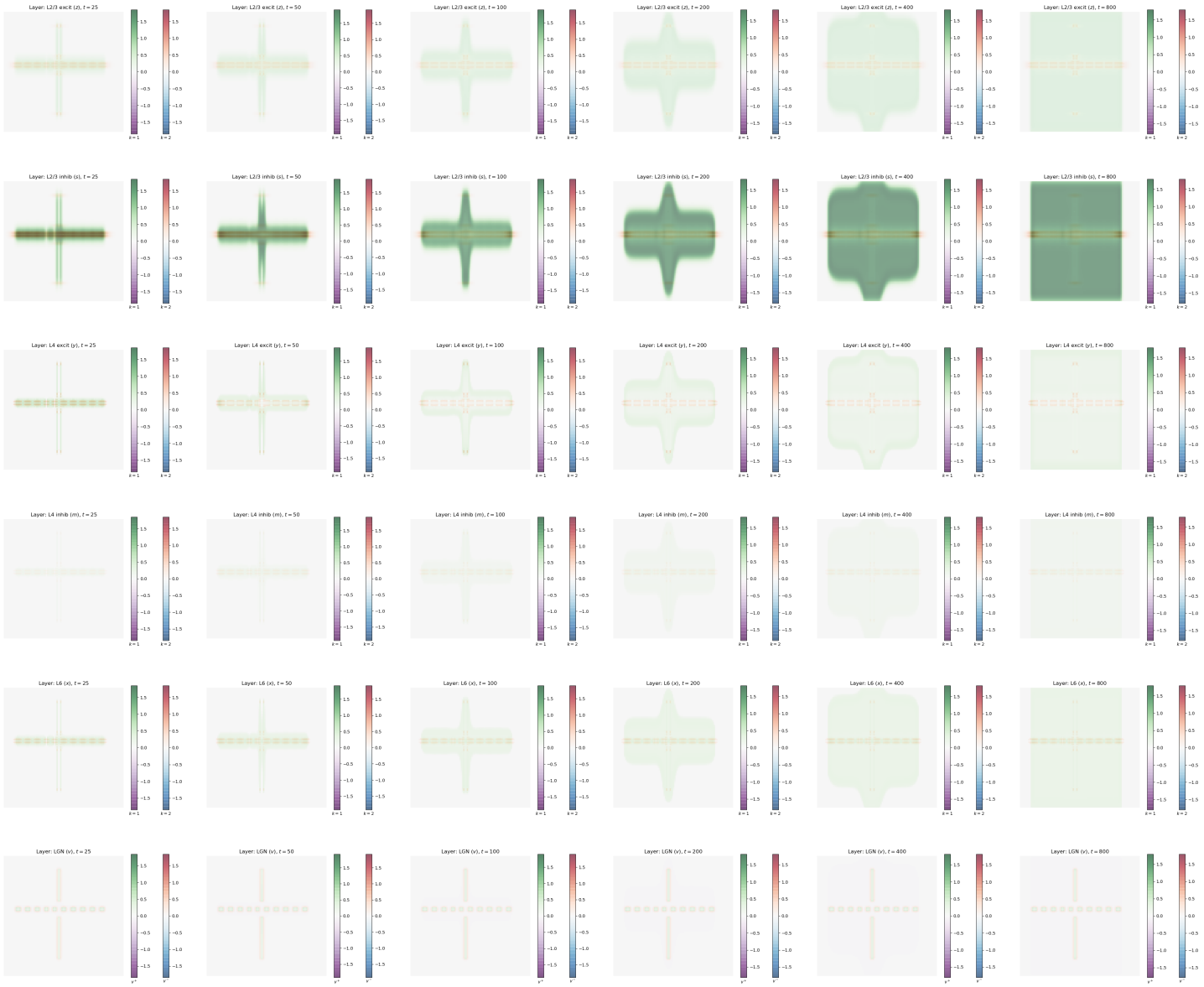

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.11.

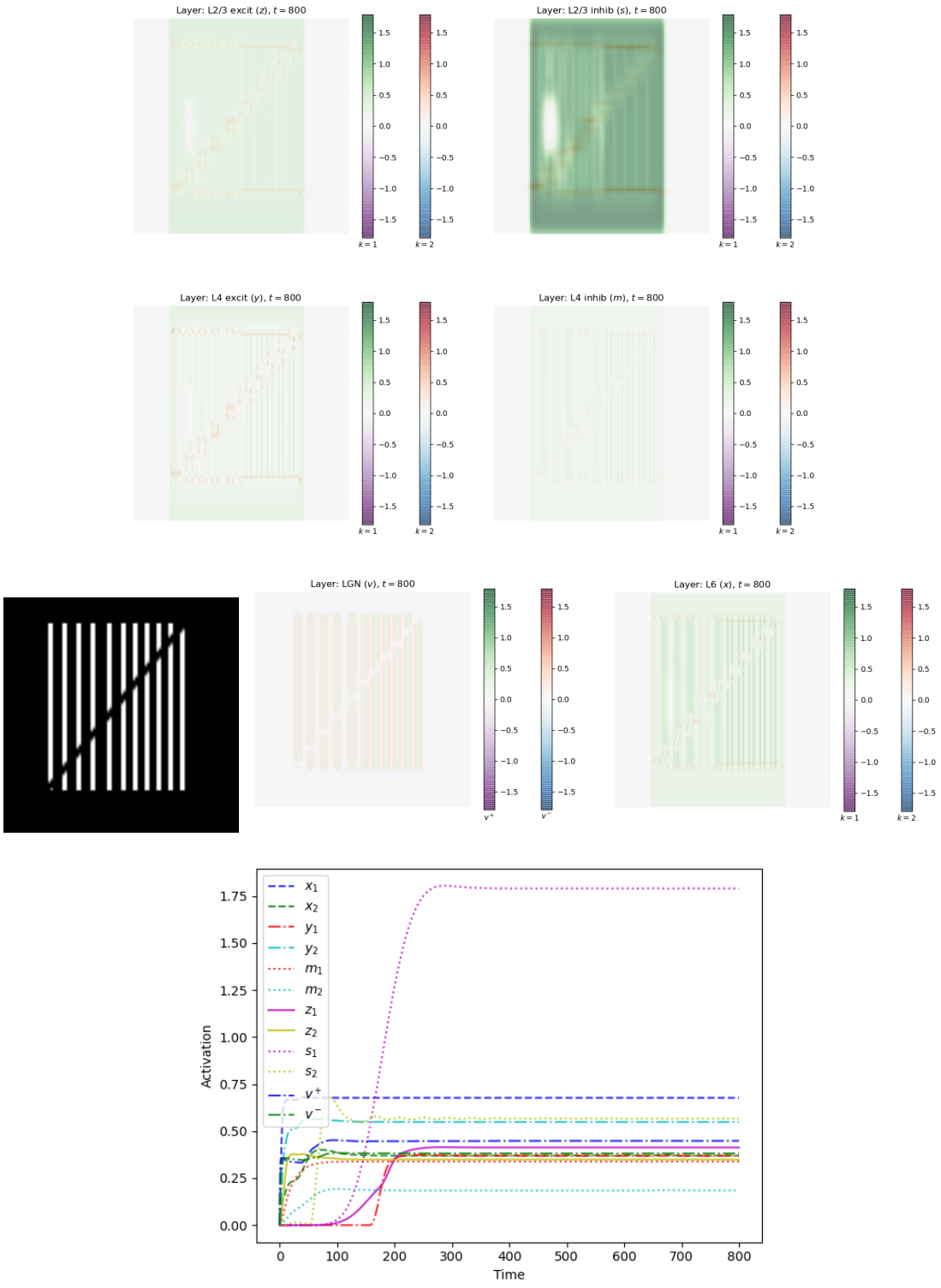

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
Activation
Time

Figure E.12.

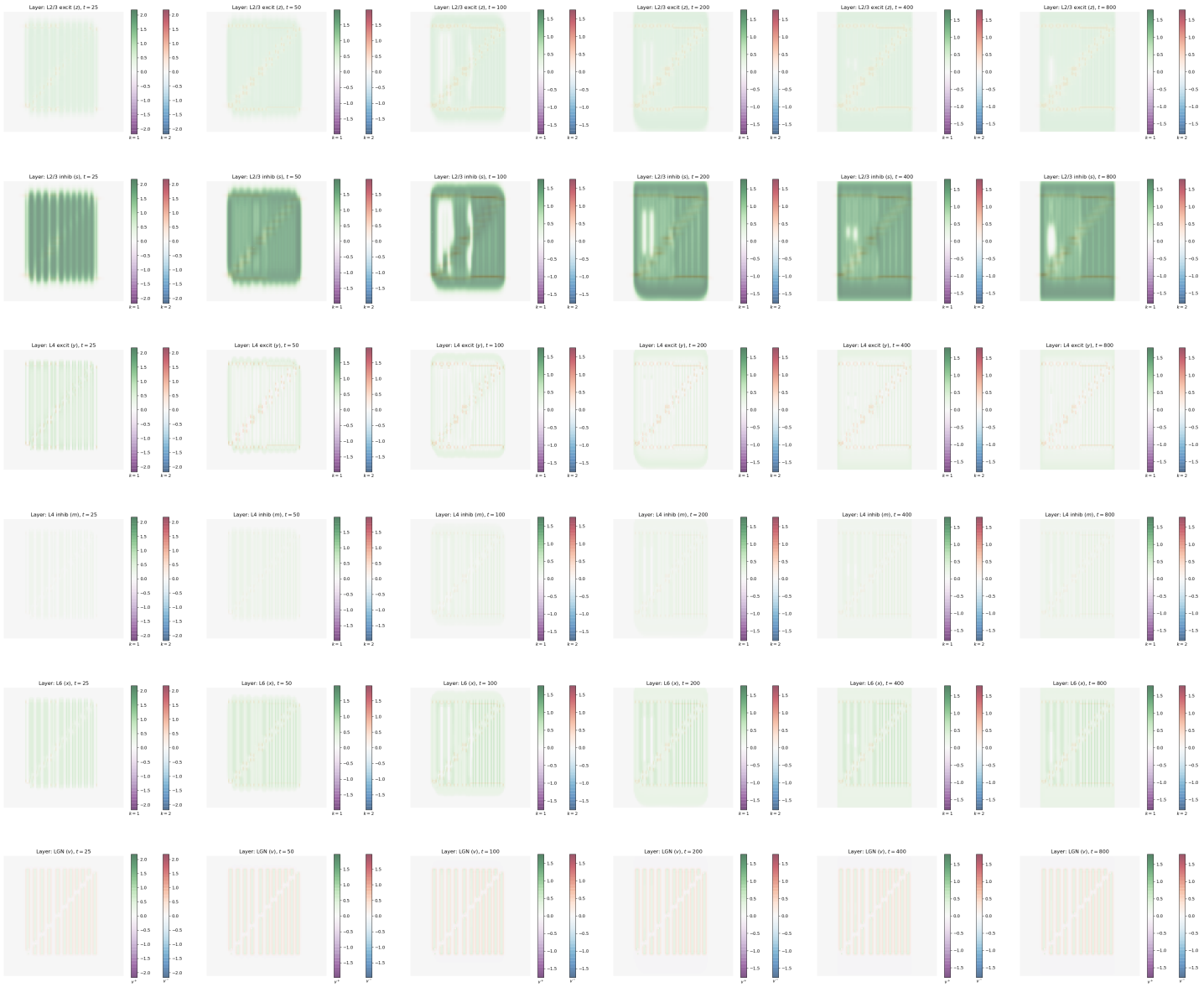

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.13.

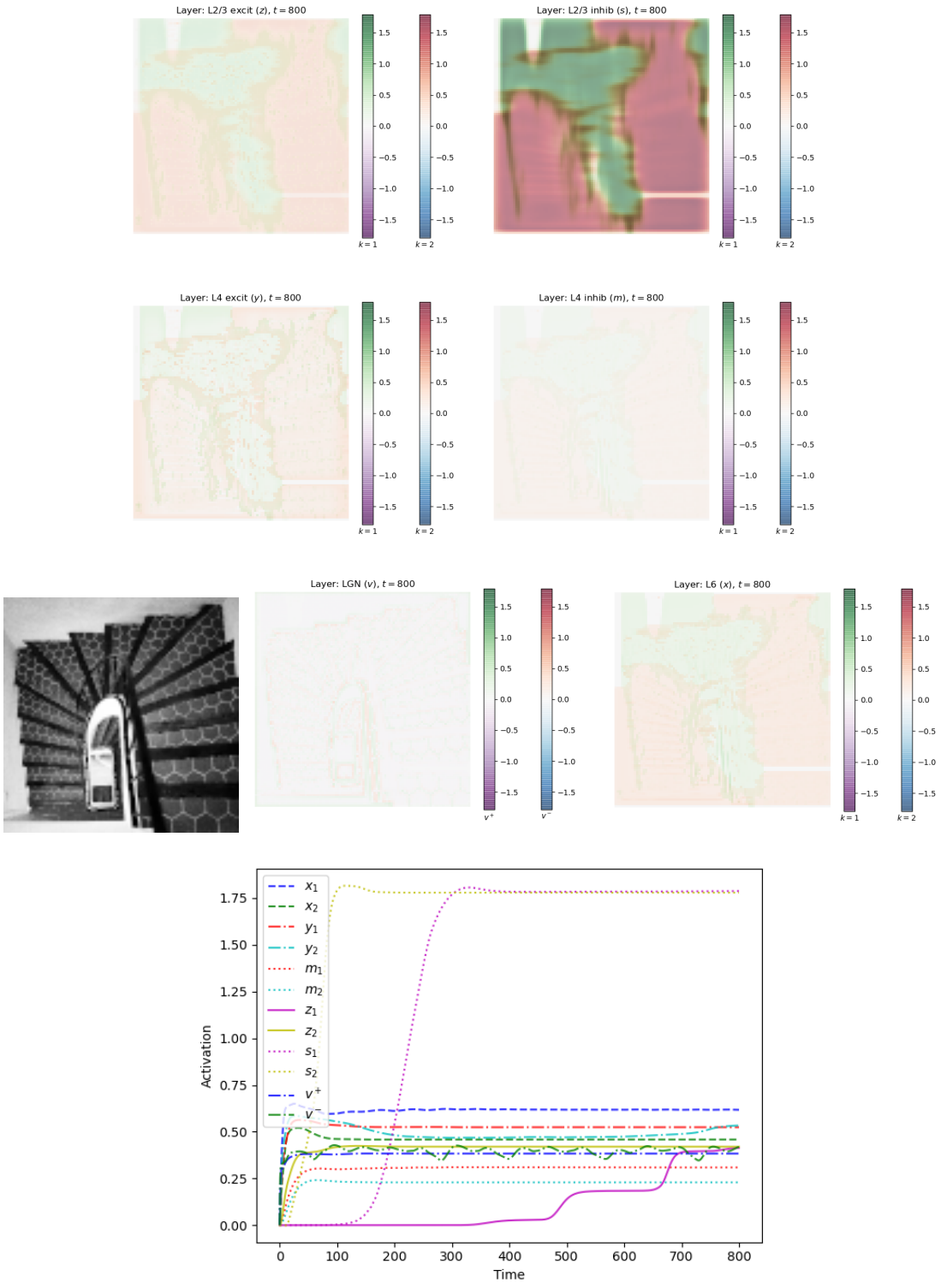

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.14.

