Operations in Multi-Dimensional Neural Fields

KogWis 2014

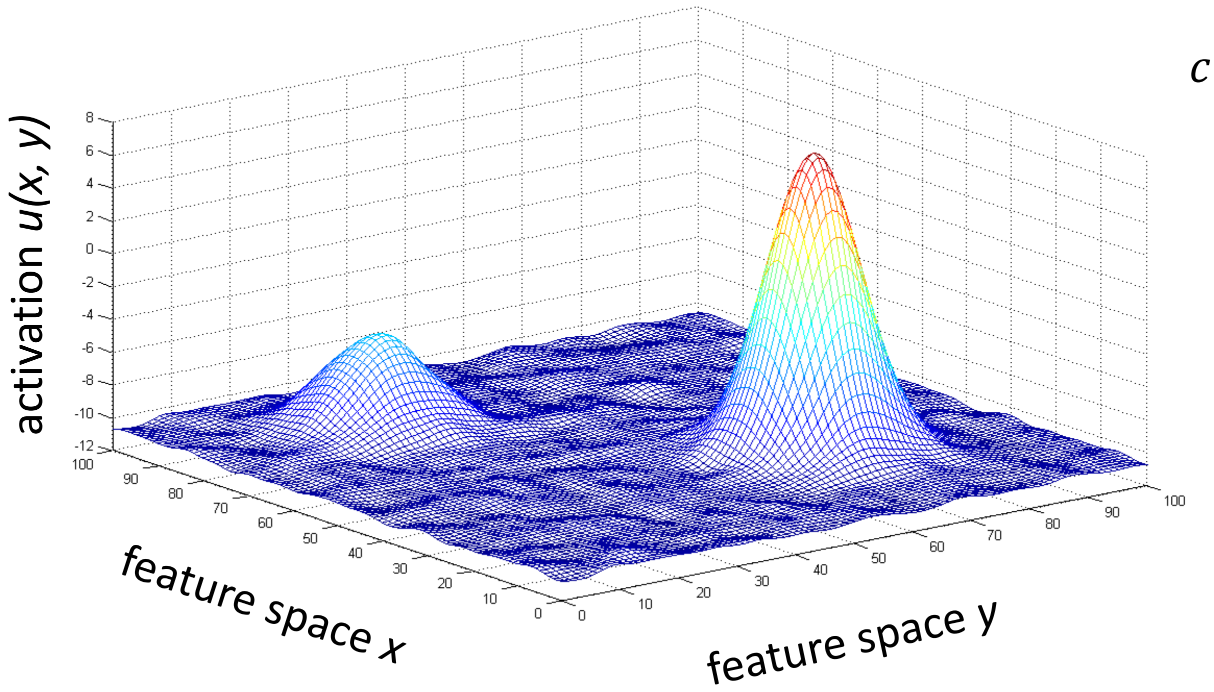
Sebastian Schneegans

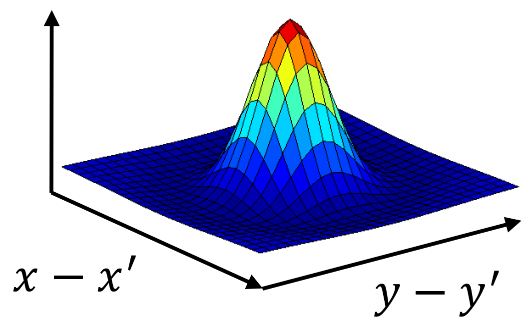
Institut für Neuroinformatik

Ruhr-Universität Bochum

sebastian.schneegans@ini.rub.de

Multi-dimensional fields

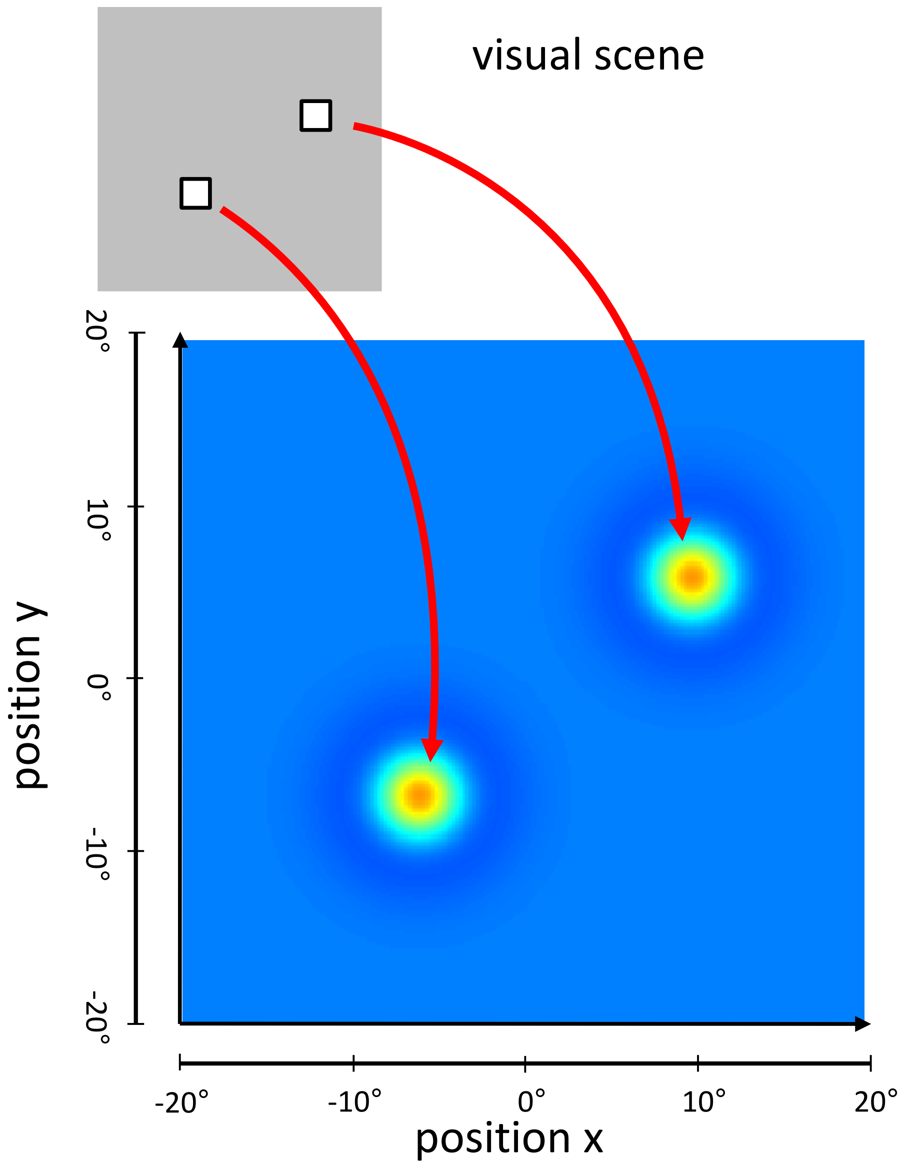


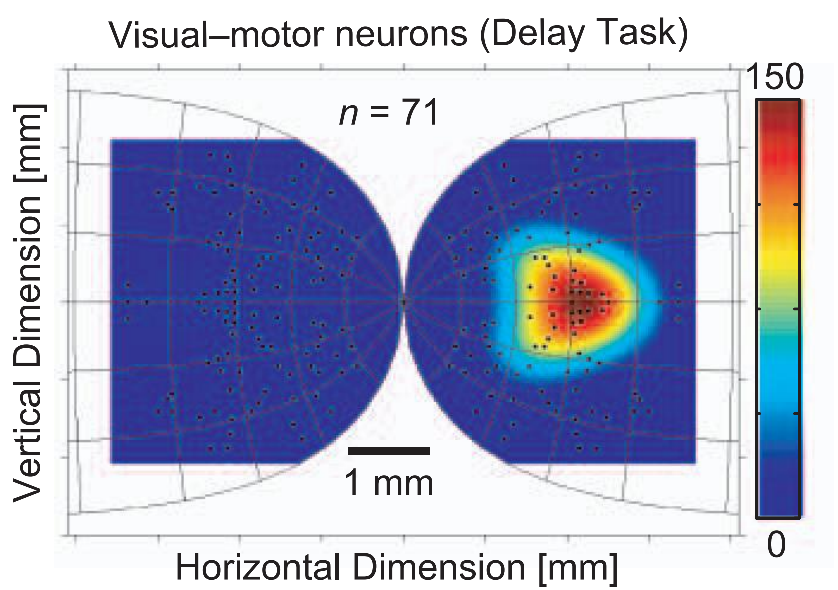


$$\tau\dot{u}(x,y) = -u(x,y) + h + s(x,y) + \iint c(x-x', y-y')g\big(u(x',y')\big)dx'dy'$$

- extension to multi-dimensional feature spaces mathematically straightforward
- requires interaction kernel of the same dimensionality

Multi-dimensional feature spaces



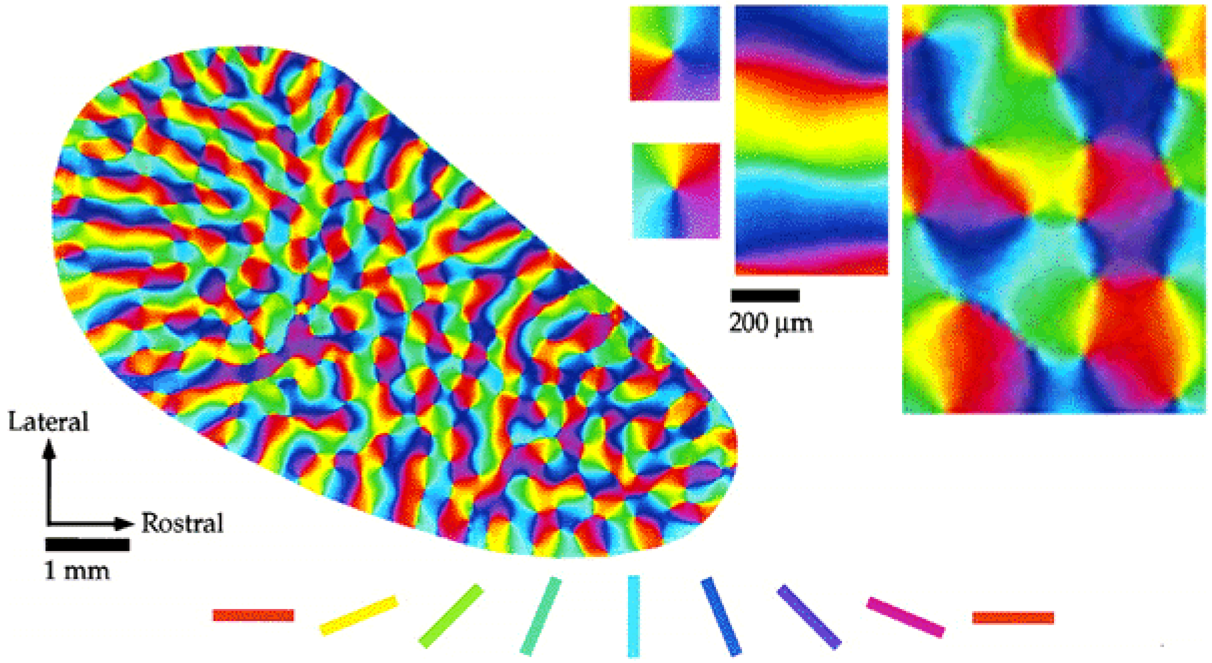


neural activity in superior colliculus
[Marino, Trappenberg, Dorris, Munoz 2012]

