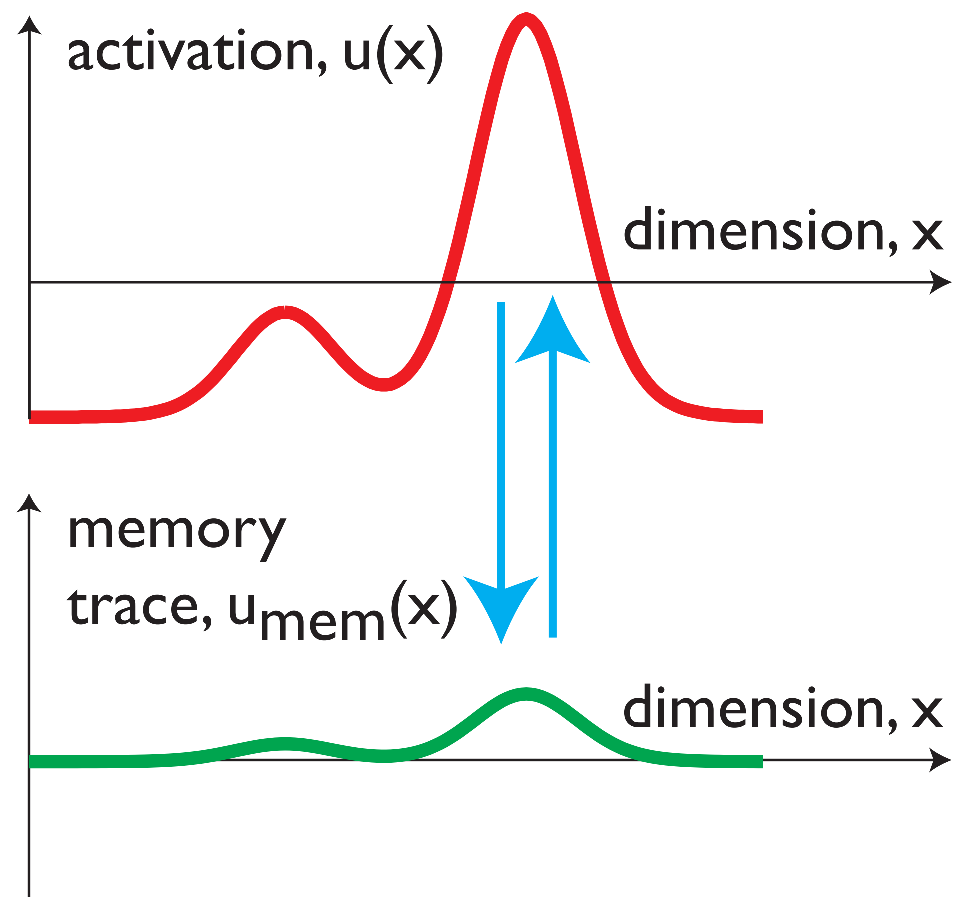
DFT and learning

Gregor Schöner

the memory trace

- building a facilitatory trace of patterns of activation
- (that can be inhibitory if they are build in an inhibitory field)



mathematics of the memory trace

$$\begin{aligned}\tau \dot{u}(x,t) &= -u(x,t) + h + S(x,t) + u_{\text{mem}}(x,t) \\ &+ \int dx' \; w(x-x') \; \sigma(u(x'))\end{aligned}$$

$$\begin{aligned}\tau_{\text{mem}} \; \dot{u}_{\text{mem}}(x,t) &= -u_{\text{mem}}(x,t) \\ &+ \int dx' \; w_{\text{mem}}(x-x')\sigma(u(x',t))\end{aligned}$$

- memory trace only evolves while activation is excited
- potentially different growth and decay rates