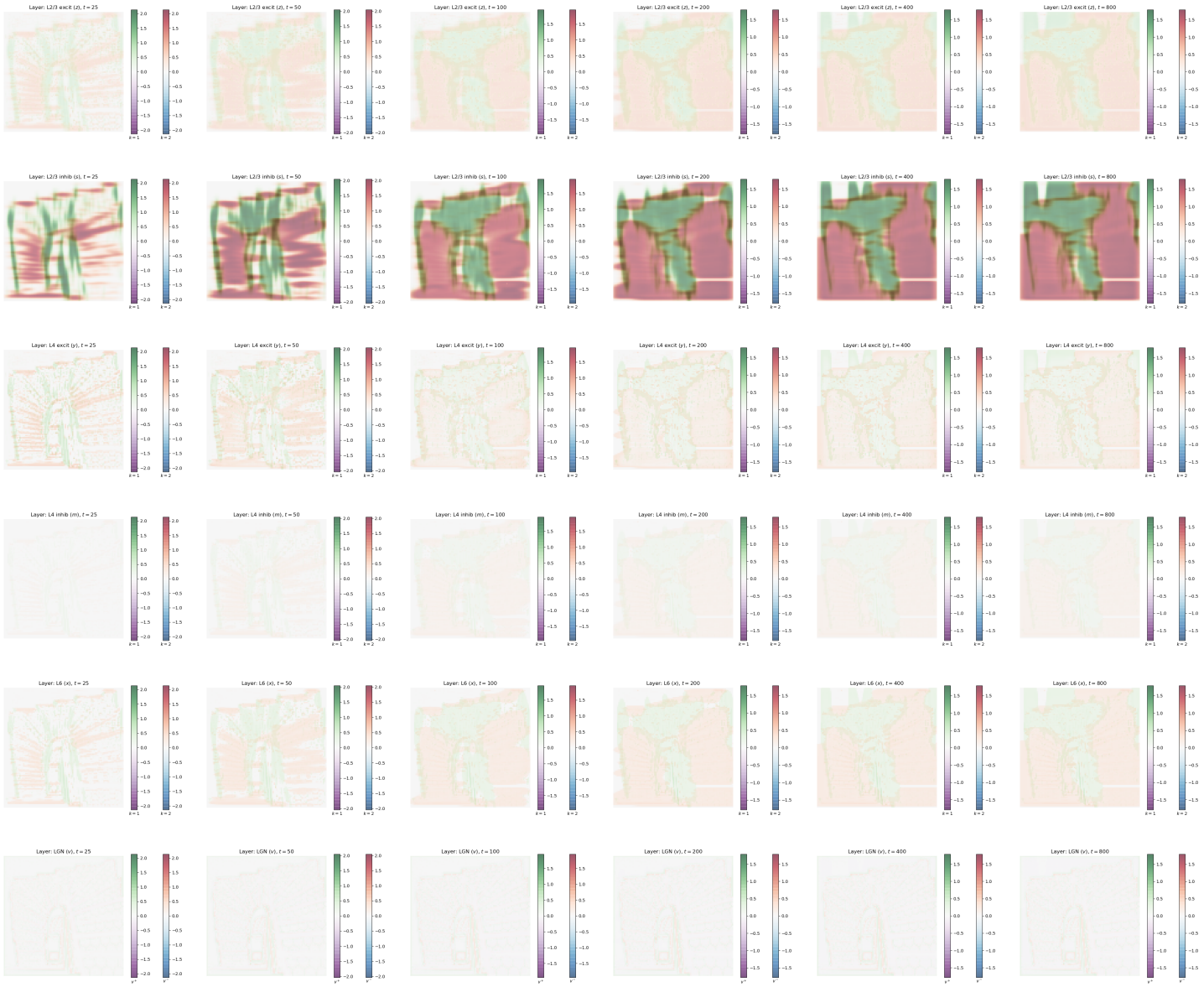

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.15.

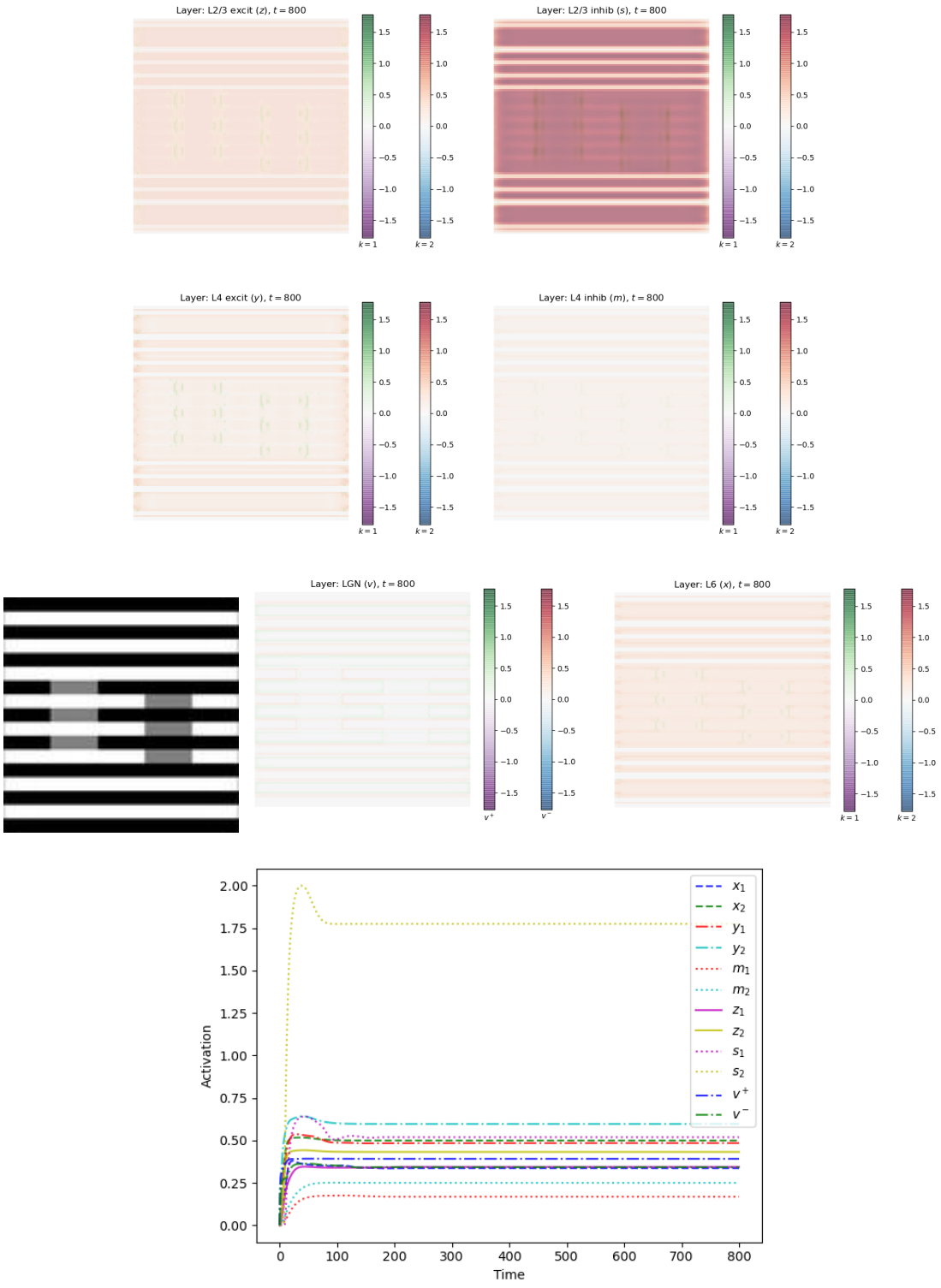

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
Activation
Time
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-

Figure E.16.

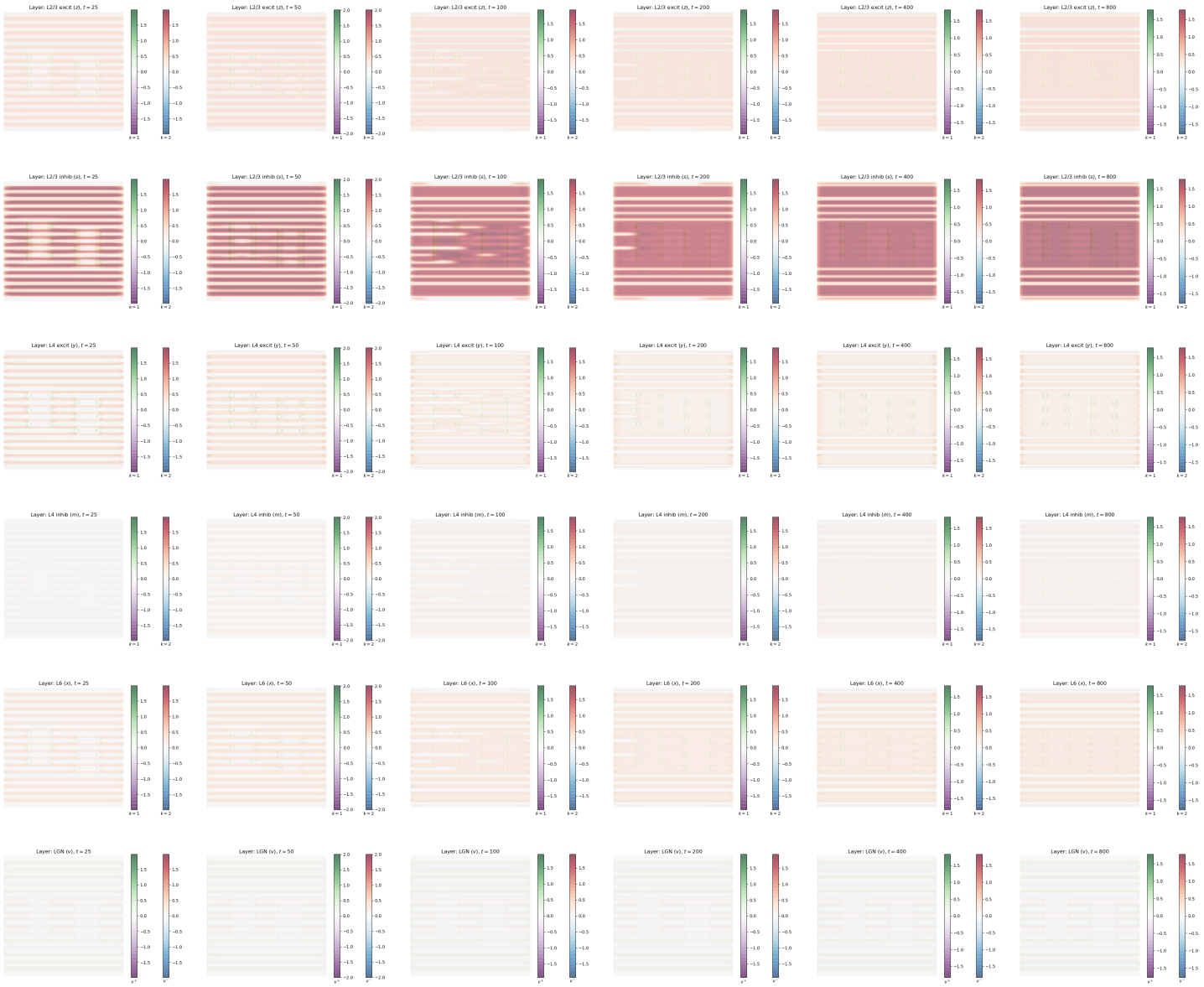

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.17.

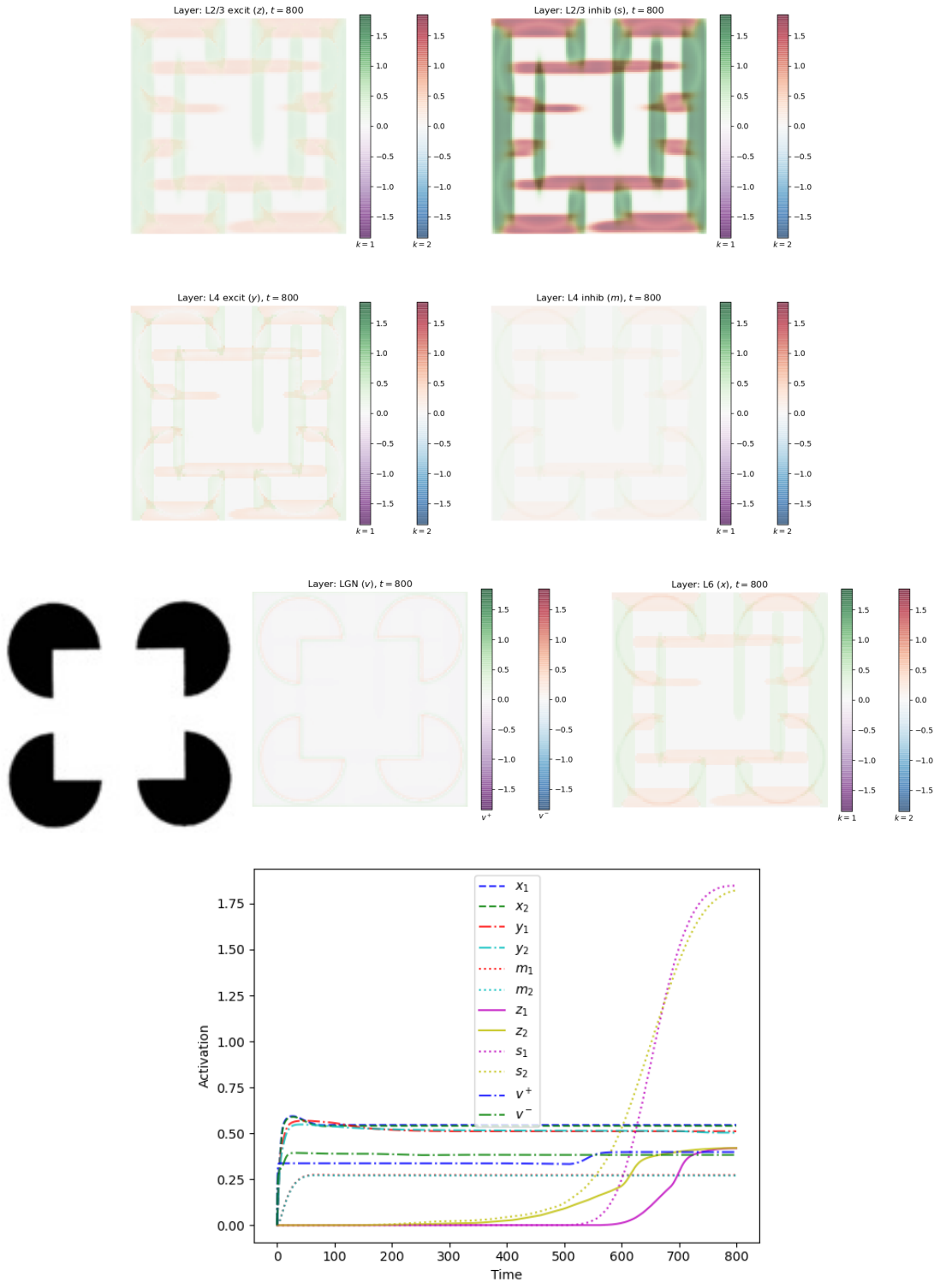

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.18.