- some feature spaces are inherently multi-dimensional, e.g. visual space (2D)
- neural representations e.g. in superior colliculus (saccade planning)

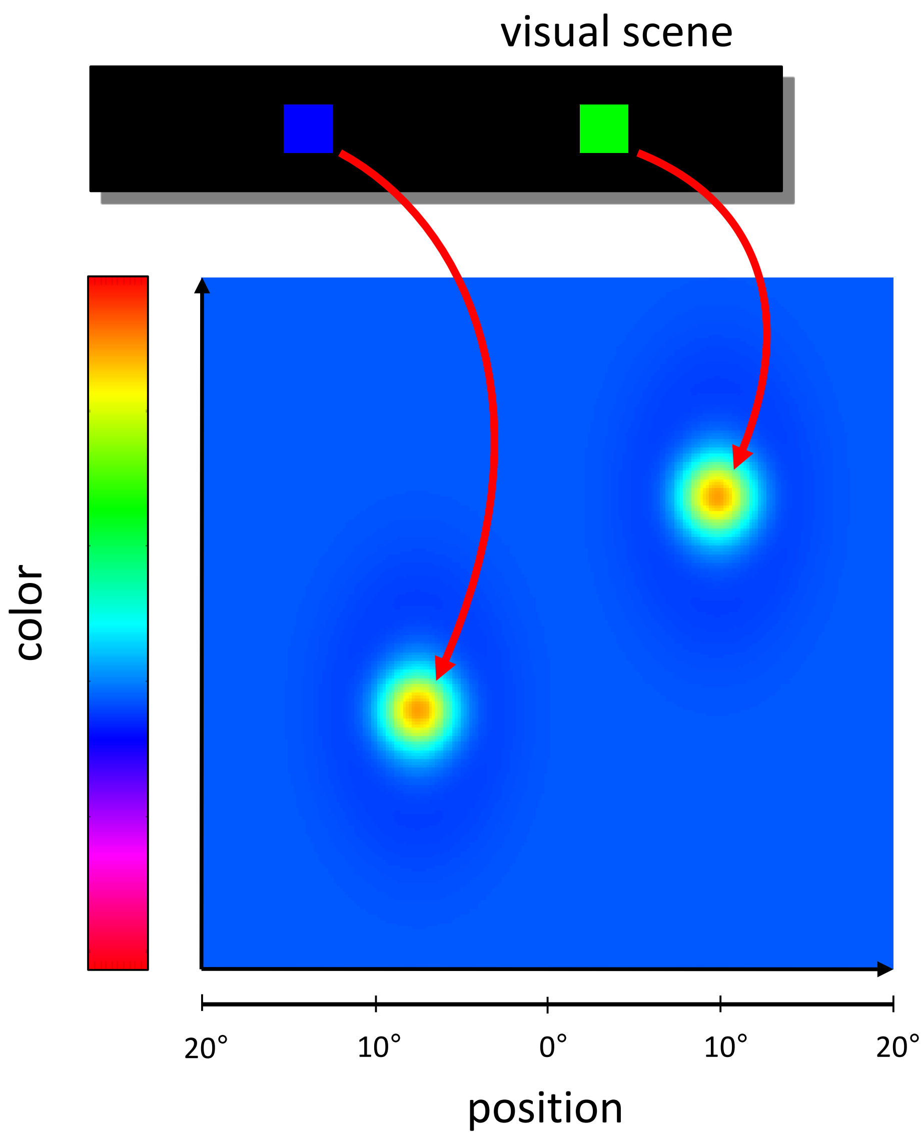
Multi-dimensional feature spaces

- multi-dimensional feature spaces can also combine qualitatively different features
- example: early visual cortex, neurons with localized spatial receptive fields and sensitivity to surface features (orientation, spatial frequency, color, …)



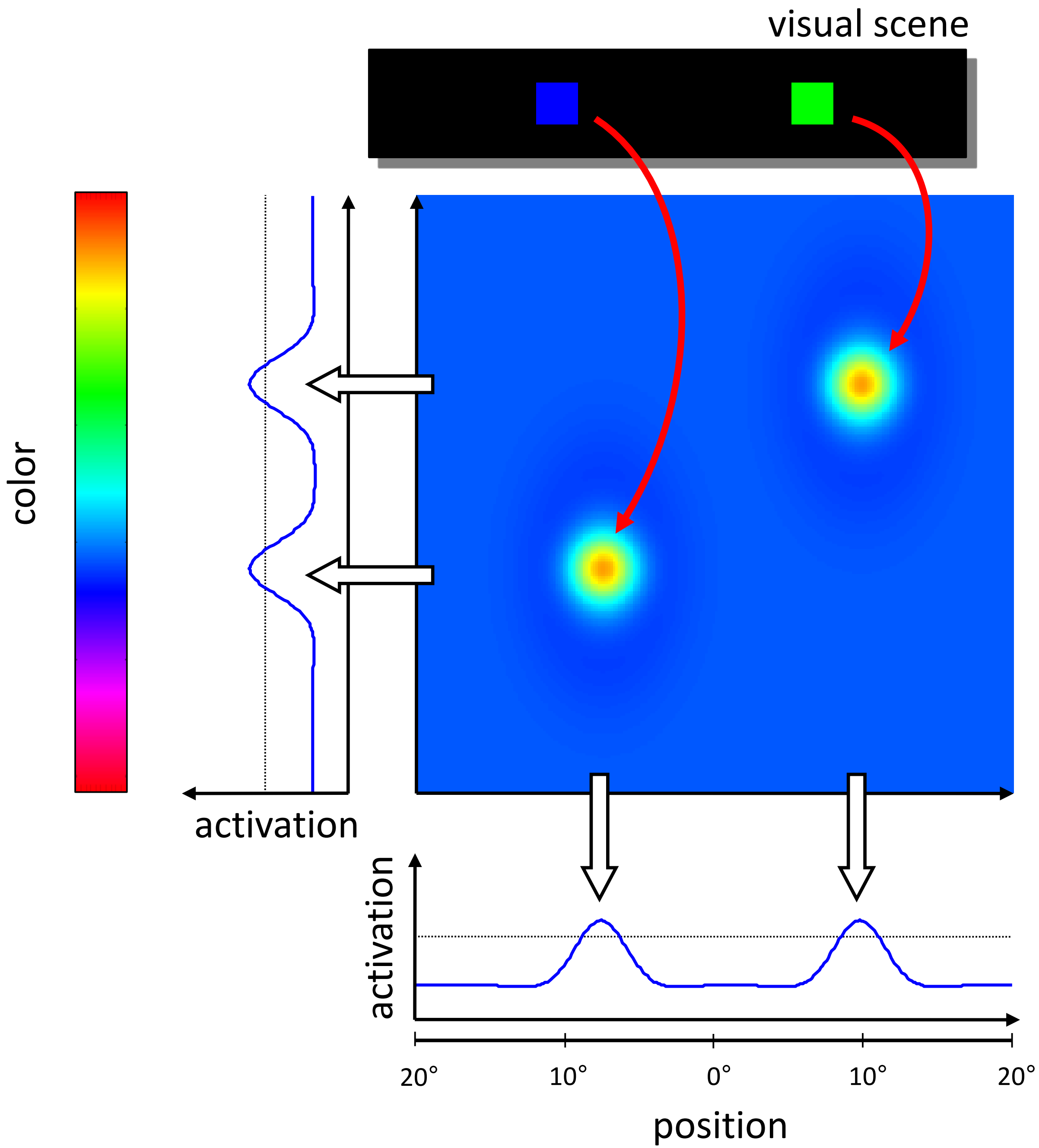
orientation map in tree shrew visual cortex [Alexander et al. 1999]

Combining features in multi-dimensional fields



- neural field defined over combination of feature spaces (space × color)
- not aimed to capture spatial arrangement of neurons in the cortex
- visual stimuli provide localized inputs

Reading out from 2D fields

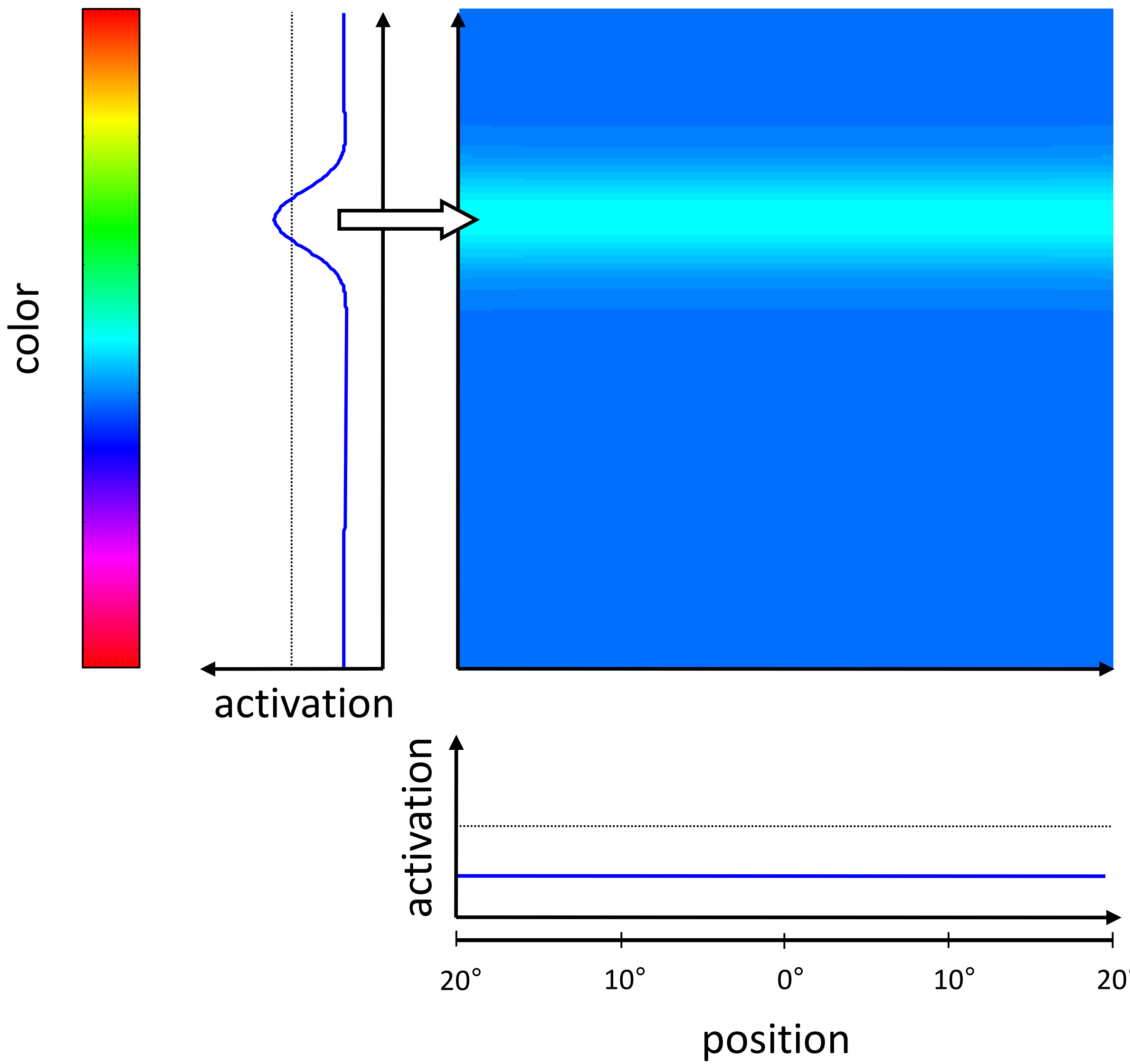


- 2D fields can interact with 1D fields
- first operation: read out of one feature dimension, integrate over discarded dimensions, e.g.