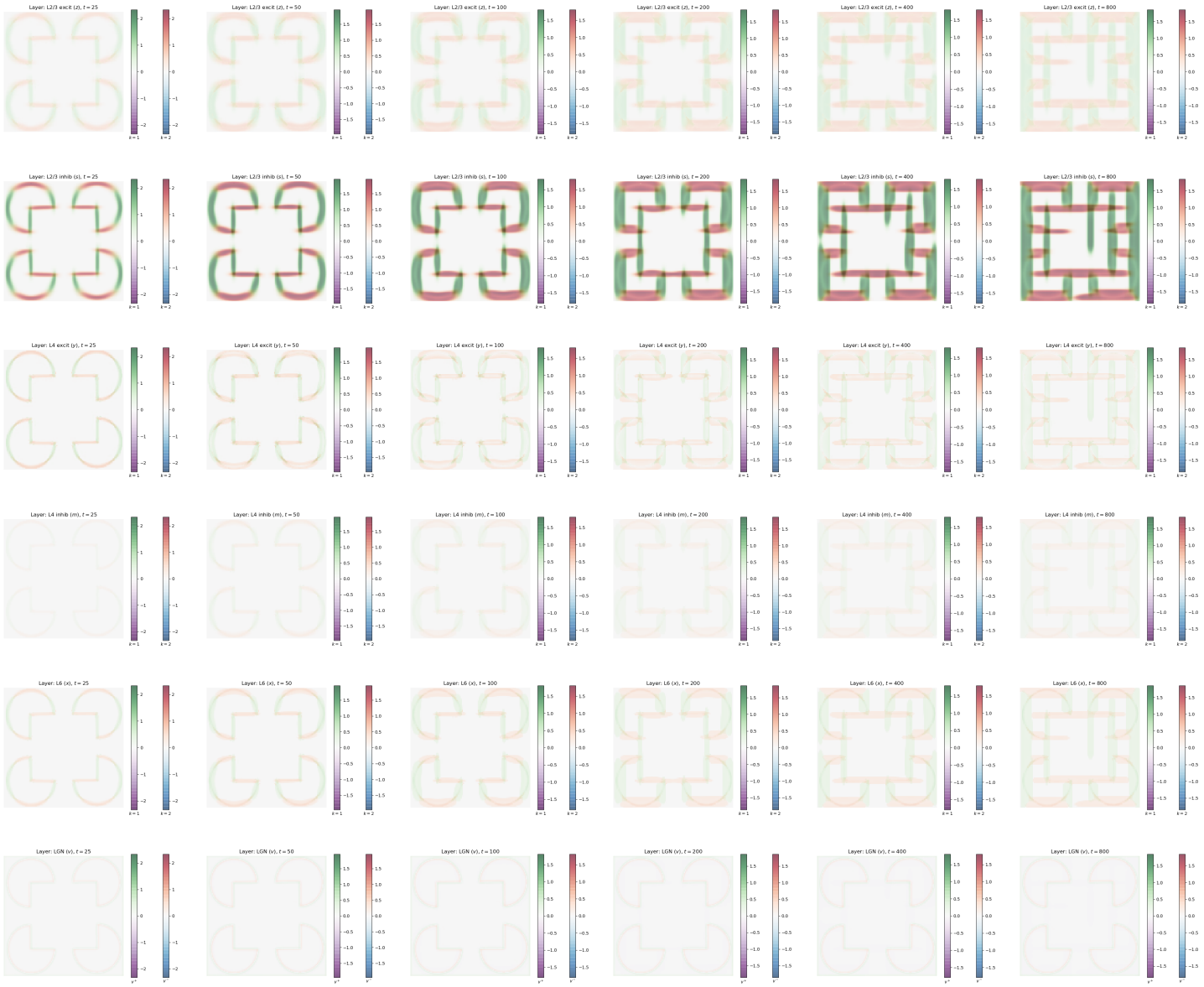

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.19.

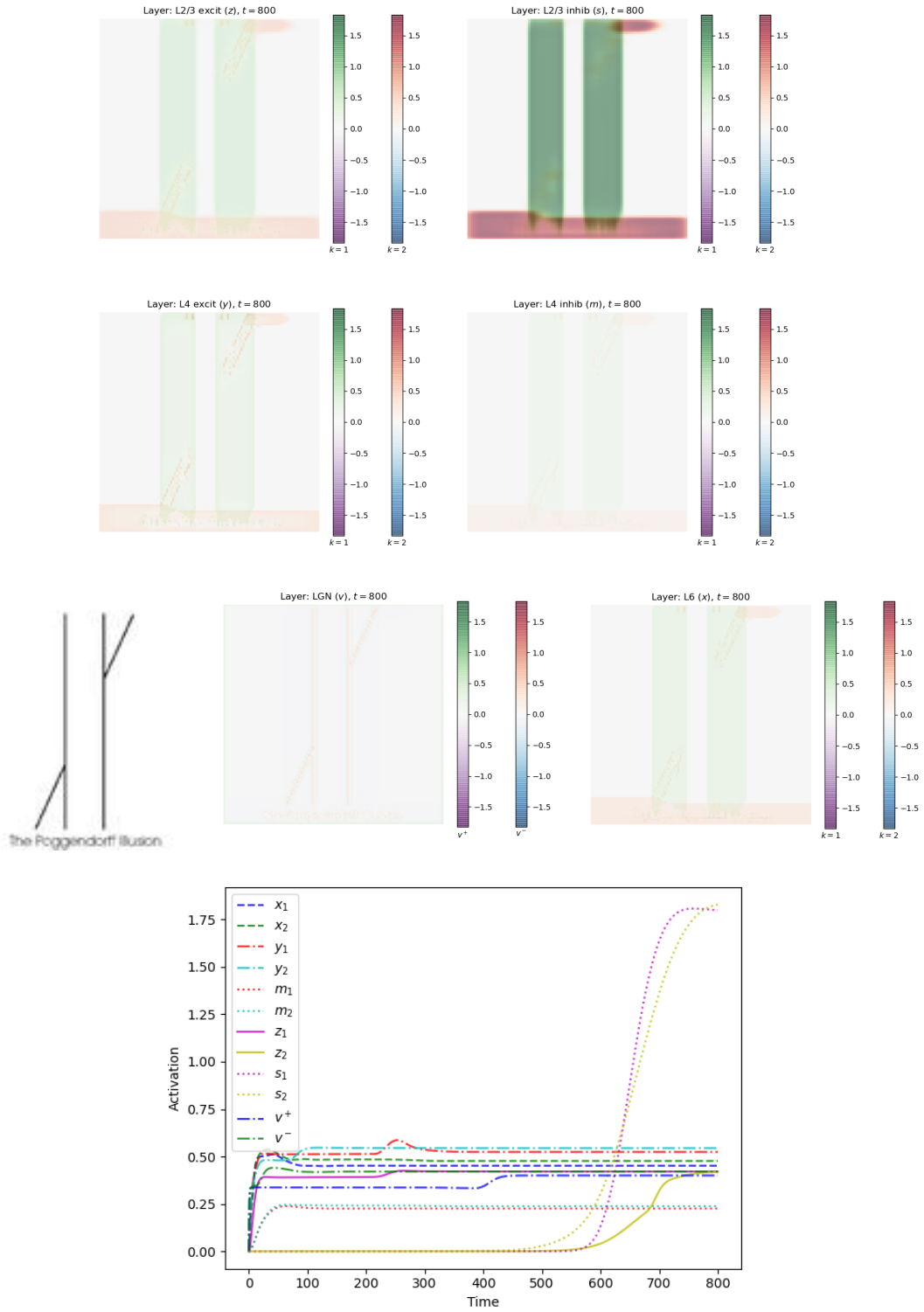

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
The Poggendorff illusion
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.20.

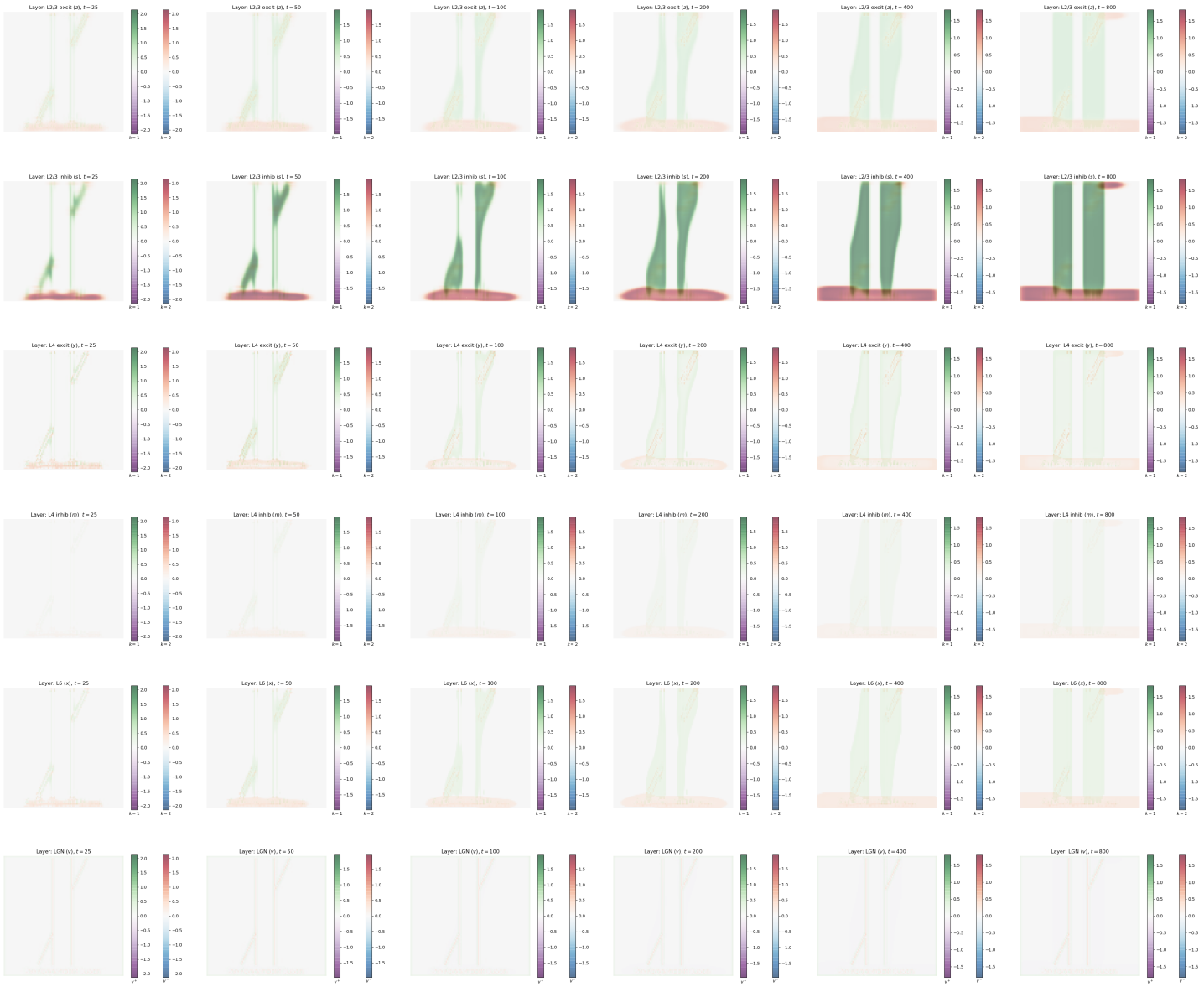

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.21.

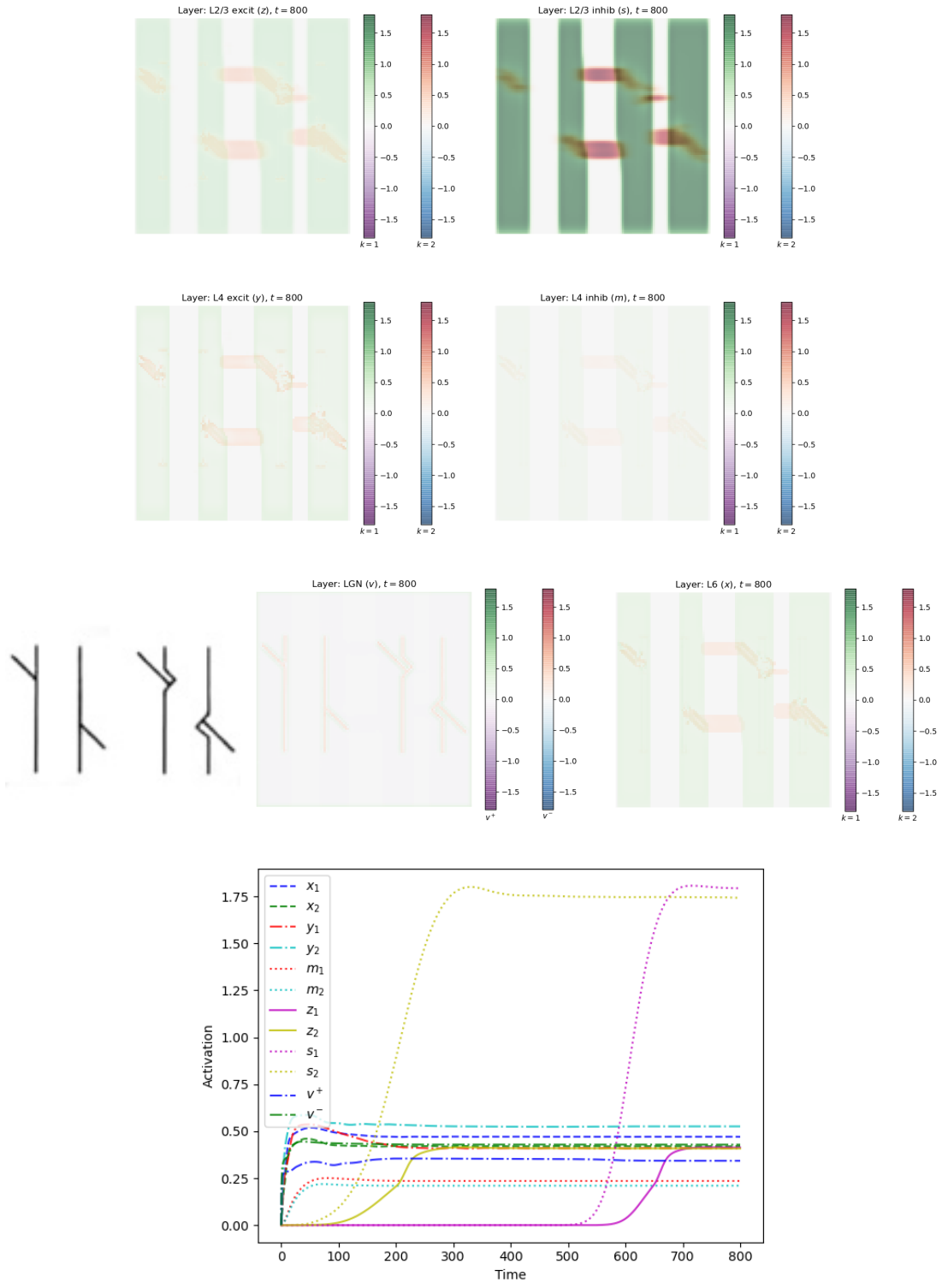

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
v+
v−
Activation
Time

Figure E.22.