$$I_s(x) = \int f\big(u_v(x,y)\big)dy$$

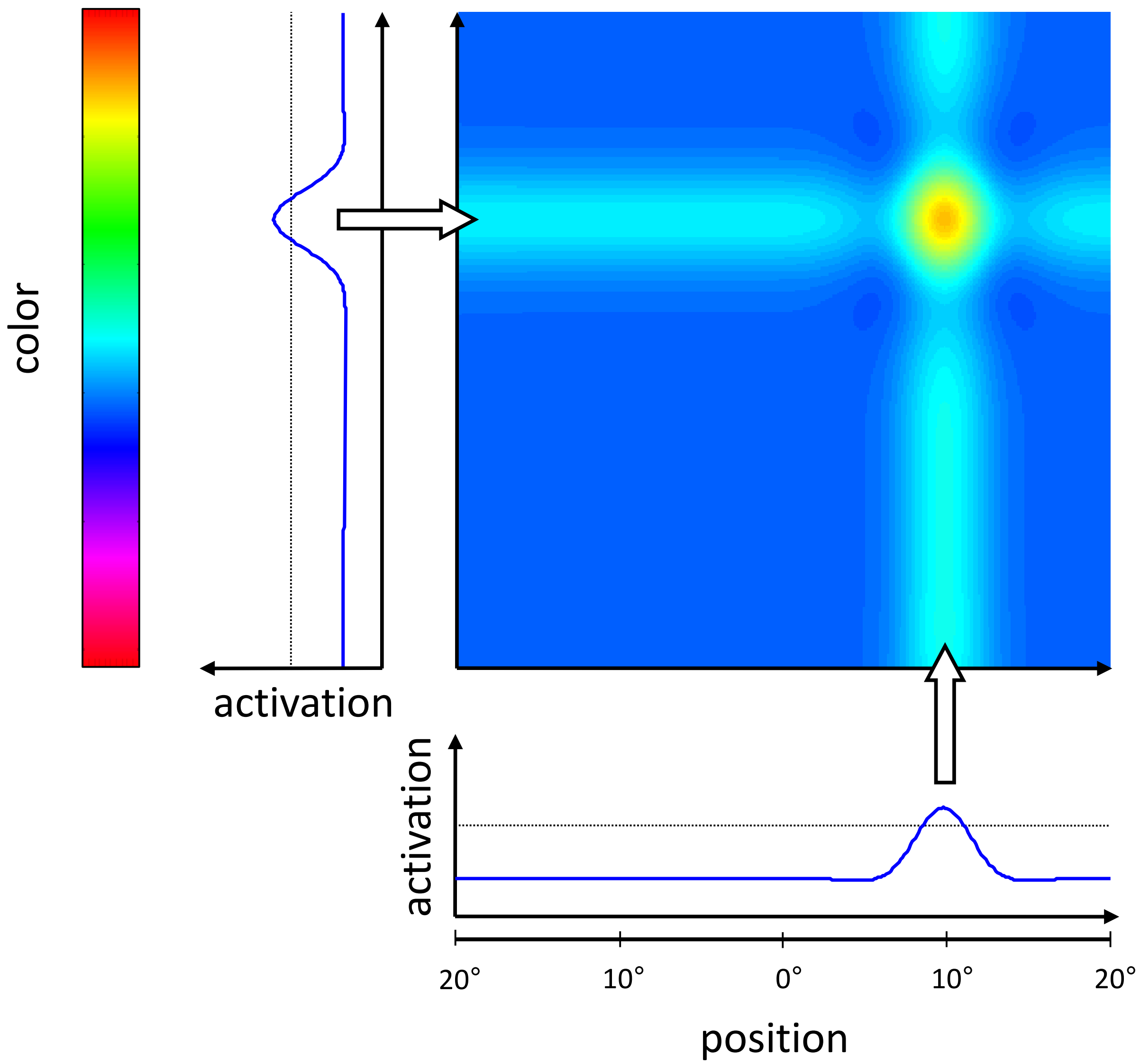
- often additional Gaussian convolution in projection for smoothness

Projections to 2D fields



- projection from 1D to 2D: ridge input
- does not specify a location in the 2nd dimension, does not typically induce a peak

Projections to 2D fields



- intersections of ridges can induce a peak and produce a combined representation of multiple features

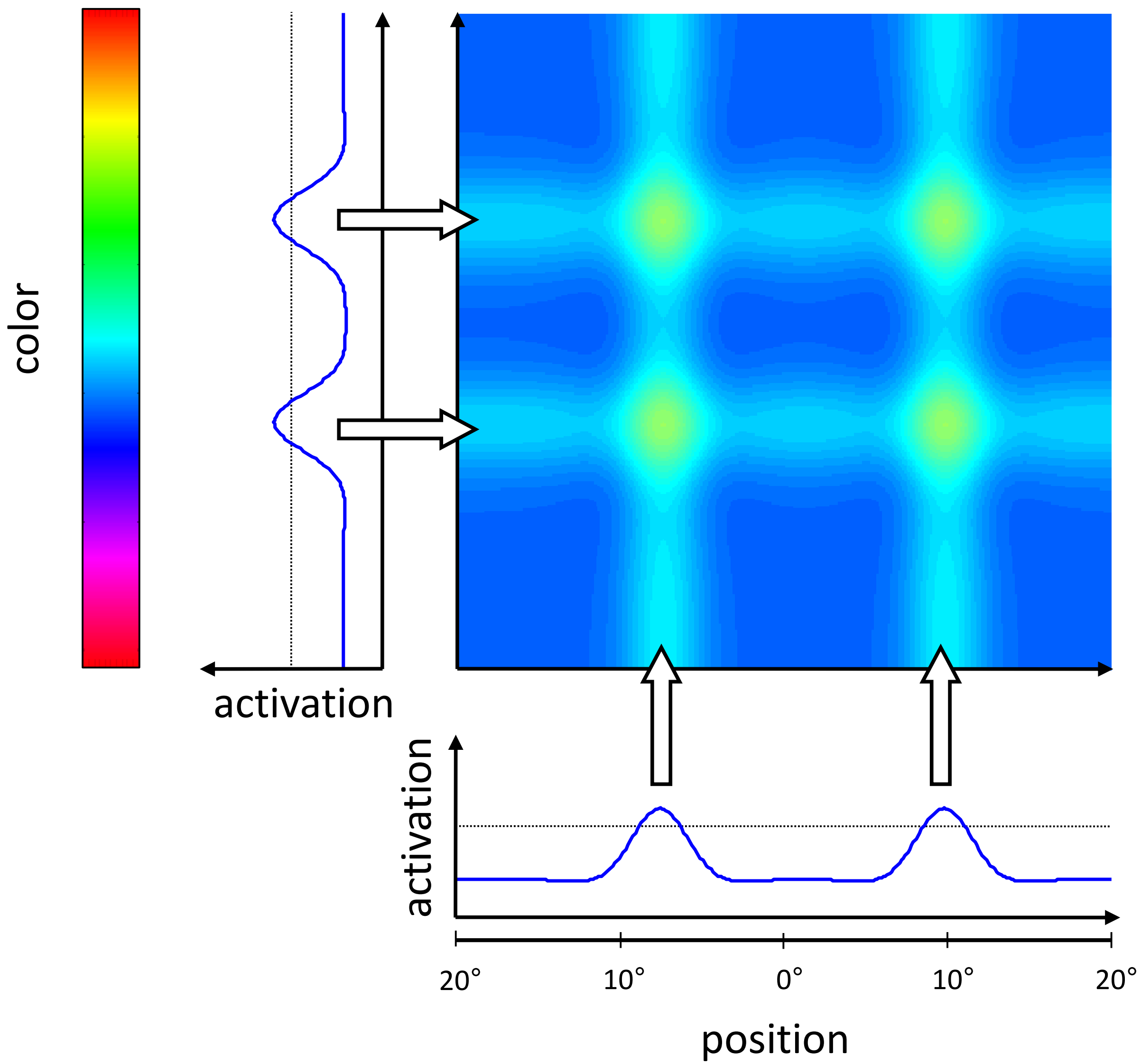
Combined vs. separate representations

separate low-dimensional representations

- are much more compact (computationally less expensive / fewer neurons) – at sampling rate of 100 neurons per dimension, 200 neurons for two 1D fields, 10000 neurons for one 2D field)
- can represent individual feature values with the same precision/reliability as a 2D field

So why use 2D fields at all?

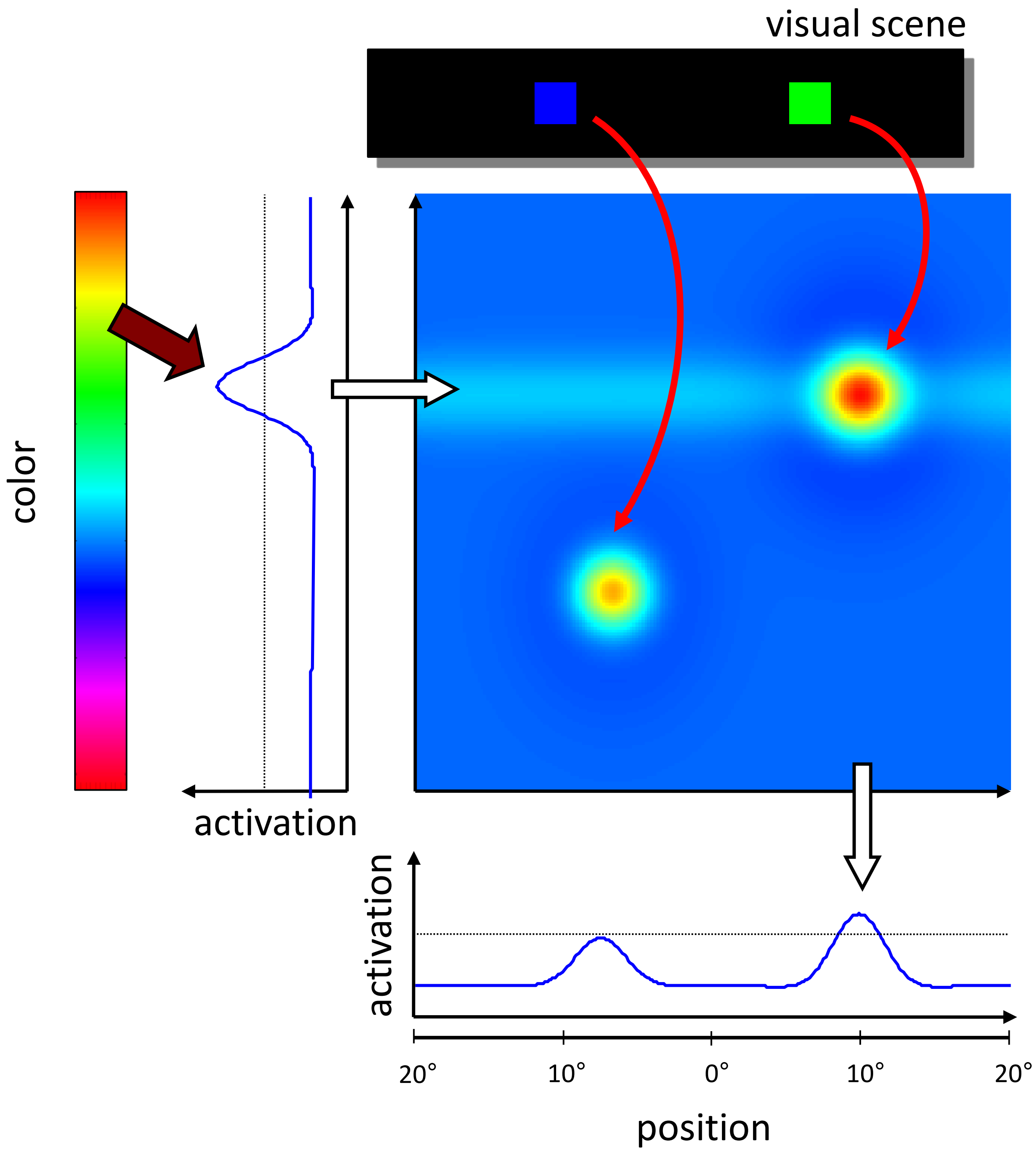
Feature conjunctions



- low-dimensional representations do not capture feature conjunctions (binding problem)
- multiple ridge inputs can produce spurious peaks

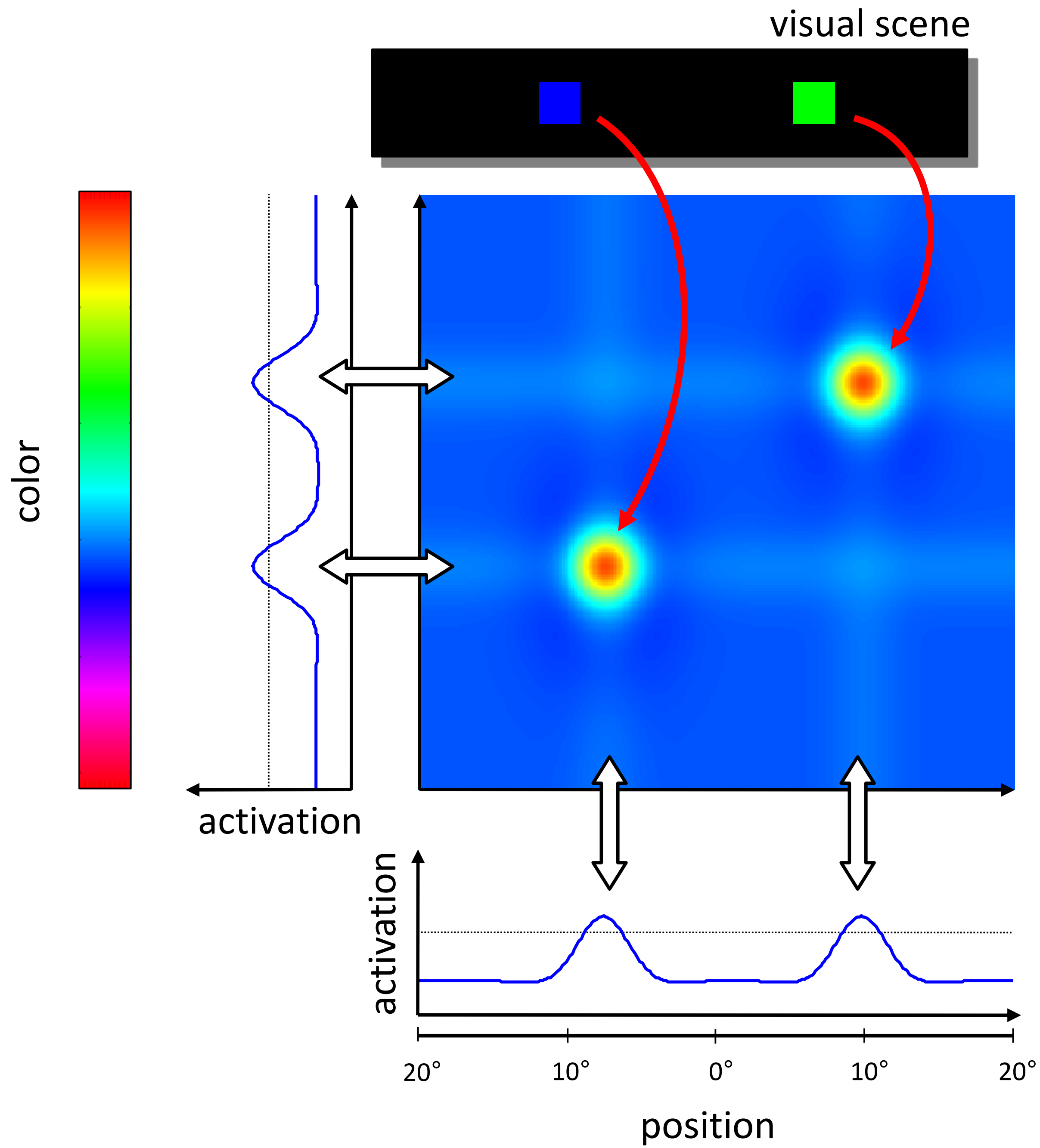
➢ need combinations of low- and high-dimensional field for efficient architectures

Visual search



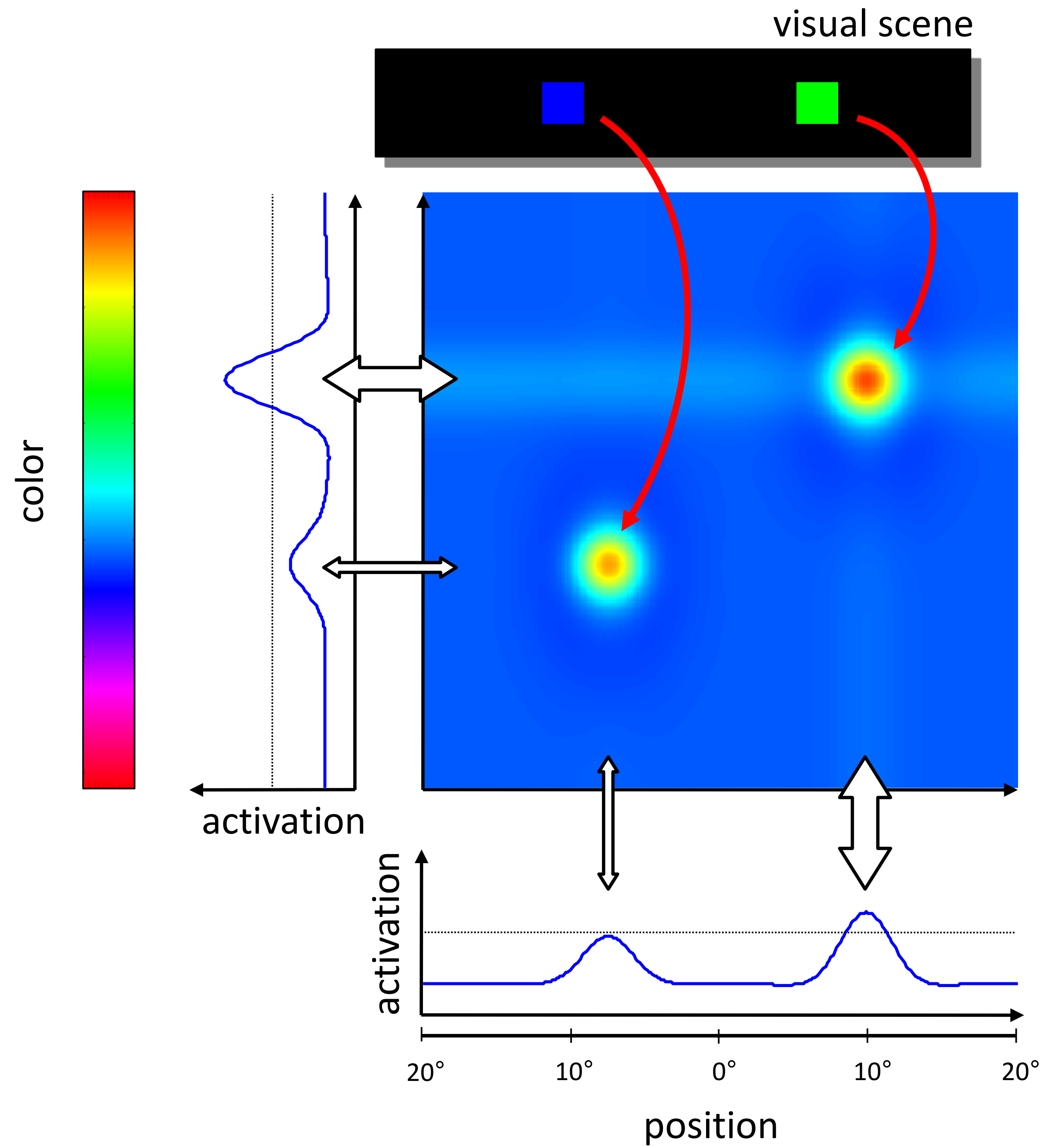
- if localized peaks are present in the 2D field, ridge input can be used to select one of them
- read-out along the 2nd dimension then allows to determine the associated feature