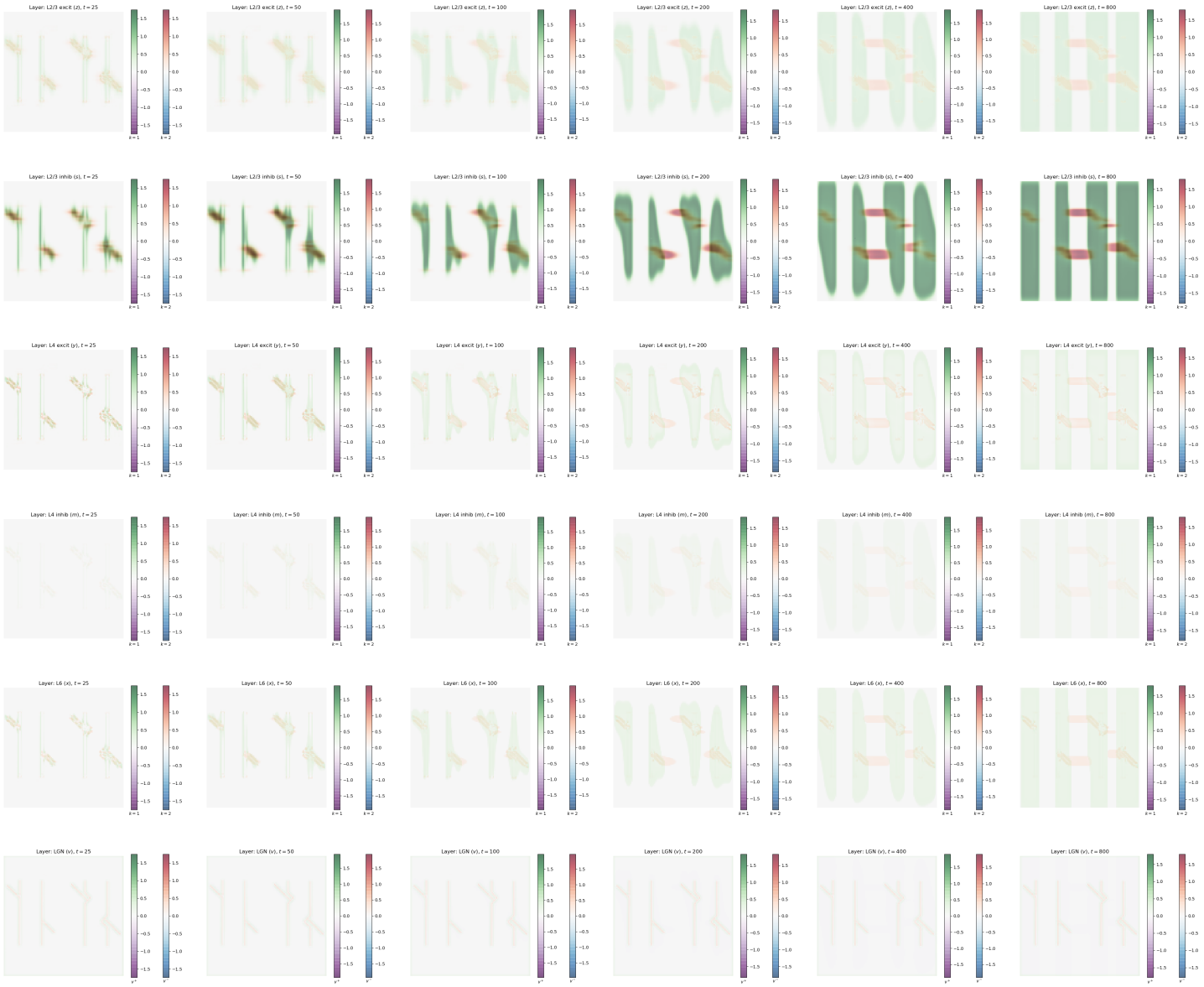

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.23.

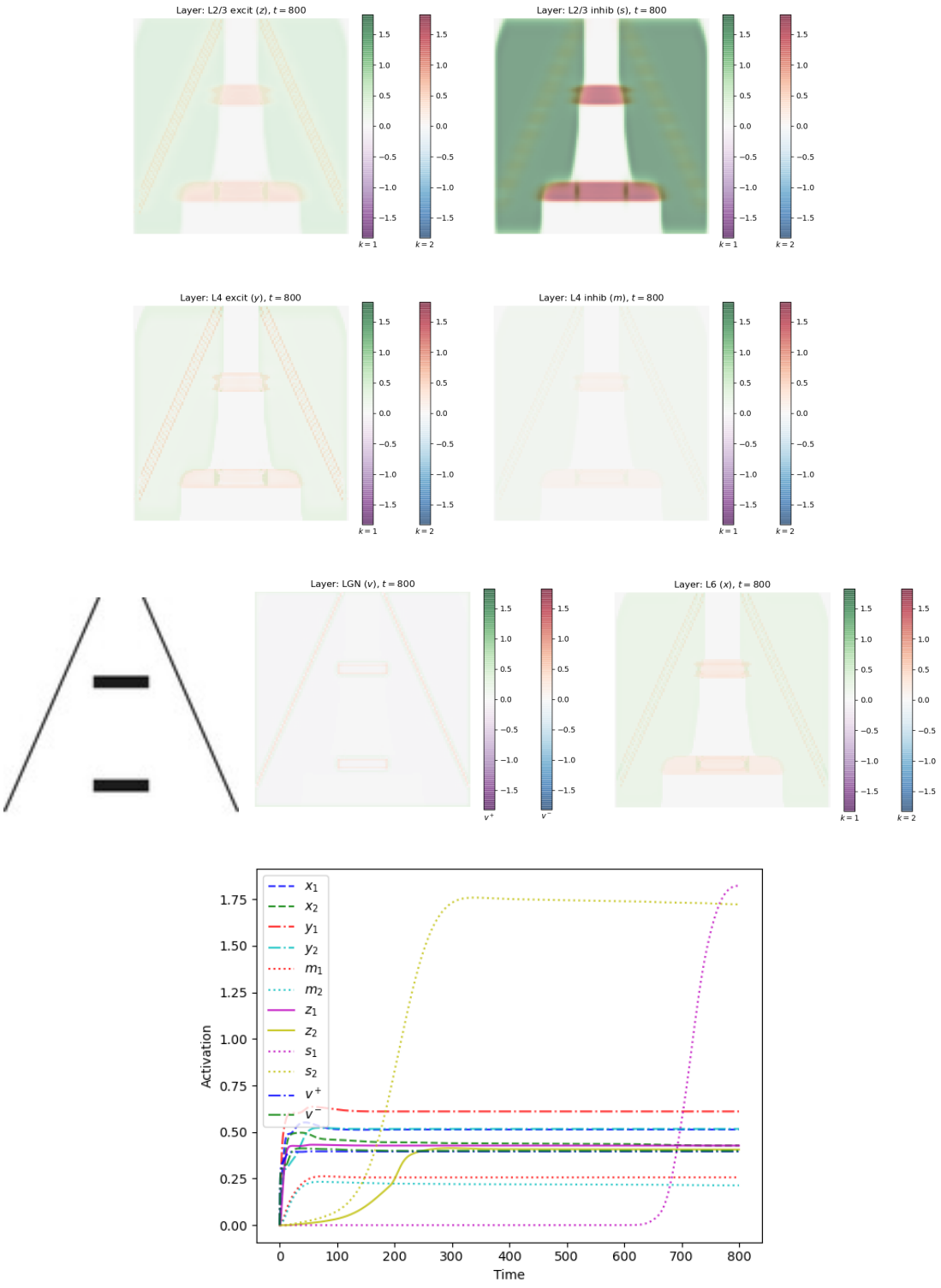

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
Activation
Time
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
v+
v−

Figure E.24.


Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.25.

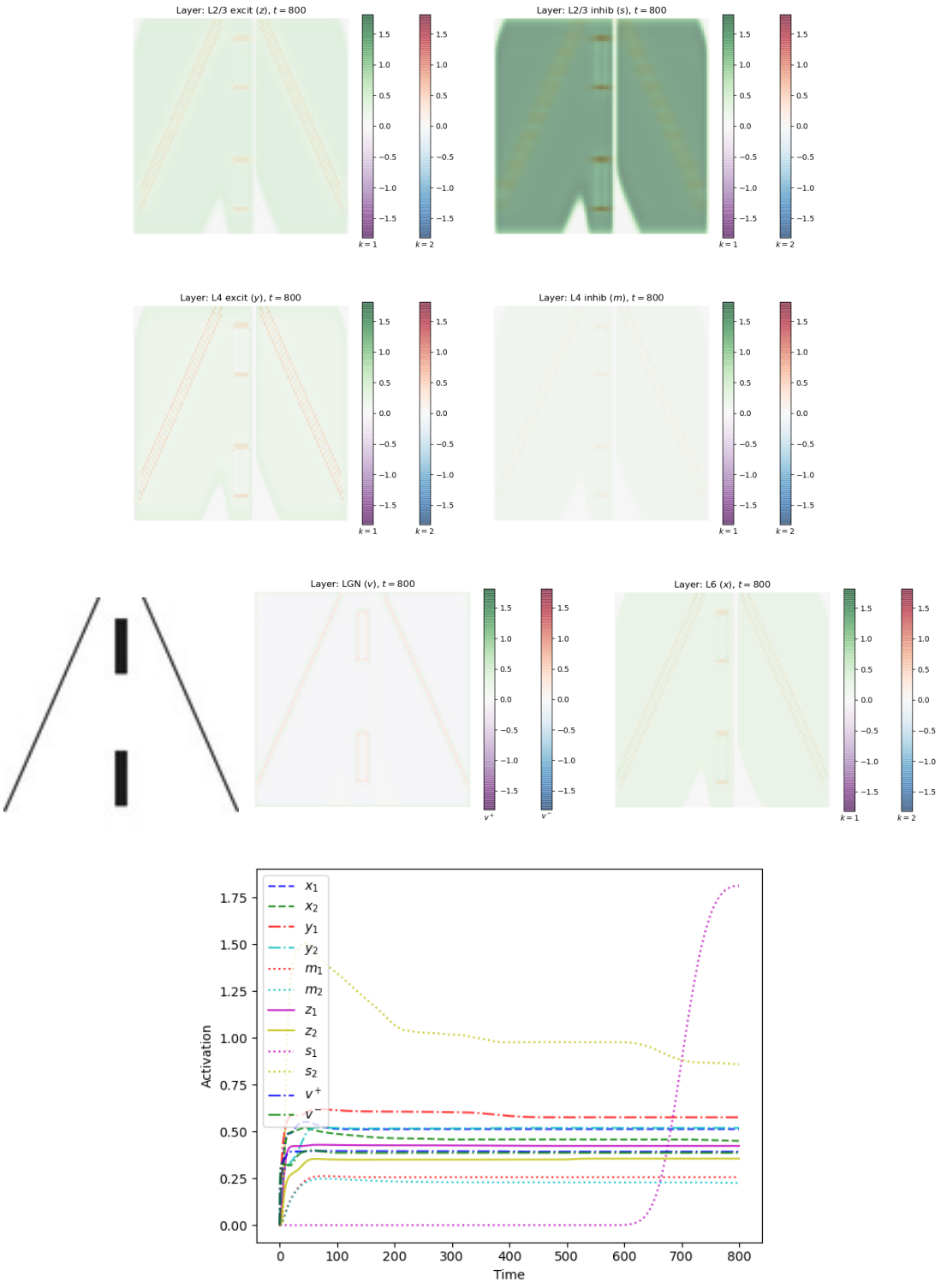

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
Activation
Time
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
v+
v−

Figure E.26.