Joint selection with bidirectional projections



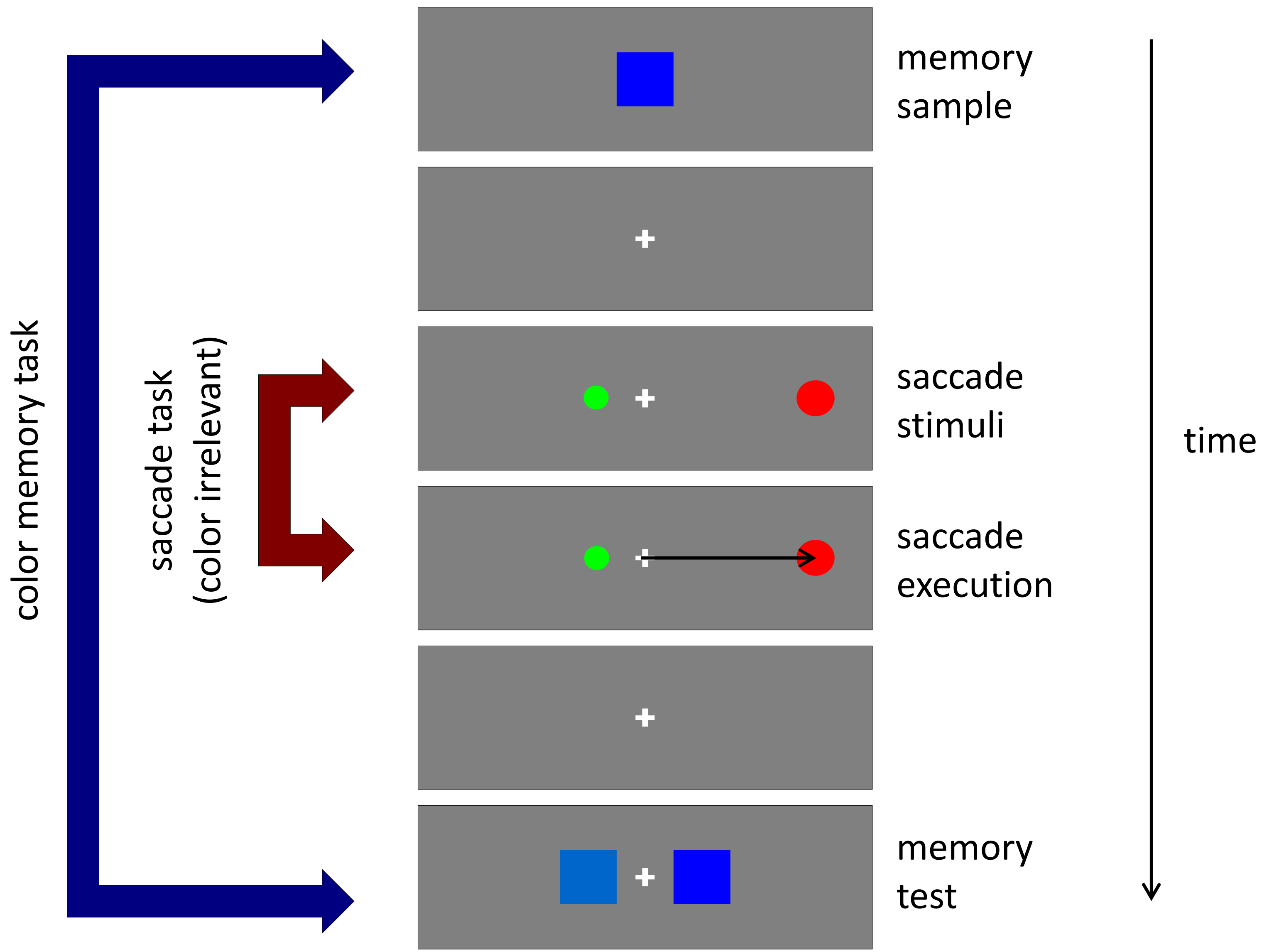
- bidirectional projections allow coupled selection in 1D fields
- can be biased by input to either 1D field

Joint selection with bidirectional projections

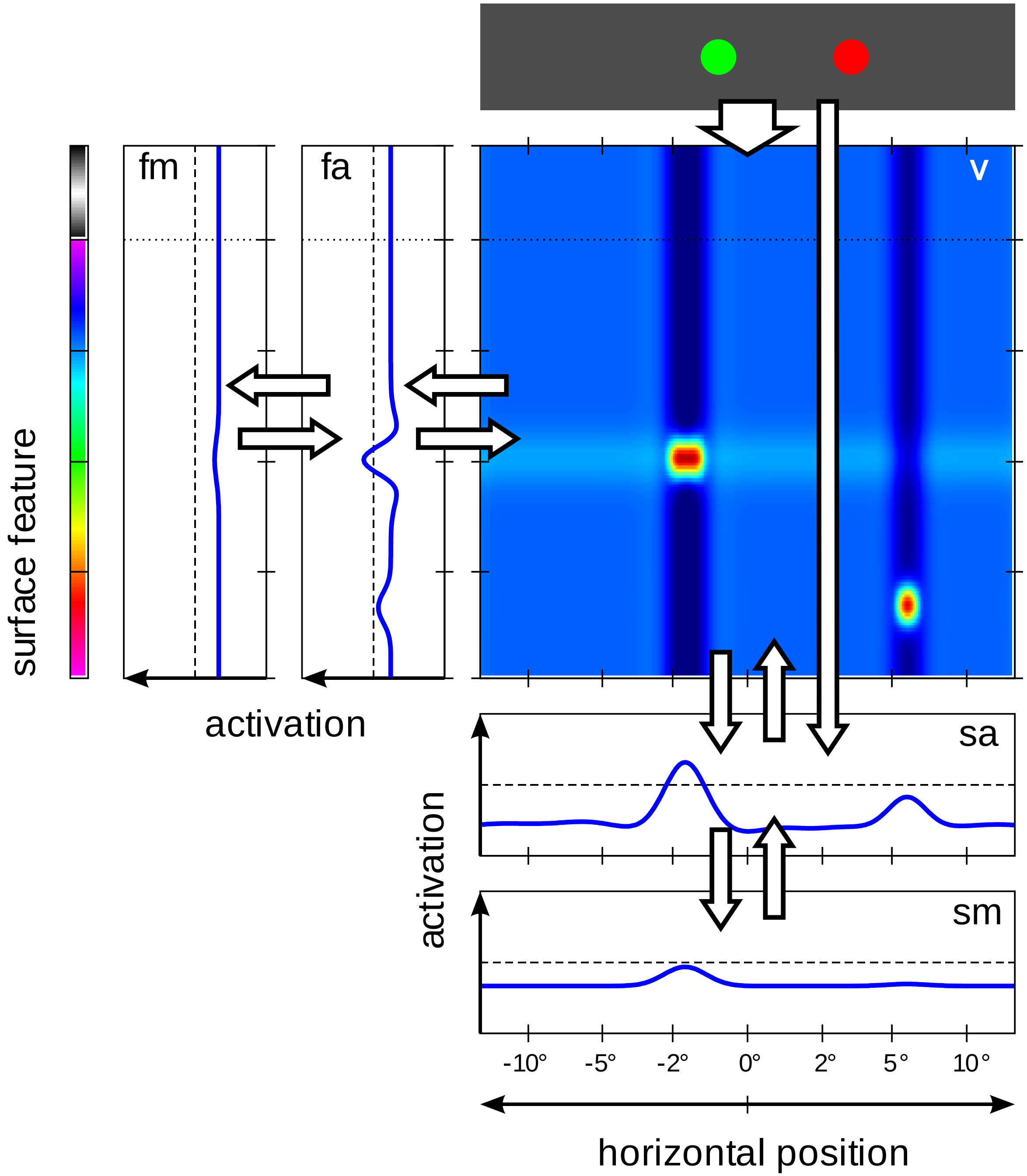


- once a single item is selected jointly in both 1D fields, ambiguity in feature conjunctions is resolved
- object features can then be processed in separate pathways
- sequential processing for multiple items

Case Study: VWM Biases Saccade Behavior



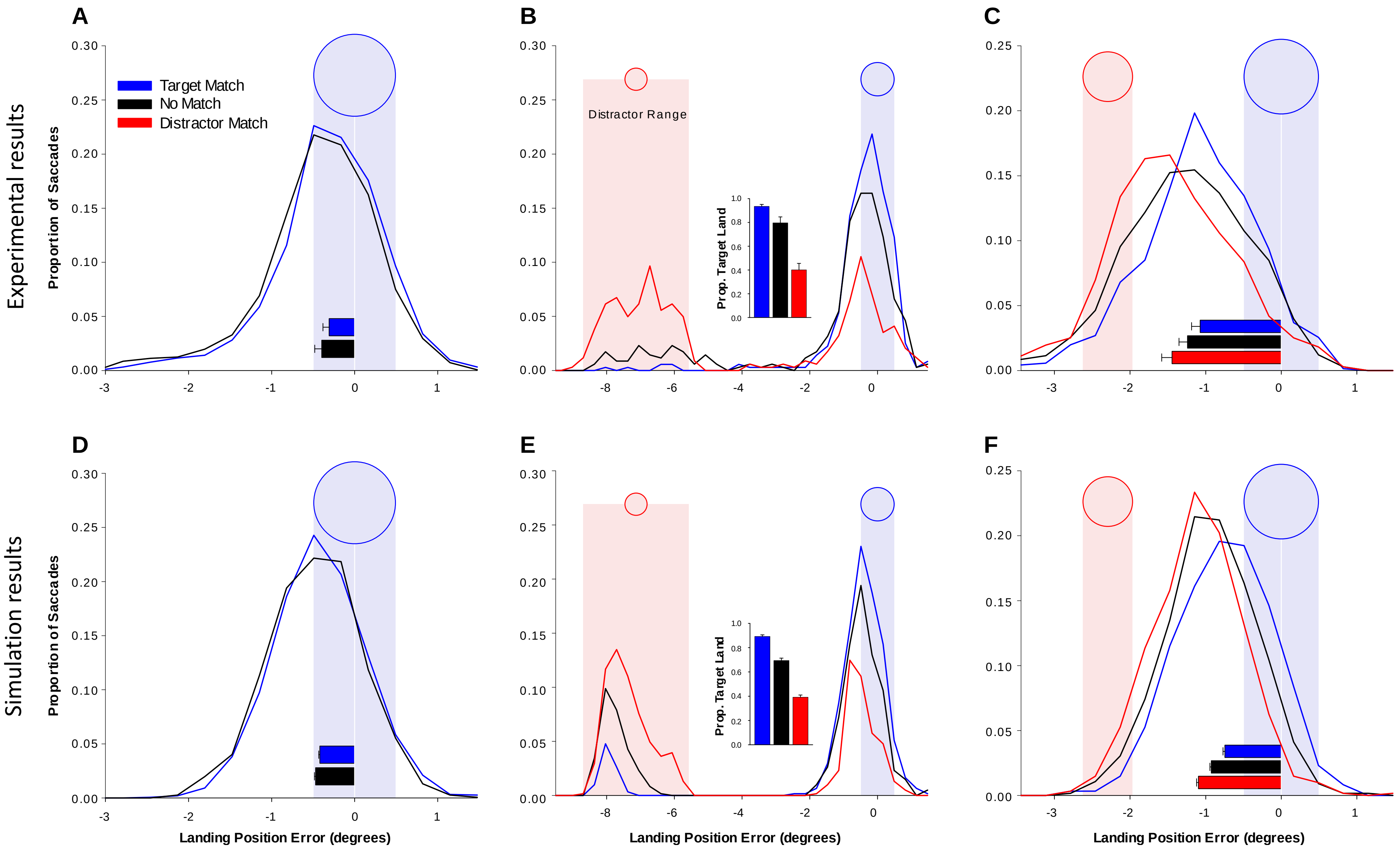
Case Study: VWM Biases Saccade Behavior



Case Study: VWM Biases Saccade Behavior

Video

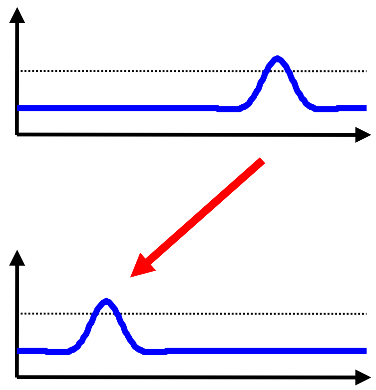
Case Study: VWM Biases Saccade Behavior



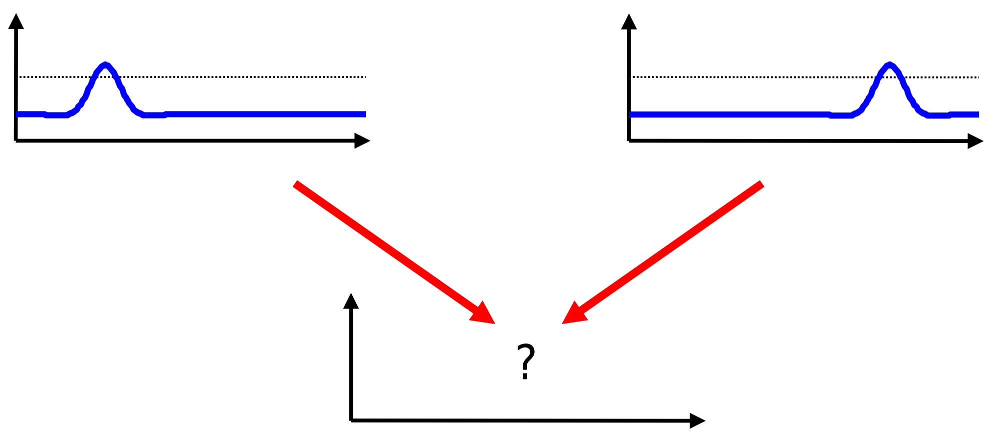
[Schneegans, Spencer, Schöner, Hwang, Hollingworth, 2014]

Operations in higher-dimensional fields

- projections between fields can implement simple mappings if they meet certain conditions (e.g. continuity)

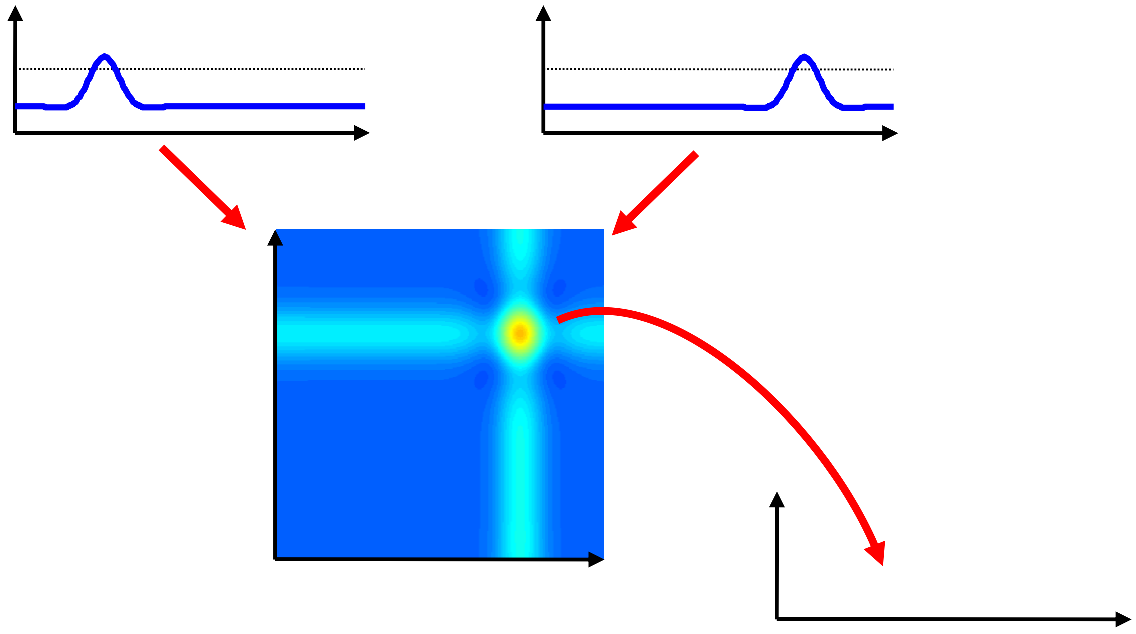


- what about operations that combine two different inputs?

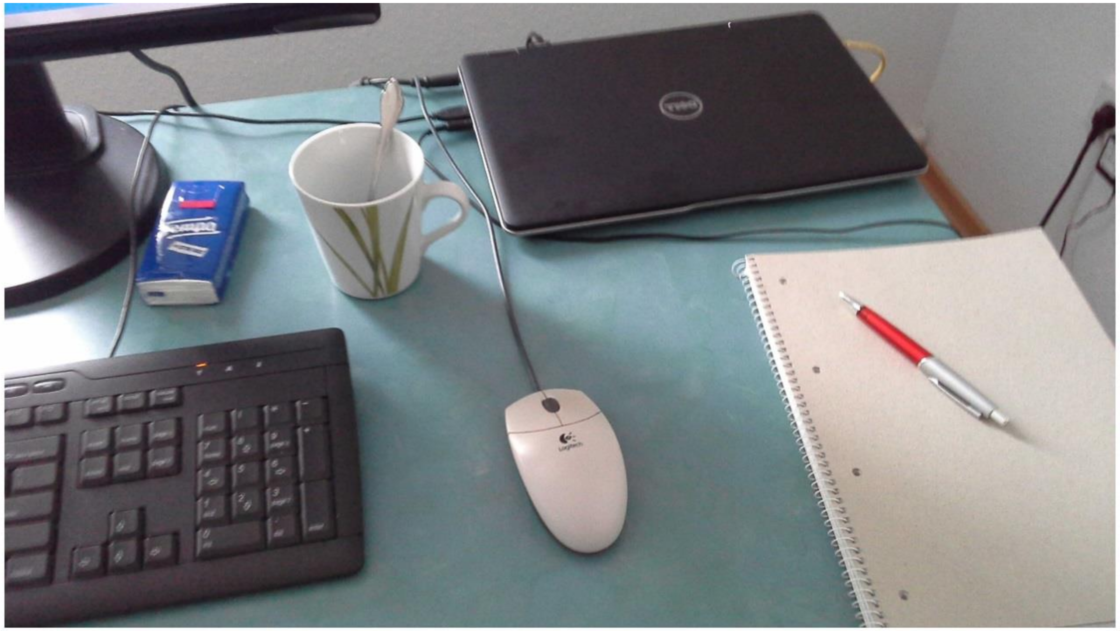


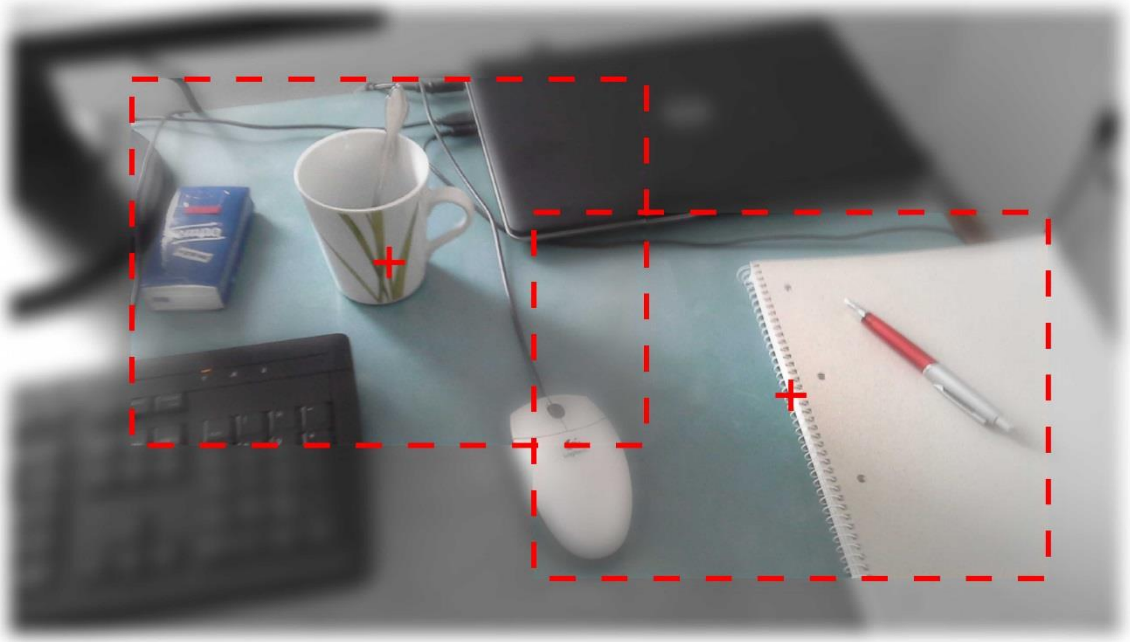
Operations in higher-dimensional fields

- combining/expanding representations into a single high-dimensional field allows arbitrary mappings to an output field (as long as mapping is continuous)