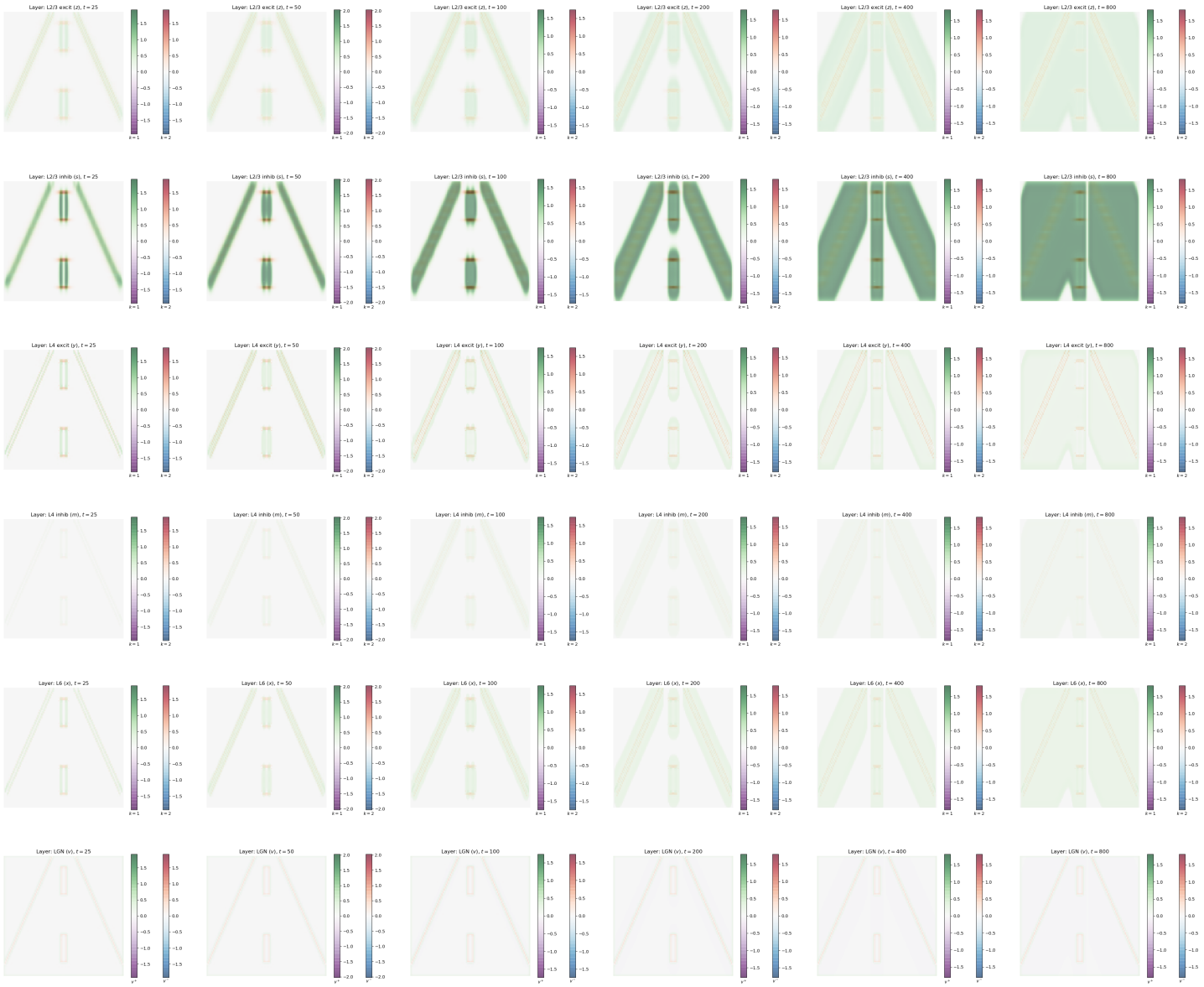

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.27.

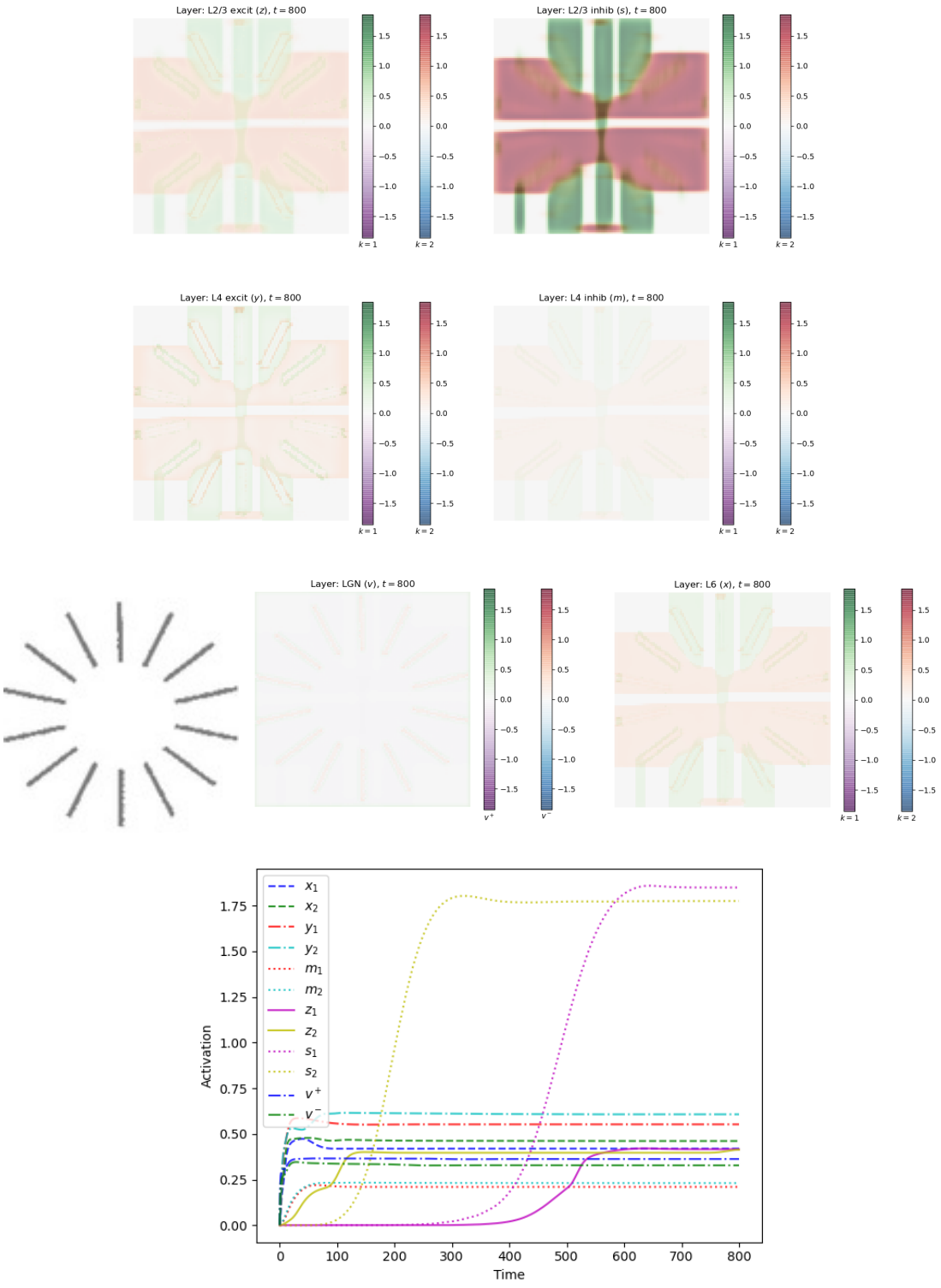

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.28.

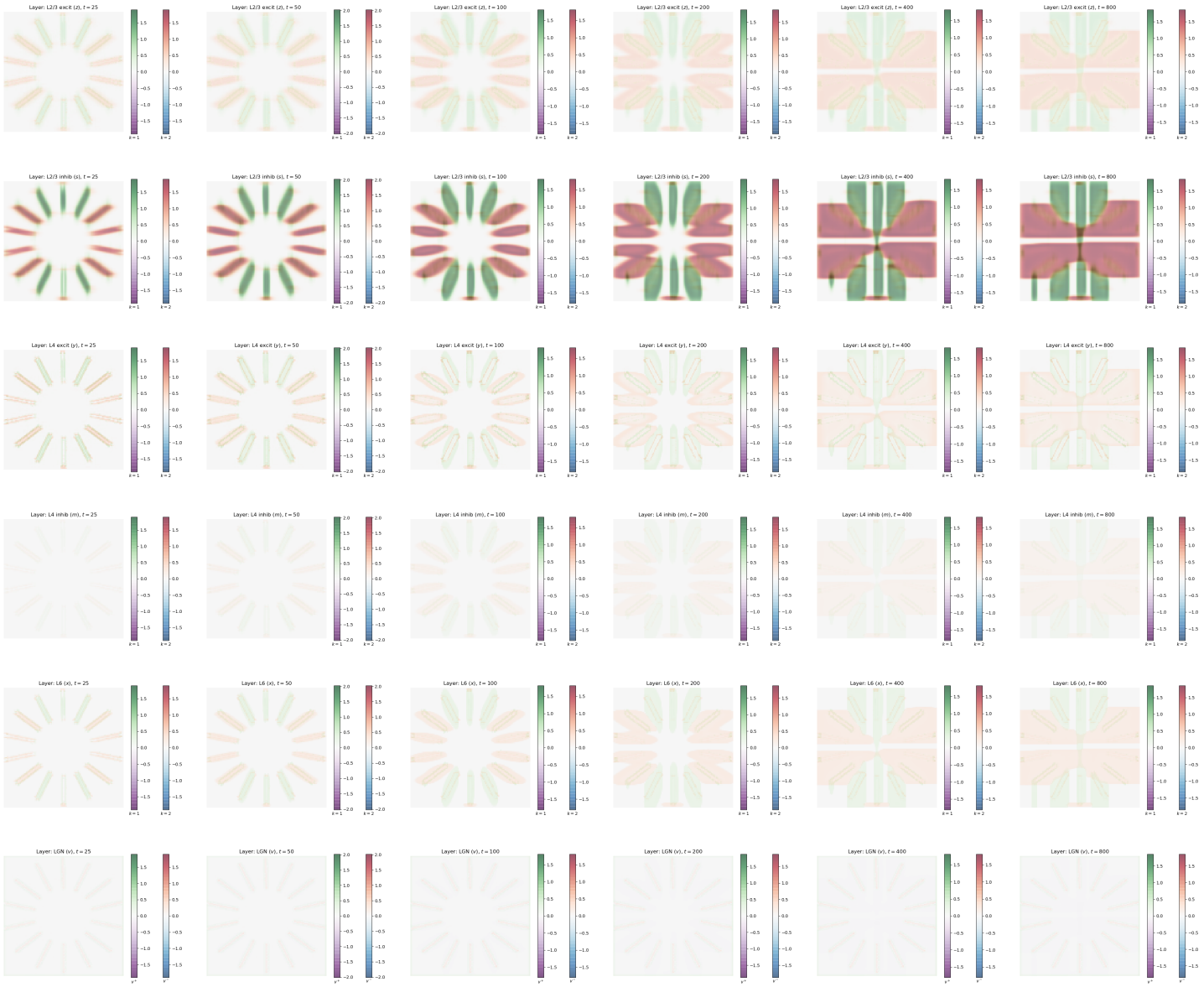

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.29.

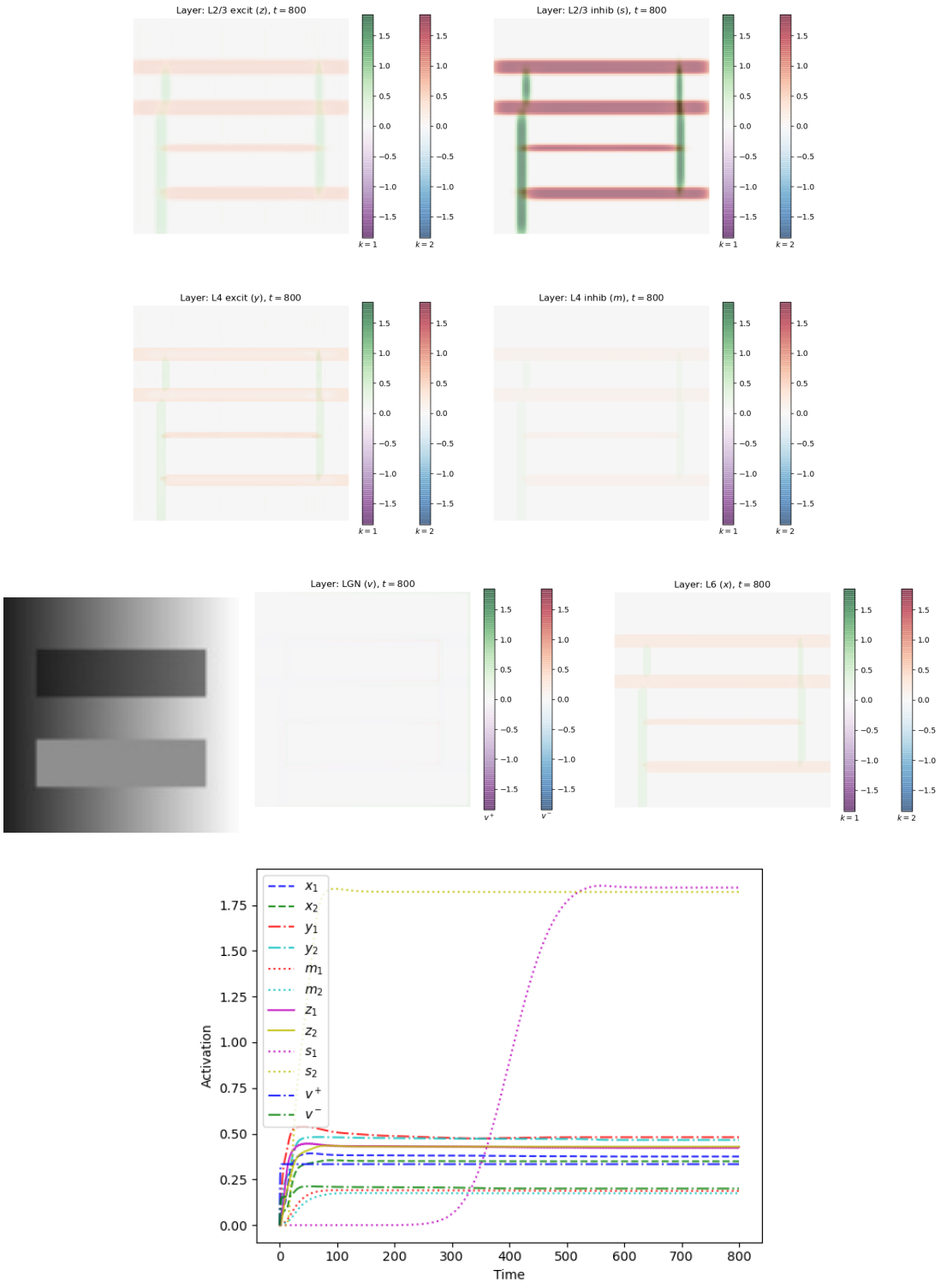

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
Activation
Time

Figure E.30.