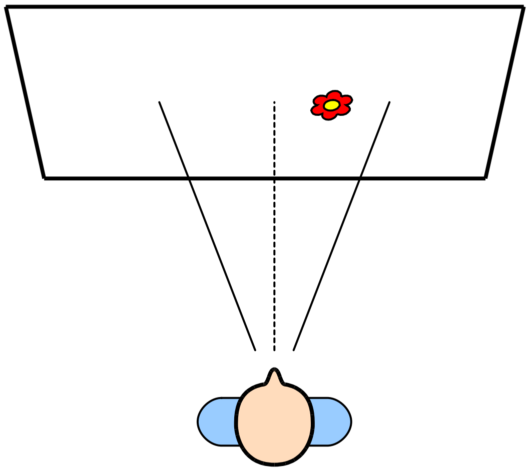


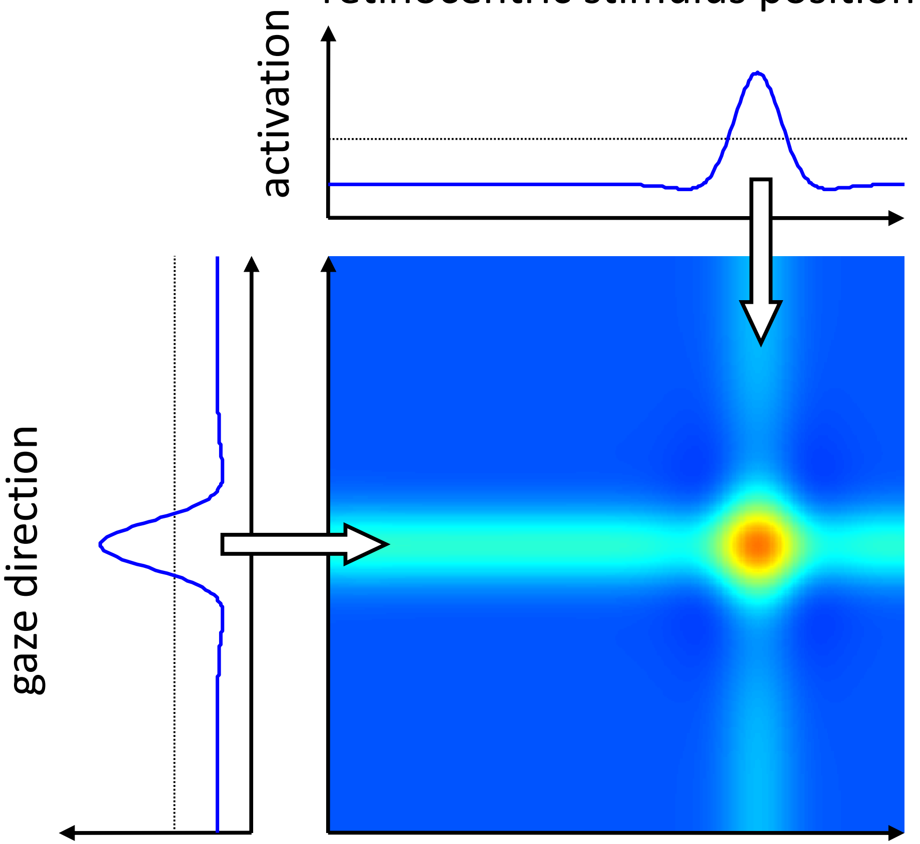
Retinocentric vs. allocentric positions





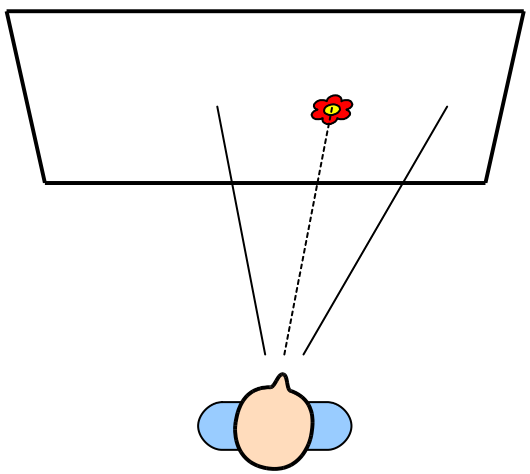
Spatial transformations

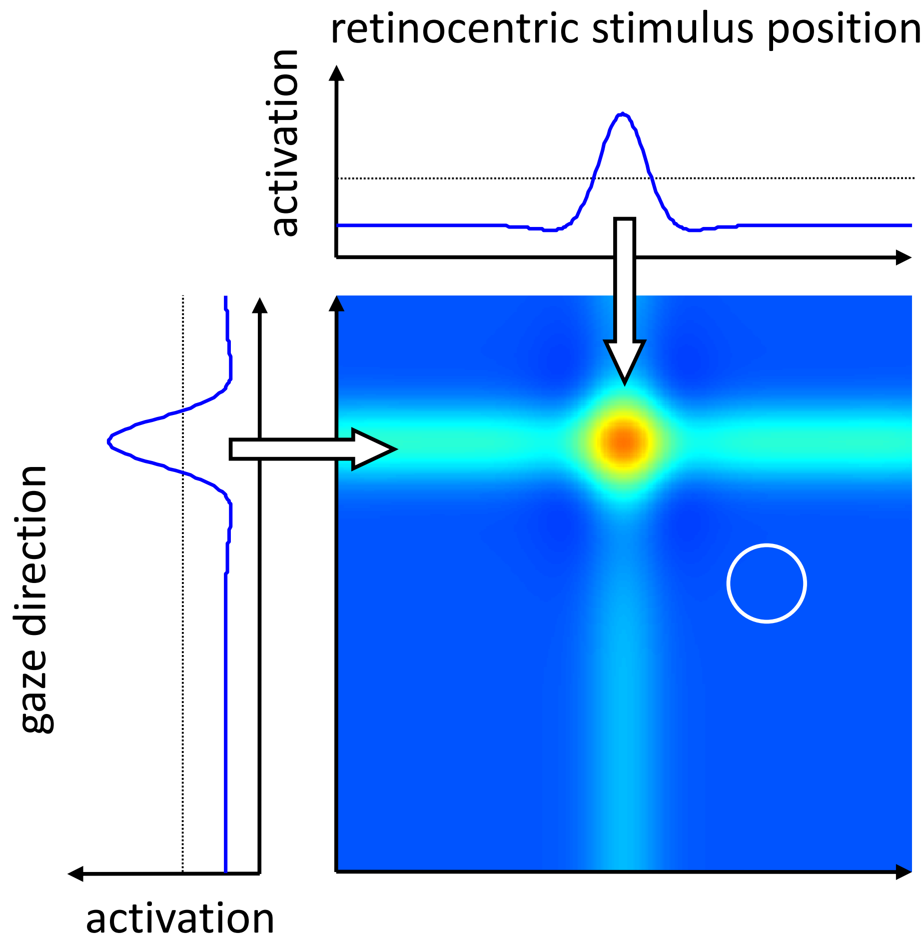




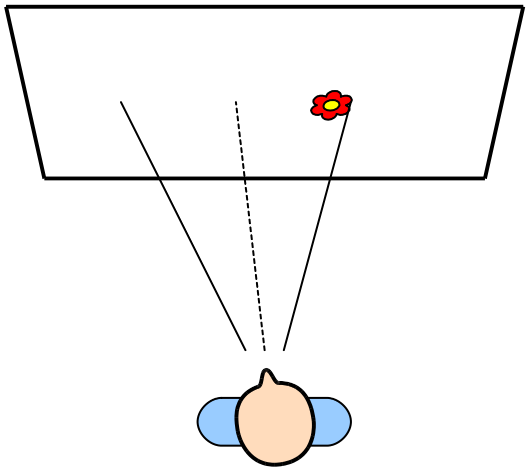
- for transformation of 1D location information: 2D field over retinal space and gaze direction