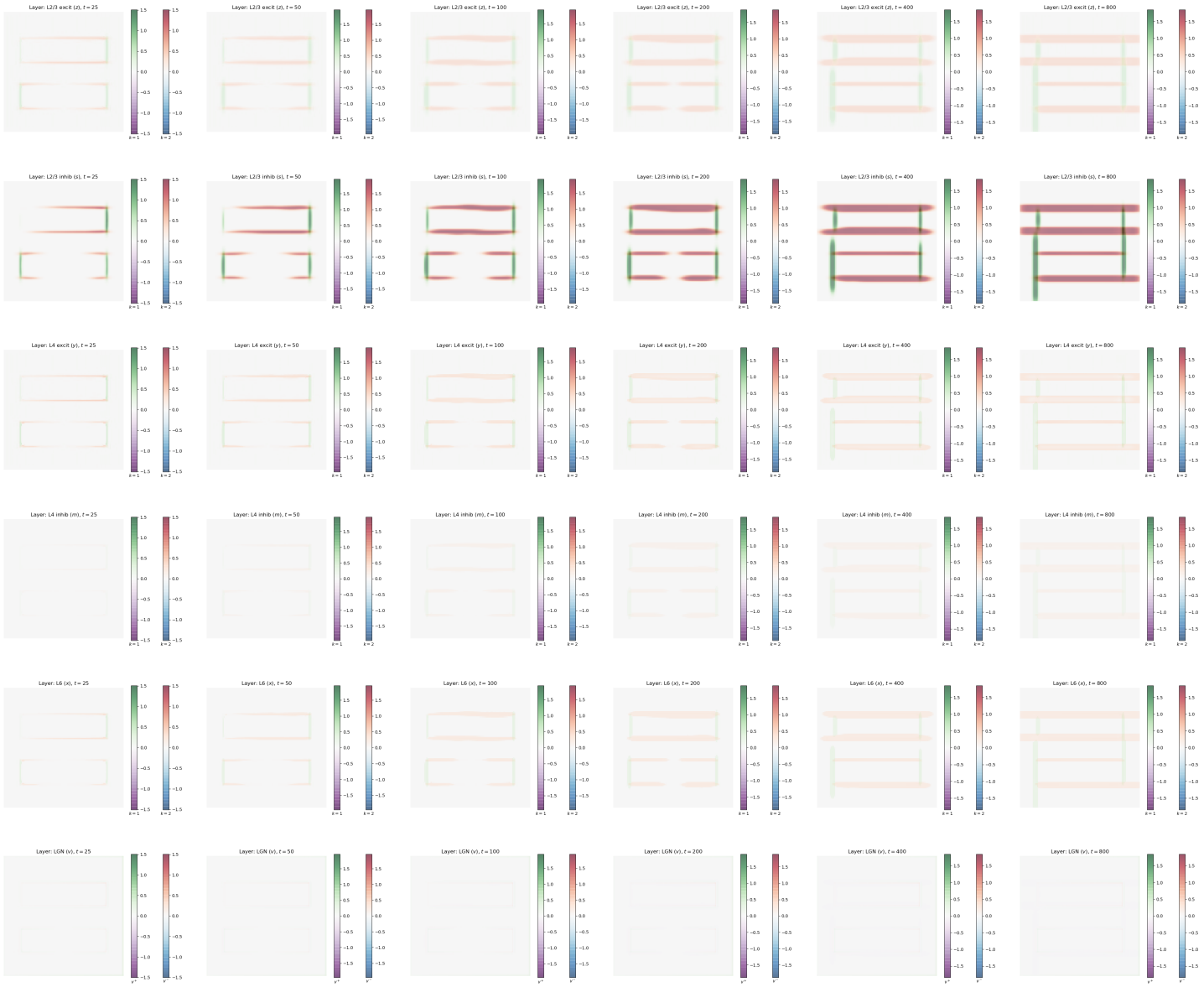

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.31.

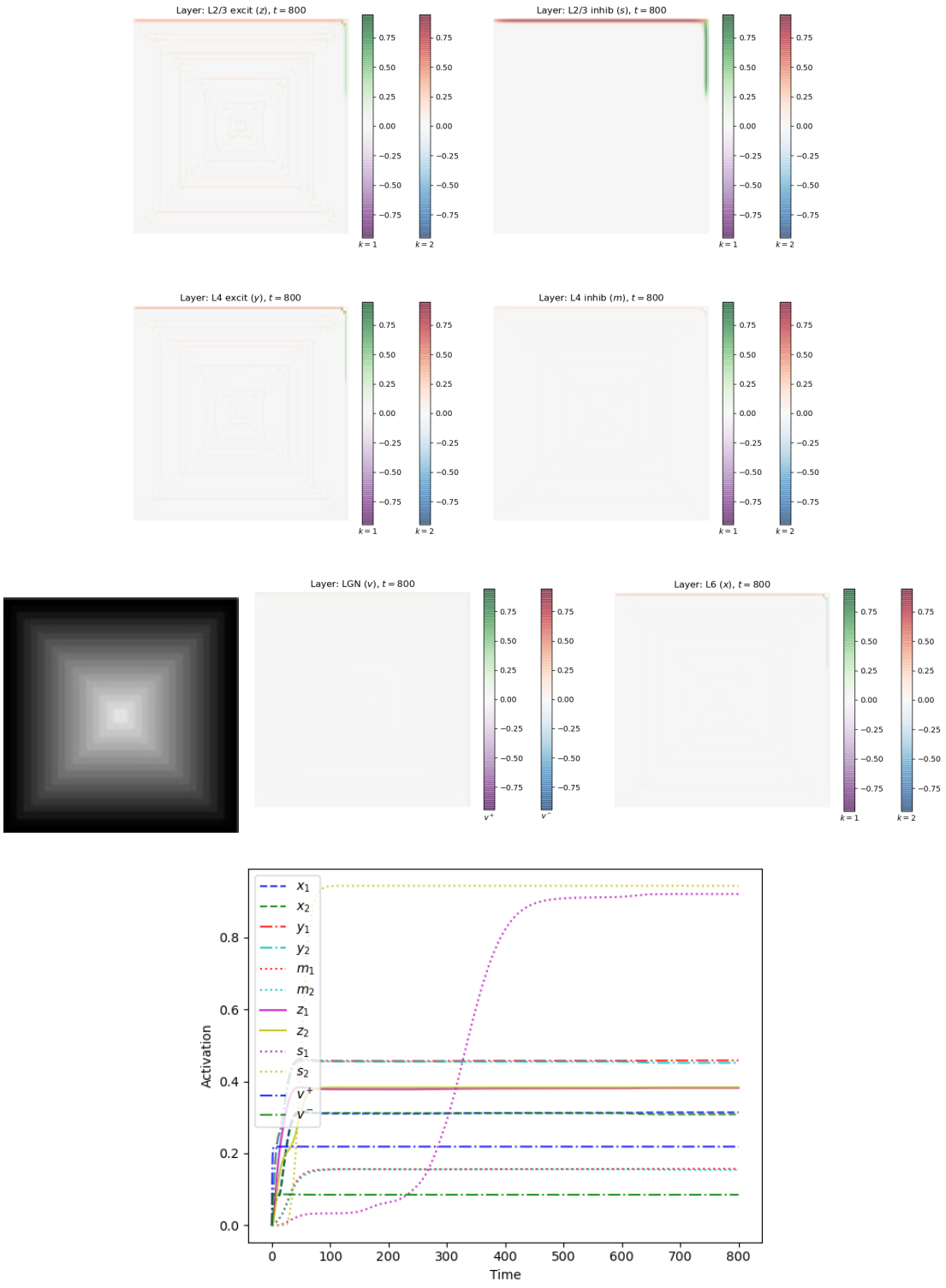

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.32.

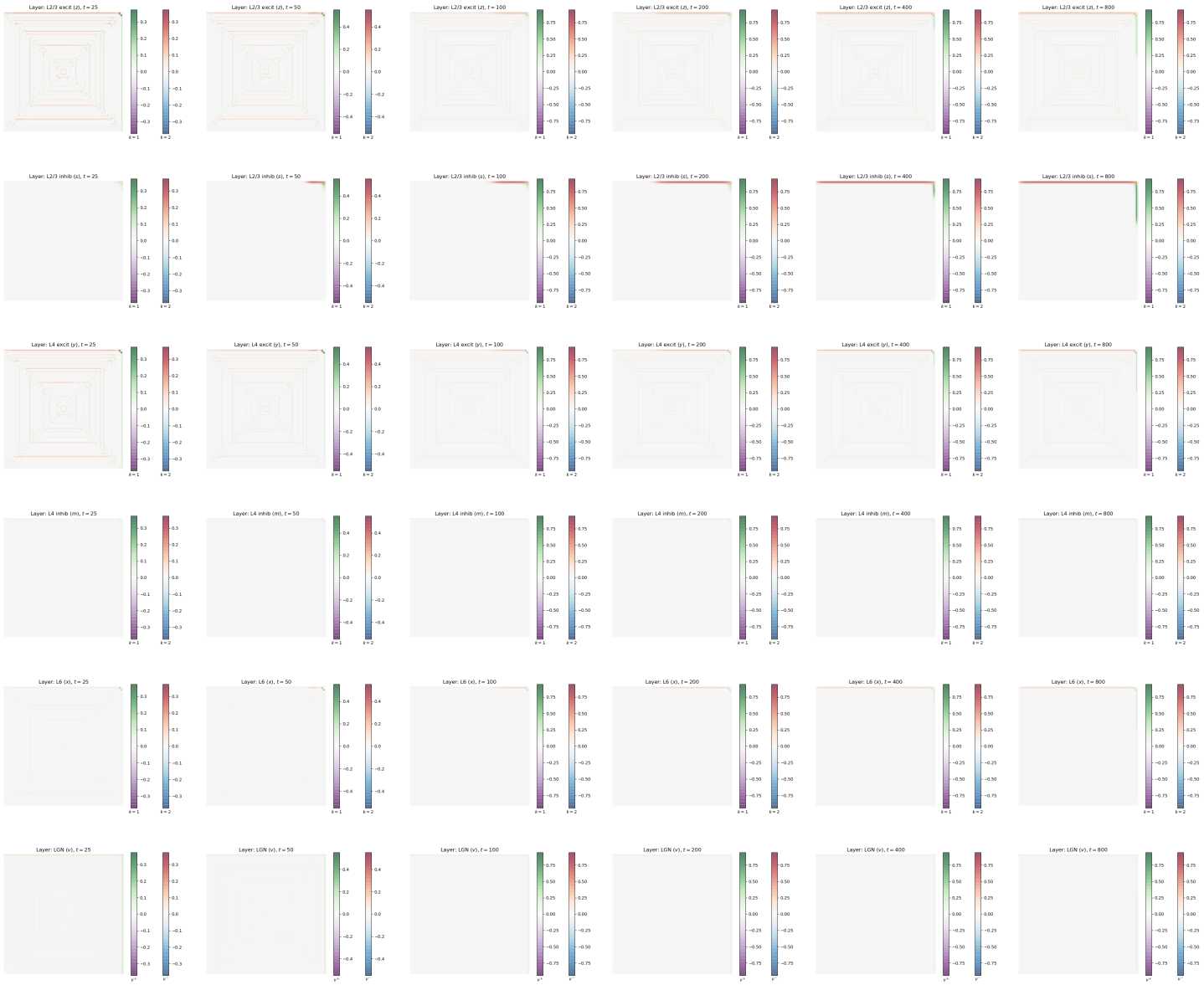

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.33.

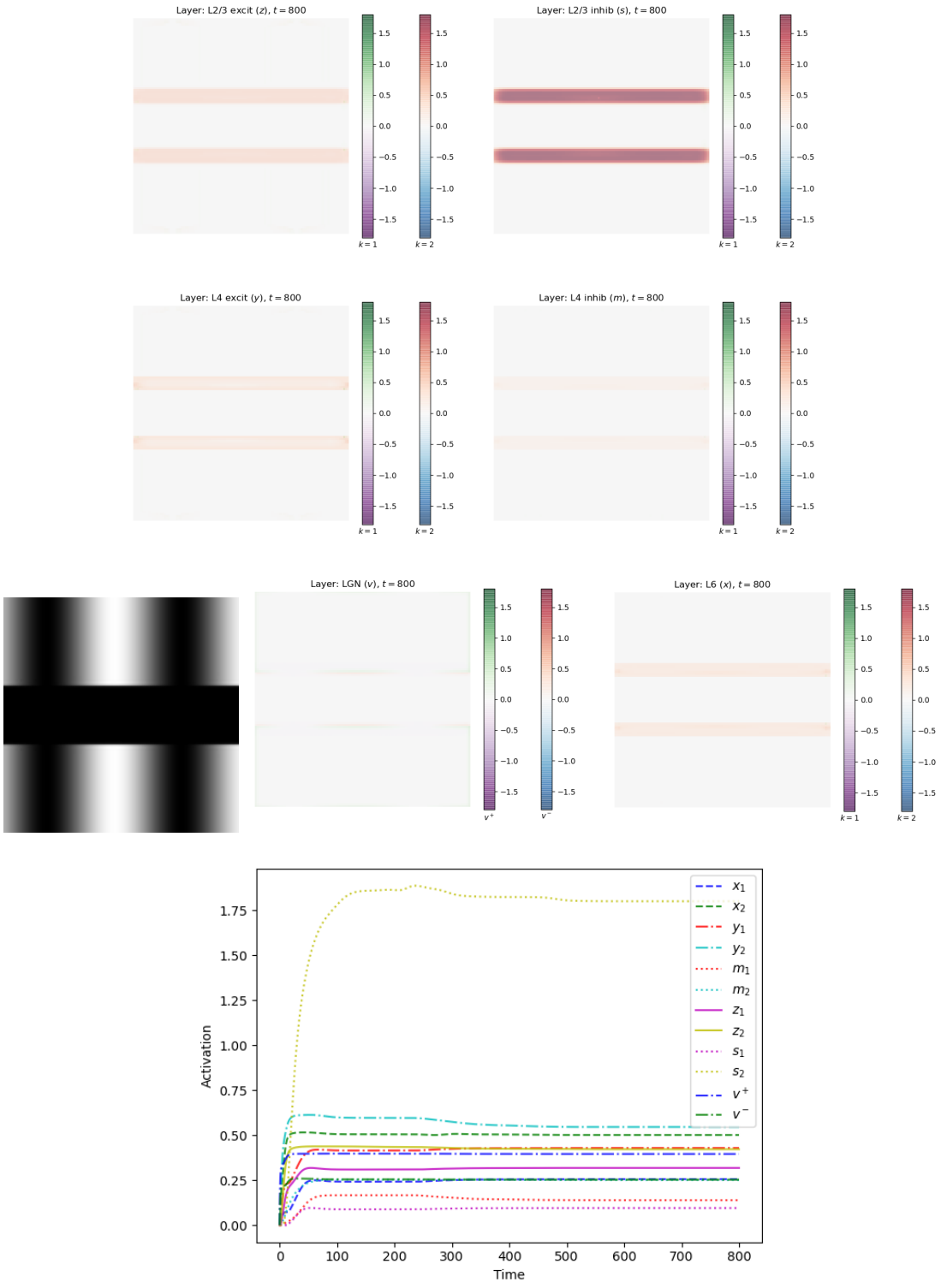

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
Activation
Time
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
v+
v−

Figure E.34.

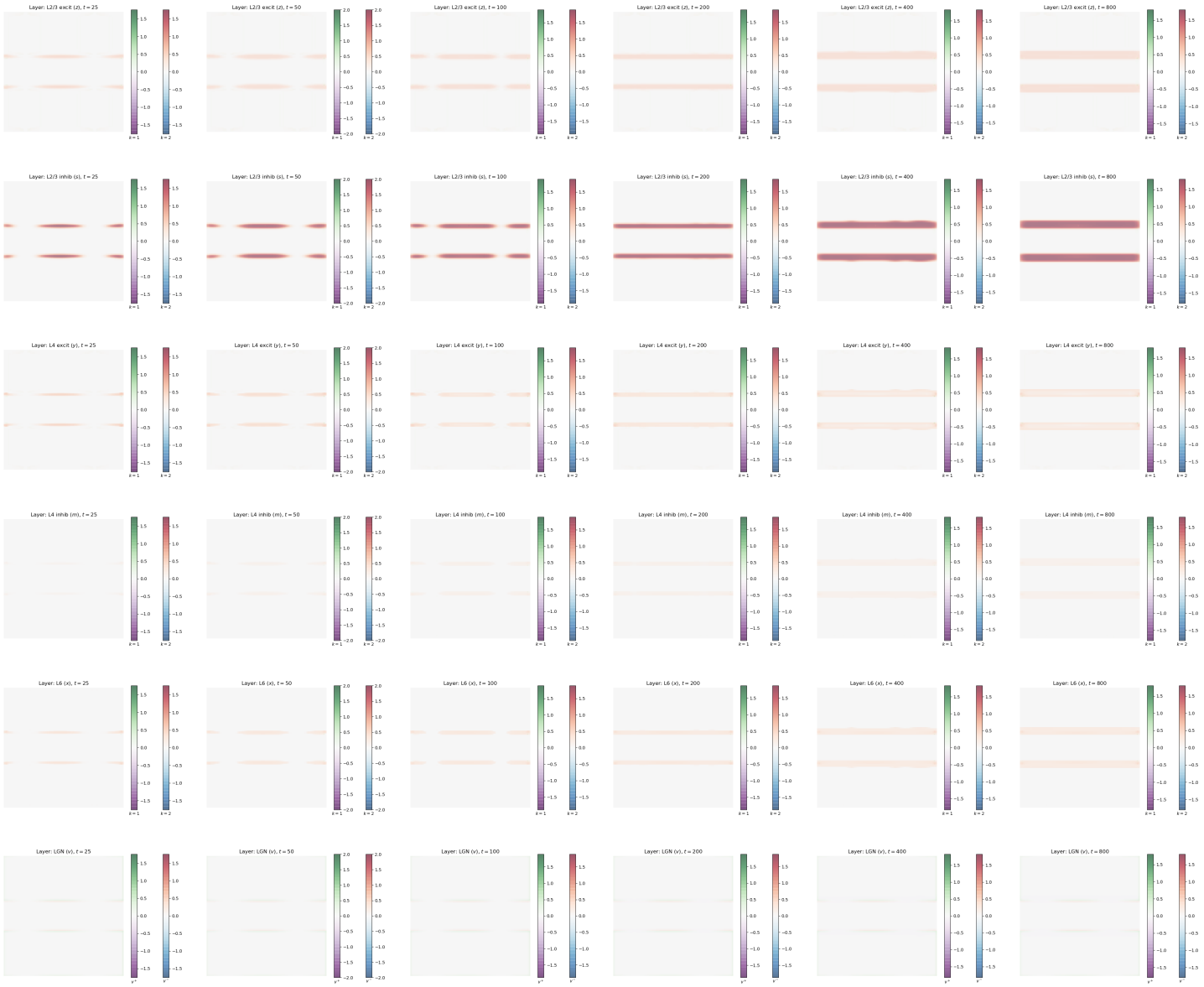

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.35.

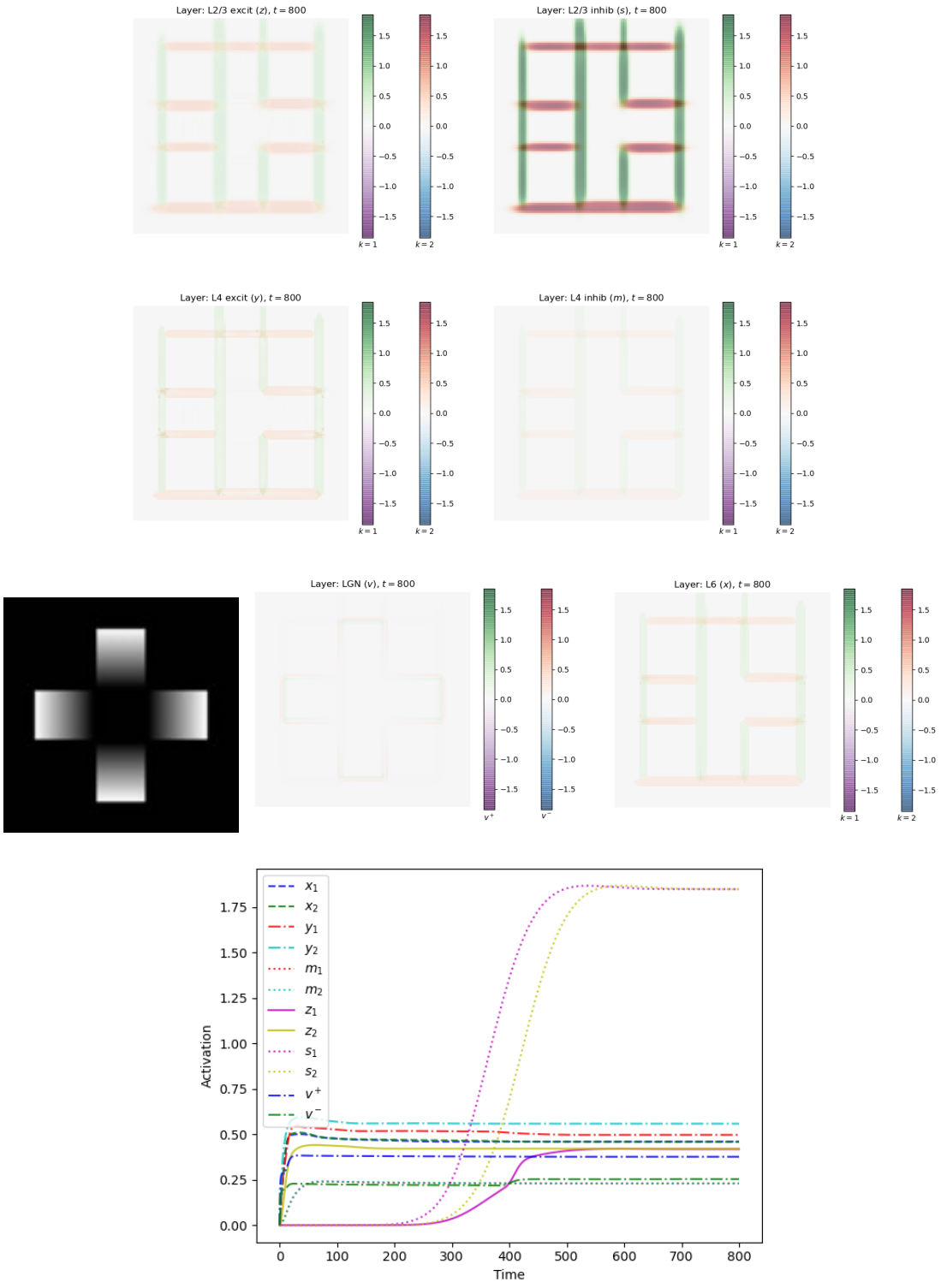

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.36.

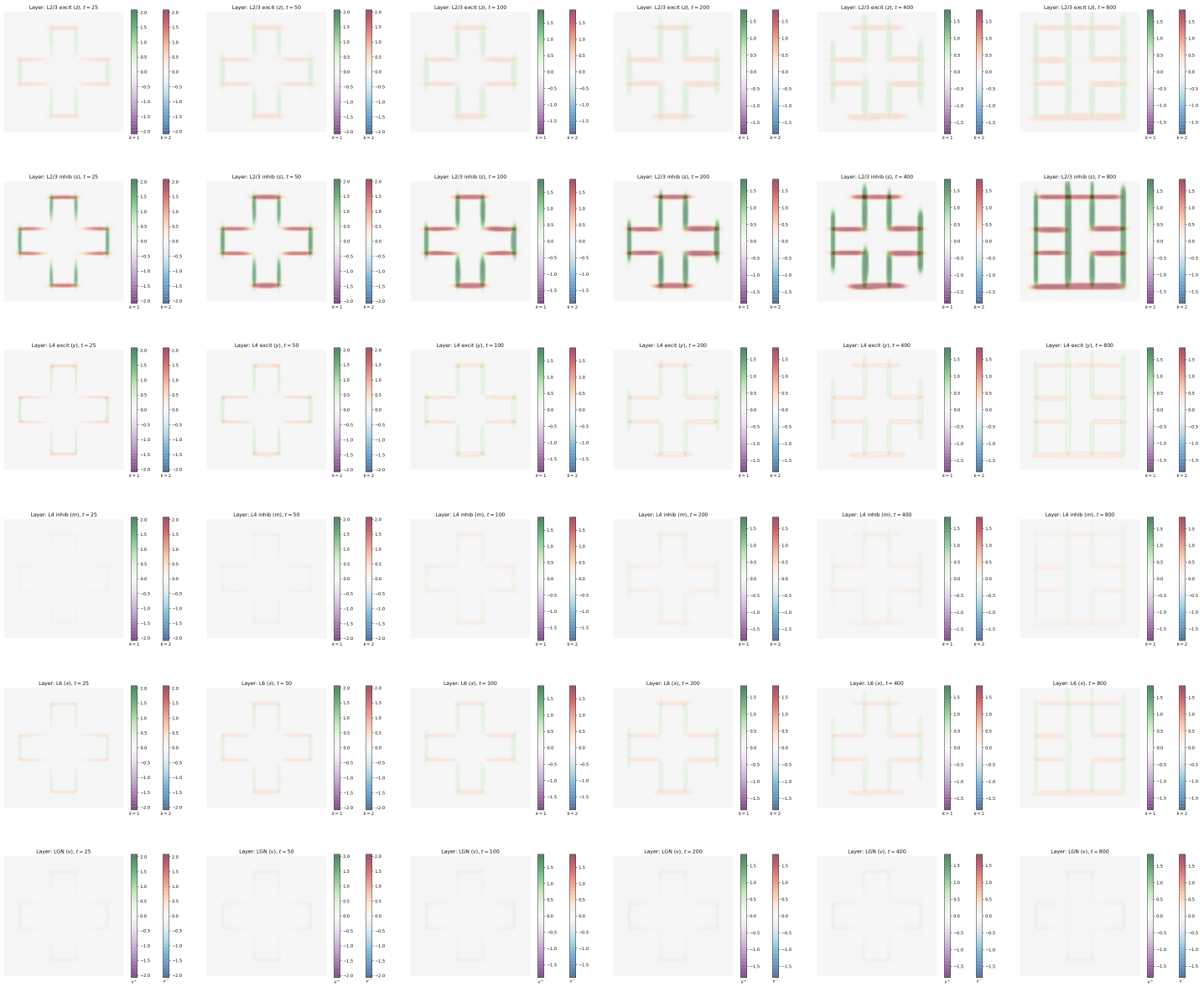

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.37.

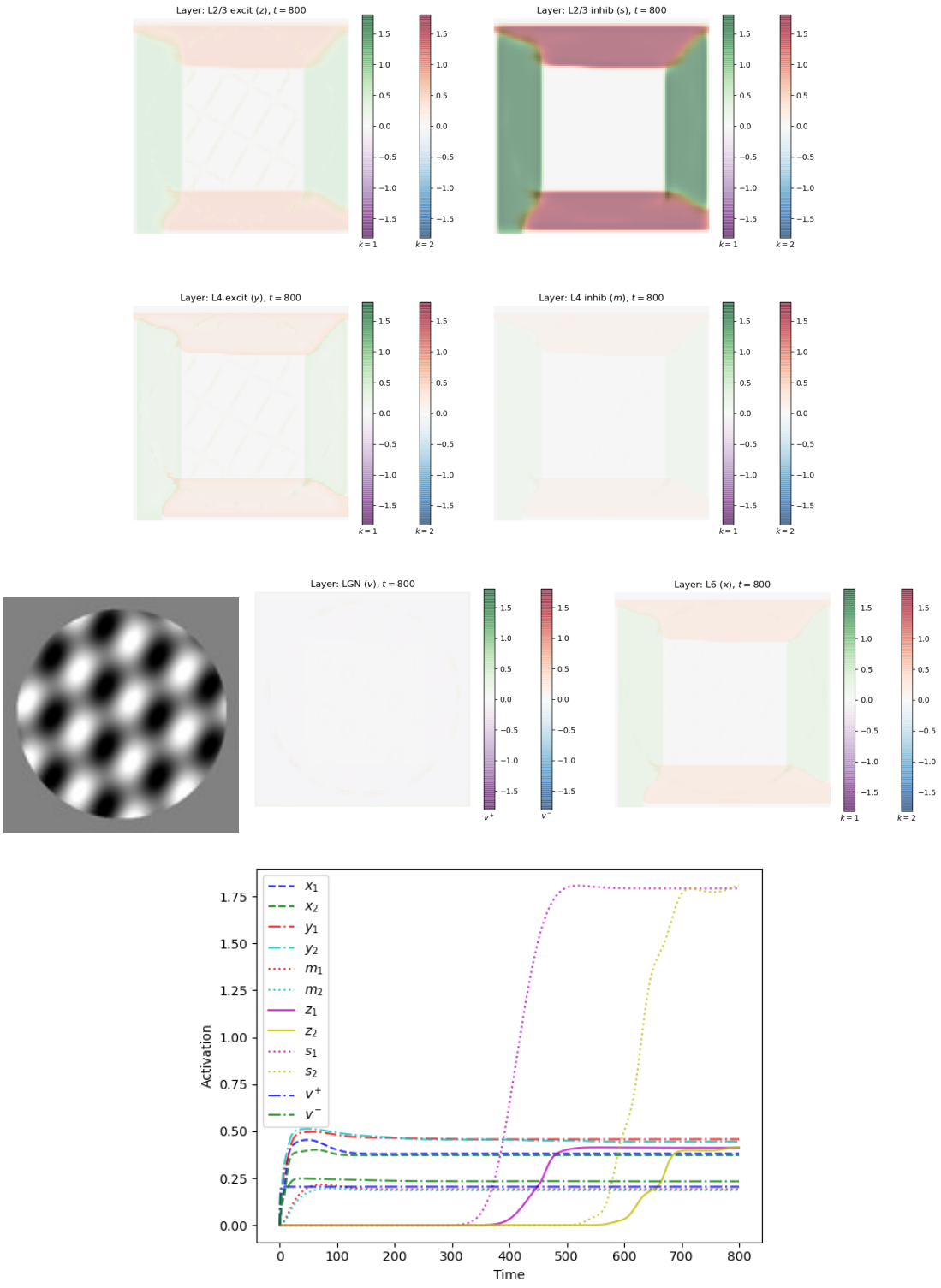

Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 800
Layer: LGN (v), t = 800
Layer: L6 (x), t = 800
k = 1
k = 2
v+
v−
x1
x2
y1
y2
m1
m2
z1
z2
s1
s2
v+
v−
Activation
Time

Figure E.38.