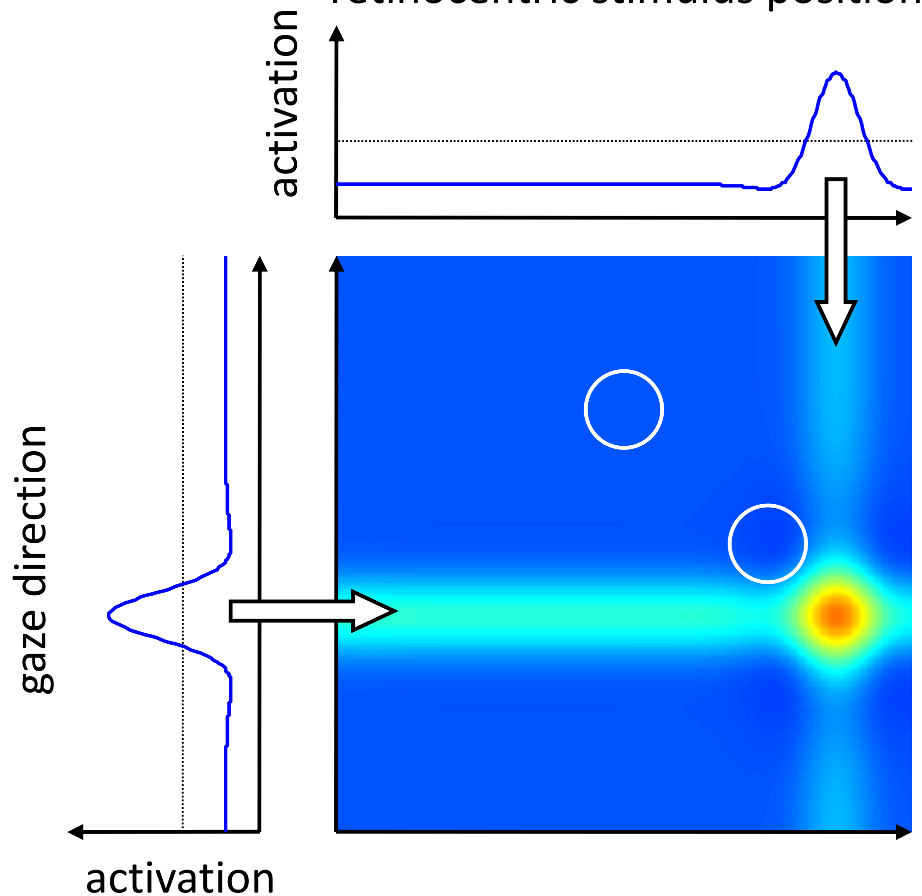
Spatial transformations





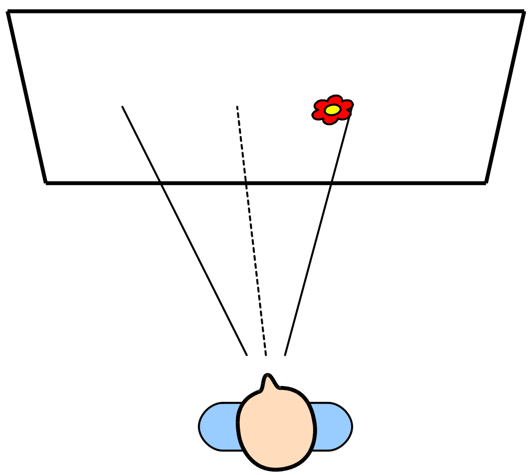
Spatial transformations

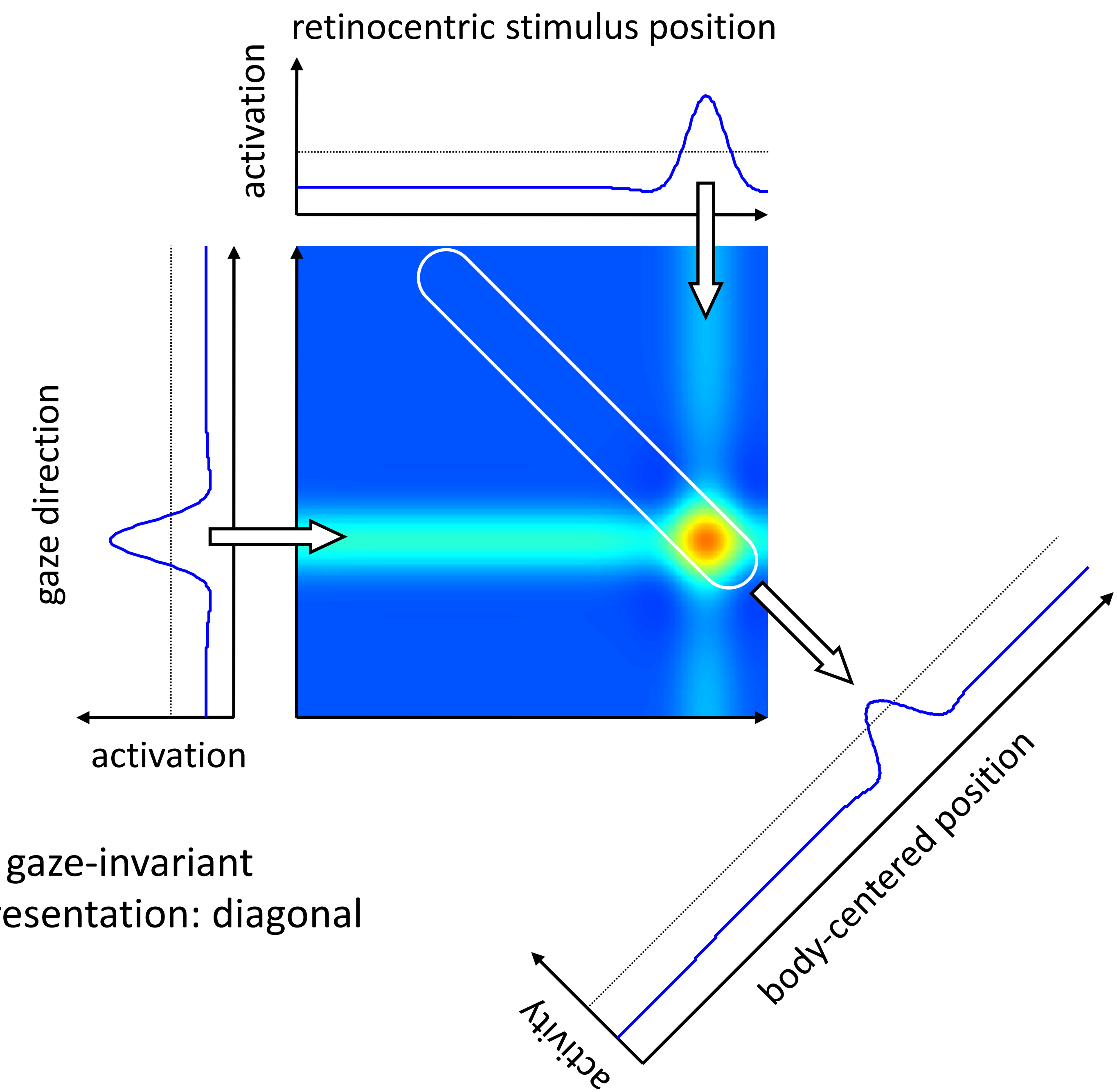




- in angular coordinates for pure rotations: retinocentric stimulus position shifts by inverse of gaze change
- → points corresponding to the same location lie on a diagonal in the combined representation

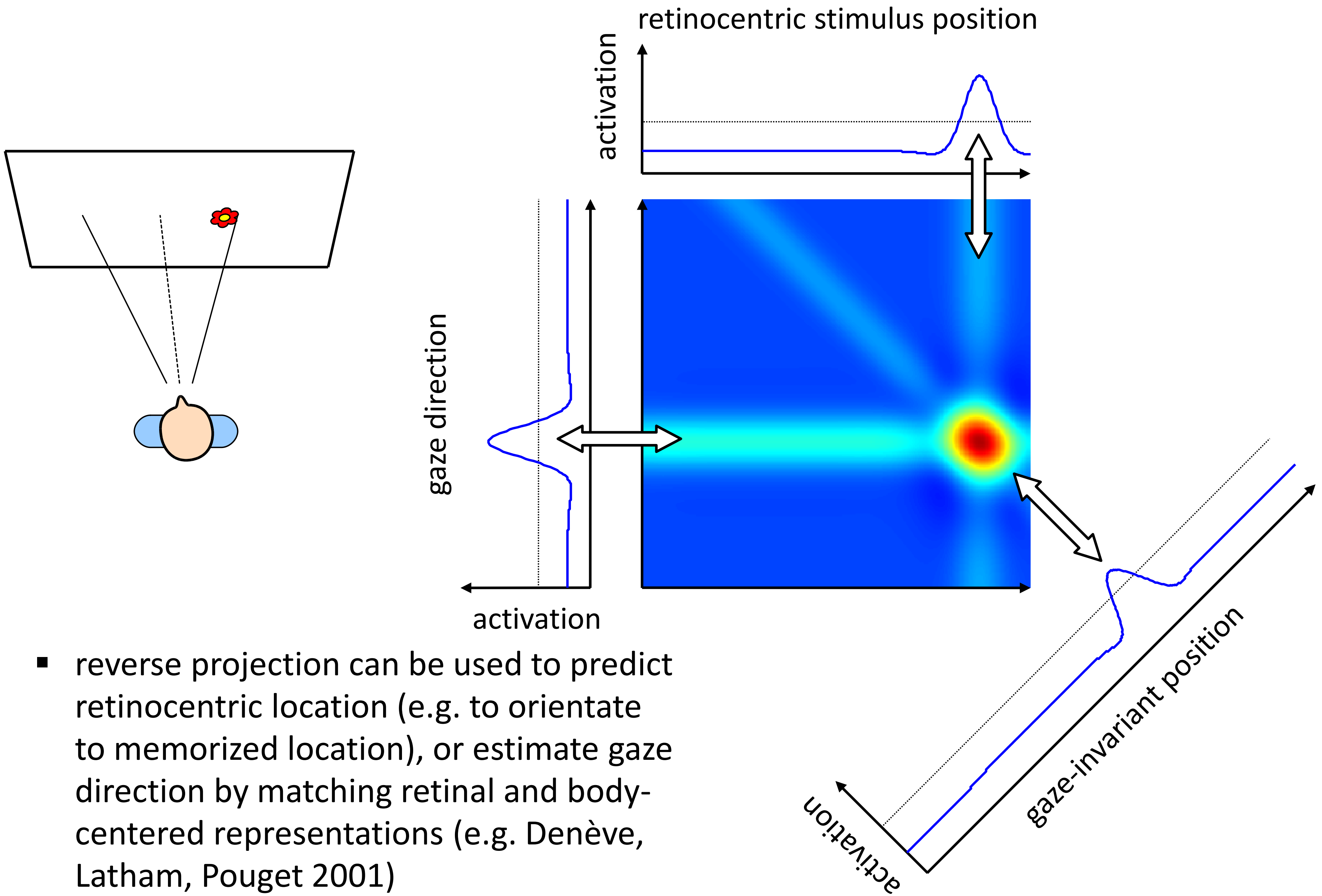
Spatial transformations





- can be mapped onto gaze-invariant (body-centered) representation: diagonal read-out

Spatial transformations



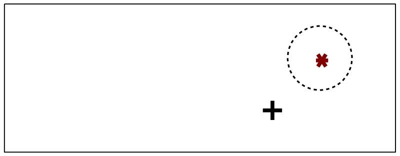
- reverse projection can be used to predict retinocentric location (e.g. to orientate to memorized location), or estimate gaze direction by matching retinal and body-centered representations (e.g. Denève, Latham, Pouget 2001)

Case Study: Saccadic Remapping Model

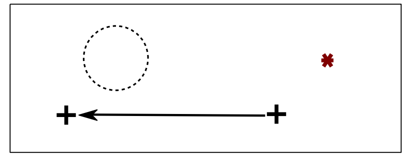
Video

Case Study: Saccadic Remapping Model

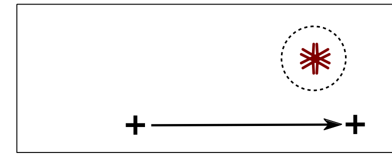
Condition



Stimulus in RF turned on and off

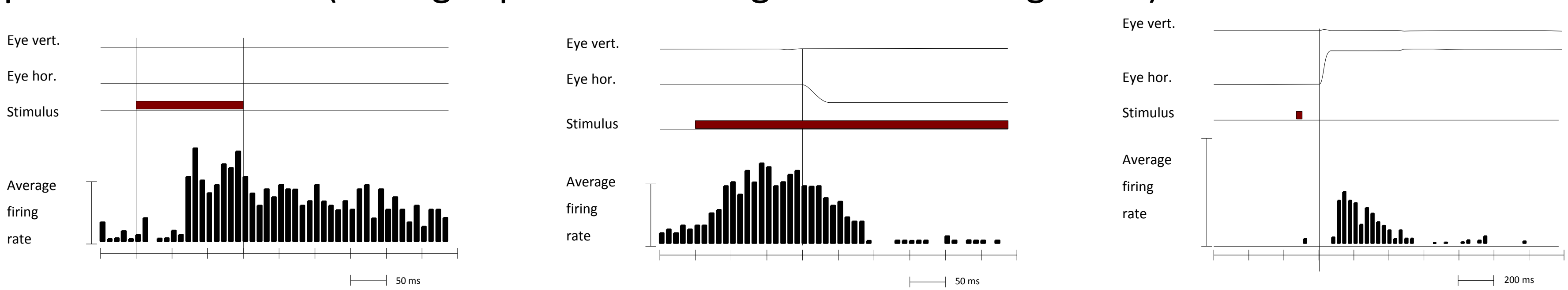


Saccade moves stimulus out of RF

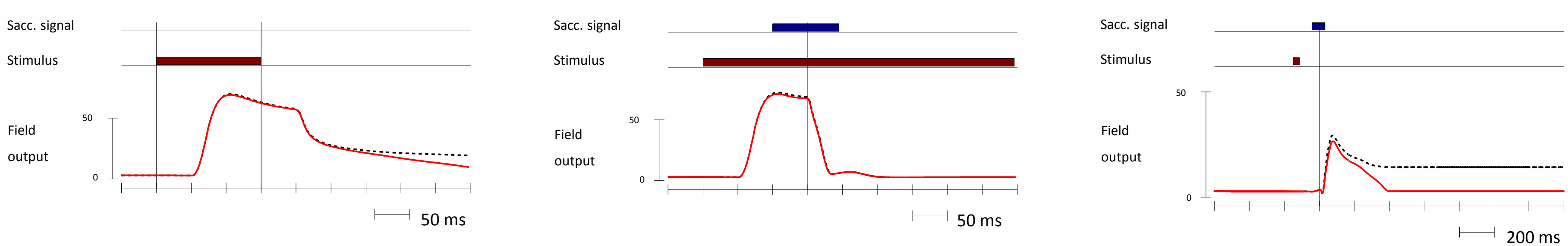


Saccade brings former stimulus position into RF

Experimental results (average spike rate of single-cell recording in LIP)



Simulation results (field output at one retinocentric position)



[Schneegans, Schöner 2012; experimental results by Duhamel et al. 1992]

Conclusions

- higher-dimensional fields can represent multiple feature dimensions in a combined fashion
- more costly than low-dimensional fields, but needed to represent feature conjunctions rather than separate feature values
- can provide associations between feature dimensions, e.g. for visual search
- can implement complex mappings between feature dimensions, e.g. for spatial transformations

Resources

cosivina

- http://bitbucket.org/sschneegans/cosivina
- object-oriented toolbox for Matlab, allows easy composition and visualization of DNF models

cedar

- http://bitbucket.org/cedar
- C++ framework for DNF models and robotics, with graphical user interface for composing architectures