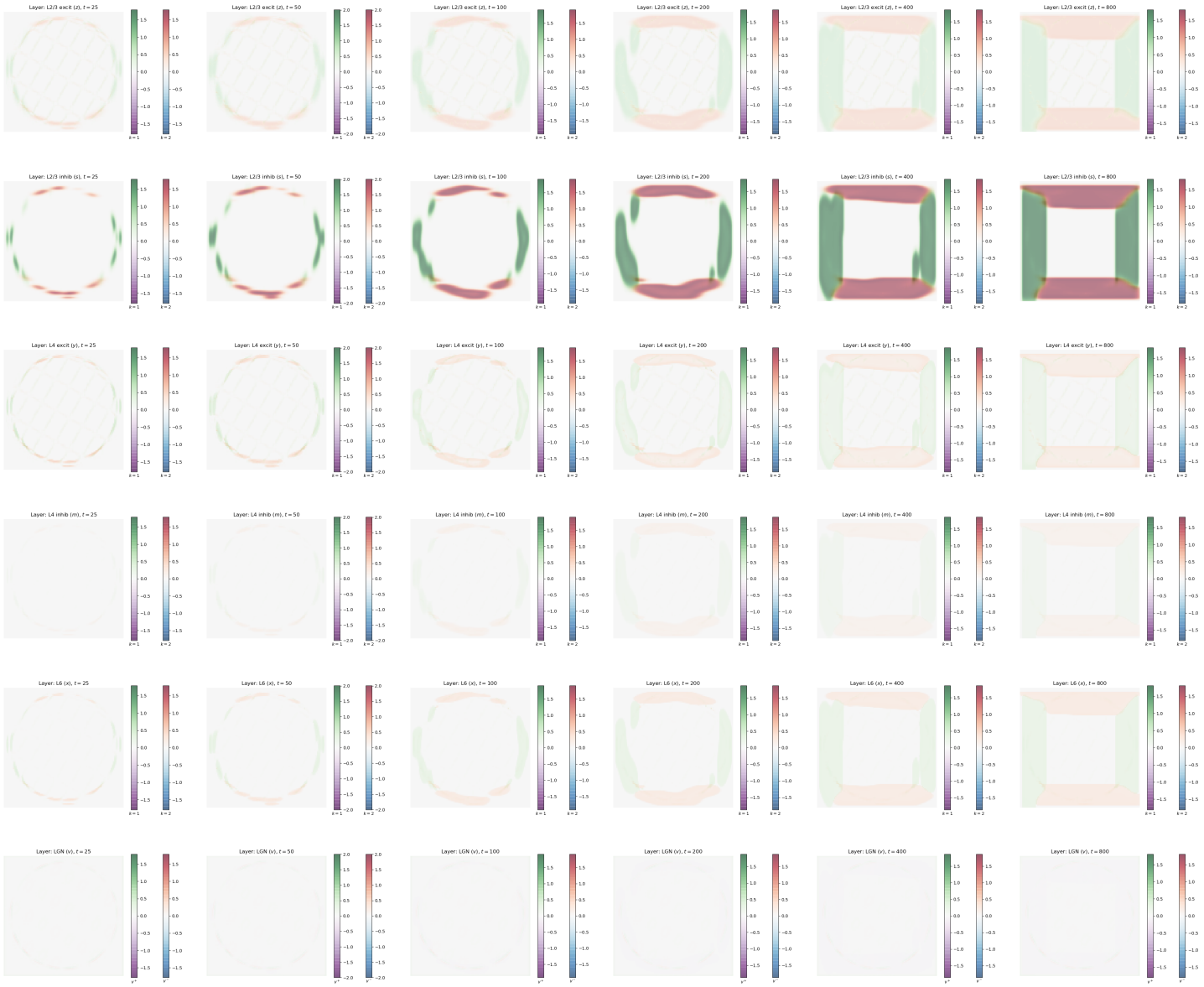

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.39.

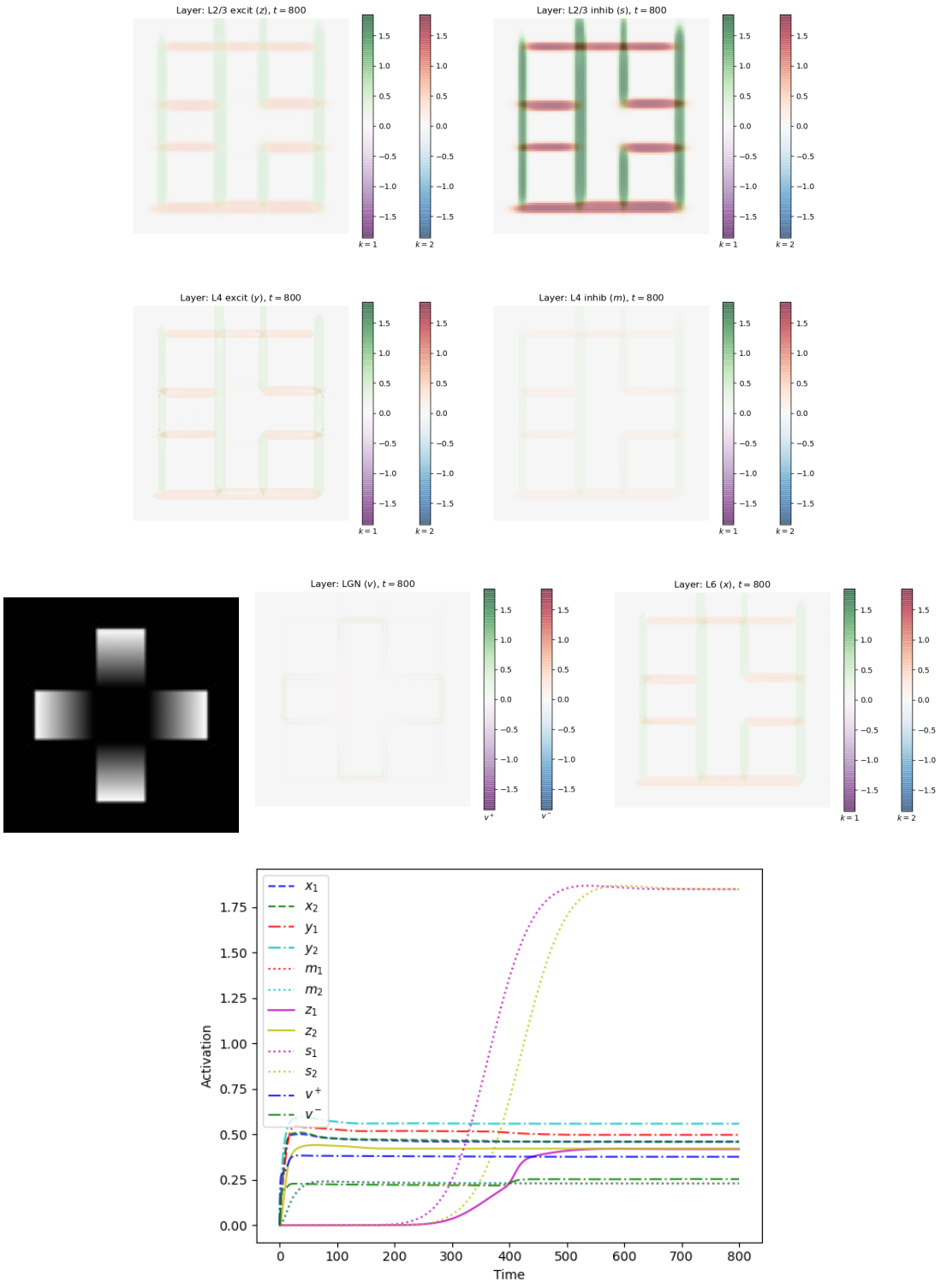

Layer: L2/3 excit (z), $t = 800$
Layer: L2/3 inhib (s), $t = 800$
Layer: L4 excit (y), $t = 800$
Layer: L4 inhib (m), $t = 800$
Layer: LGN (v), $t = 800$
Layer: L6 (x), $t = 800$
$k = 1$
$k = 2$
v^+
v^-
x_1
x_2
y_1
y_2
m_1
m_2
z_1
z_2
s_1
s_2
v^+
v^-
Activation
Time

Figure E.40.

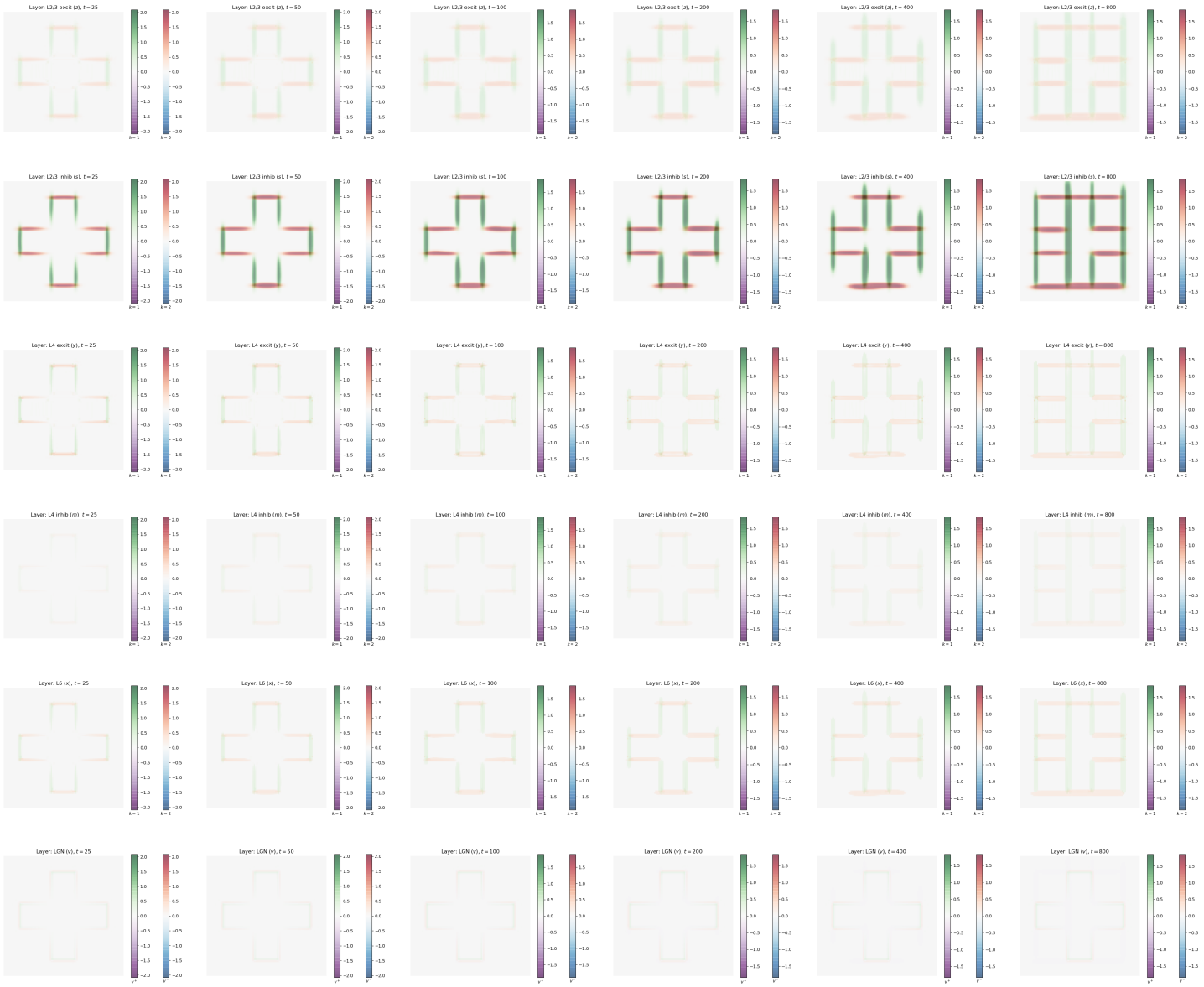

Layer: L2/3 excit (z), t = 25
Layer: L2/3 excit (z), t = 50
Layer: L2/3 excit (z), t = 100
Layer: L2/3 excit (z), t = 200
Layer: L2/3 excit (z), t = 400
Layer: L2/3 excit (z), t = 800
Layer: L2/3 inhib (s), t = 25
Layer: L2/3 inhib (s), t = 50
Layer: L2/3 inhib (s), t = 100
Layer: L2/3 inhib (s), t = 200
Layer: L2/3 inhib (s), t = 400
Layer: L2/3 inhib (s), t = 800
Layer: L4 excit (y), t = 25
Layer: L4 excit (y), t = 50
Layer: L4 excit (y), t = 100
Layer: L4 excit (y), t = 200
Layer: L4 excit (y), t = 400
Layer: L4 excit (y), t = 800
Layer: L4 inhib (m), t = 25
Layer: L4 inhib (m), t = 50
Layer: L4 inhib (m), t = 100
Layer: L4 inhib (m), t = 200
Layer: L4 inhib (m), t = 400
Layer: L4 inhib (m), t = 800
Layer: L6 (x), t = 25
Layer: L6 (x), t = 50
Layer: L6 (x), t = 100
Layer: L6 (x), t = 200
Layer: L6 (x), t = 400
Layer: L6 (x), t = 800
Layer: LGN (v), t = 25
Layer: LGN (v), t = 50
Layer: LGN (v), t = 100
Layer: LGN (v), t = 200
Layer: LGN (v), t = 400
Layer: LGN (v), t = 800

Figure E.41.

Appendix F

Source code

F.I. SOURCE CODE

The source code developed during the project may be accessed from https://github.com/nialltc/MEngProject.jl

The repository has the following structure:

- `src`
  - contains *Julia* sourcecode developed for the project
    - `MEngProject.jl`
      - *Julia* package file for the project
    - `LaminartInitFunc.jl`
      - contains functions that converts input to u^+ and stores in a named tuple
    - `LaminartFunc.jl`
      - contains functions to create function for use during solving
    - `LaminartEqConv.jl`
      - contains model equation functions implemented with `NNlib.jl.conv`
    - `LaminartEqImfilter.jl`
      - contains model equation functions implemented with `imfilter`
    - `LaminartEqImfilterGPU_FFT.jl`
      - contains model equation functions implemented with `imfilter` on GPGPU using FFT algorithm
    - `LaminartEqImfilterGPU_FIR.jl`
      - contains model equation functions implemented with `imfilter`on GPGPU using FIR algorithm
    - `LaminartEqImfilterGPU_IIR.jl`
      - contains model equation functions implemented with `imfilter`on GPGPU using IIR algorithm
    - `LaminartEqImfilter_old.jl`
      - contains old version of model equation functions
    - `LaminartKernels.jl`
      - contains functions used by `LaminartInitFunc.jl` to generate kernels
    - `Parameters.jl`
      - holds values for model parameters for `LaminartInitFunc.jl`
    - `Utils.jl`
      - contains plotting functions
- `scripts`
  - contains scripts used to run tests and benchmarks
- `notebooks`
  - contains *Jupyter* notebooks used during development, testing and benchmarking
- `data`
  - contains input images
- `plots`
  - contains output plots from development and testing