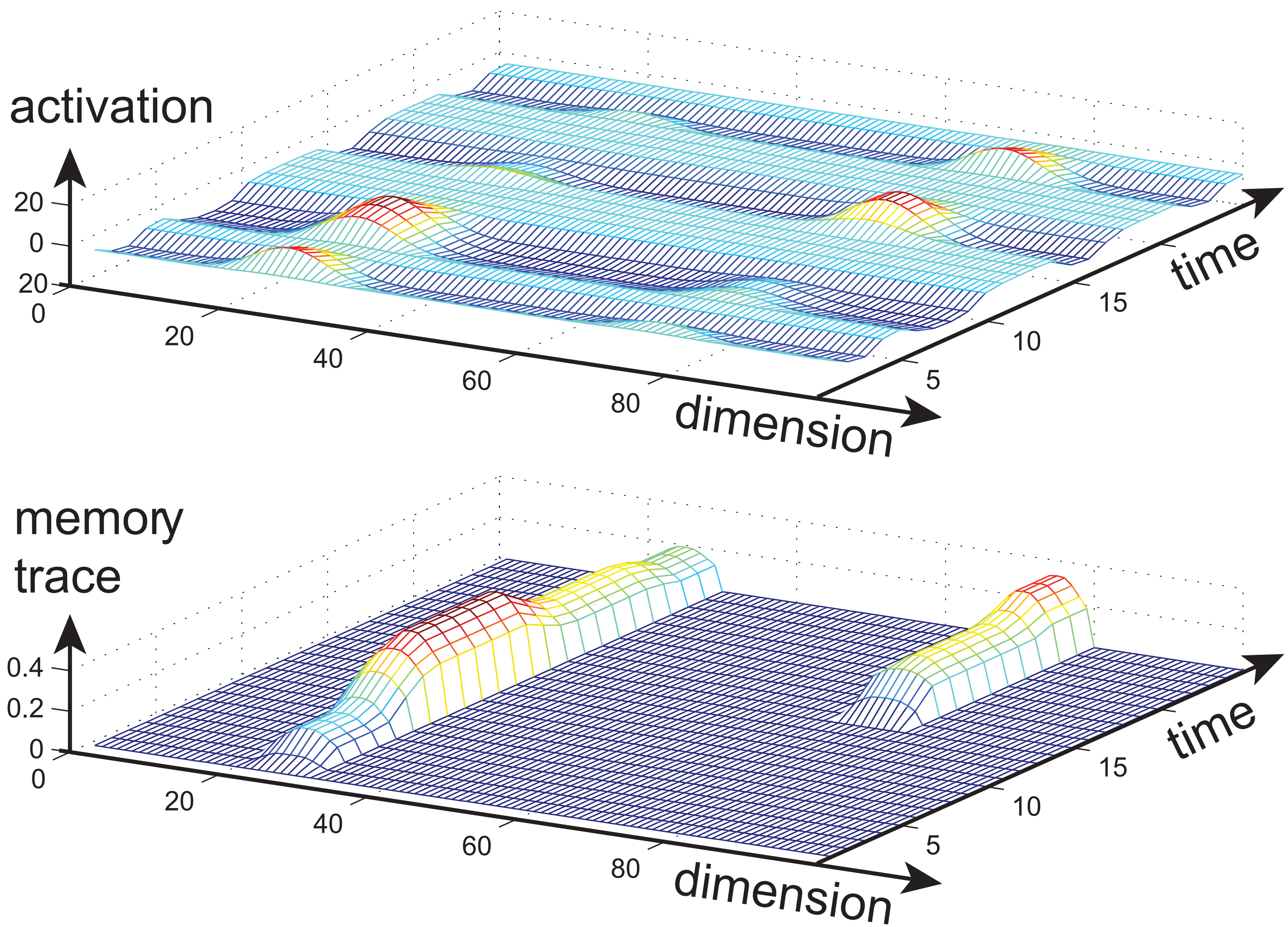
Dynamics of the memory trace

- different growth and decay rates

$$\tau_l \dot{P}(x,t) = \lambda_{build}\Big(-P(x,t) + f\big(u(x,t)\big)\Big)f\big(u(x,t)\big)$$
$$- \lambda_{decay} P(x,t)\Big(1 - f\big(u(x,t)\big)\Big).$$

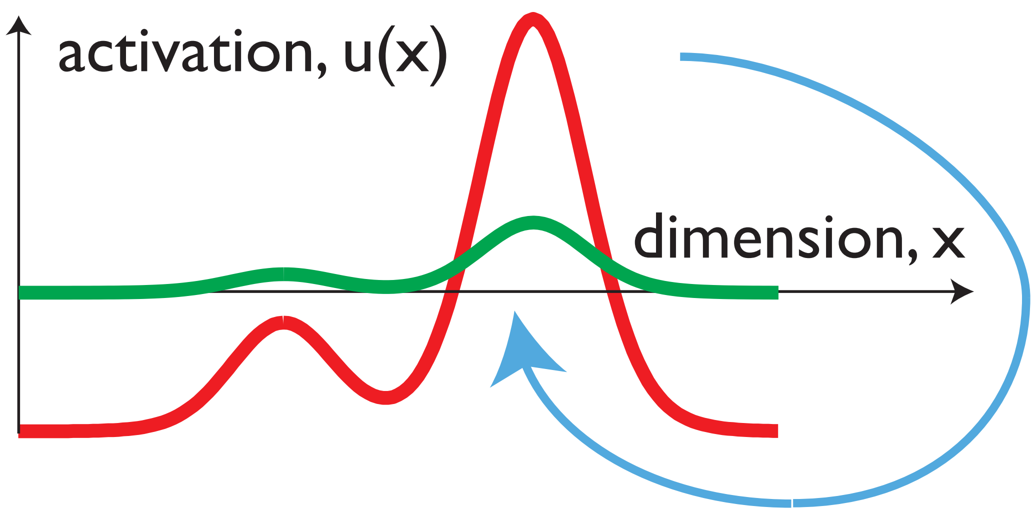
[Sandamirskaya, 2014]

memory trace reflects history of decisions formation



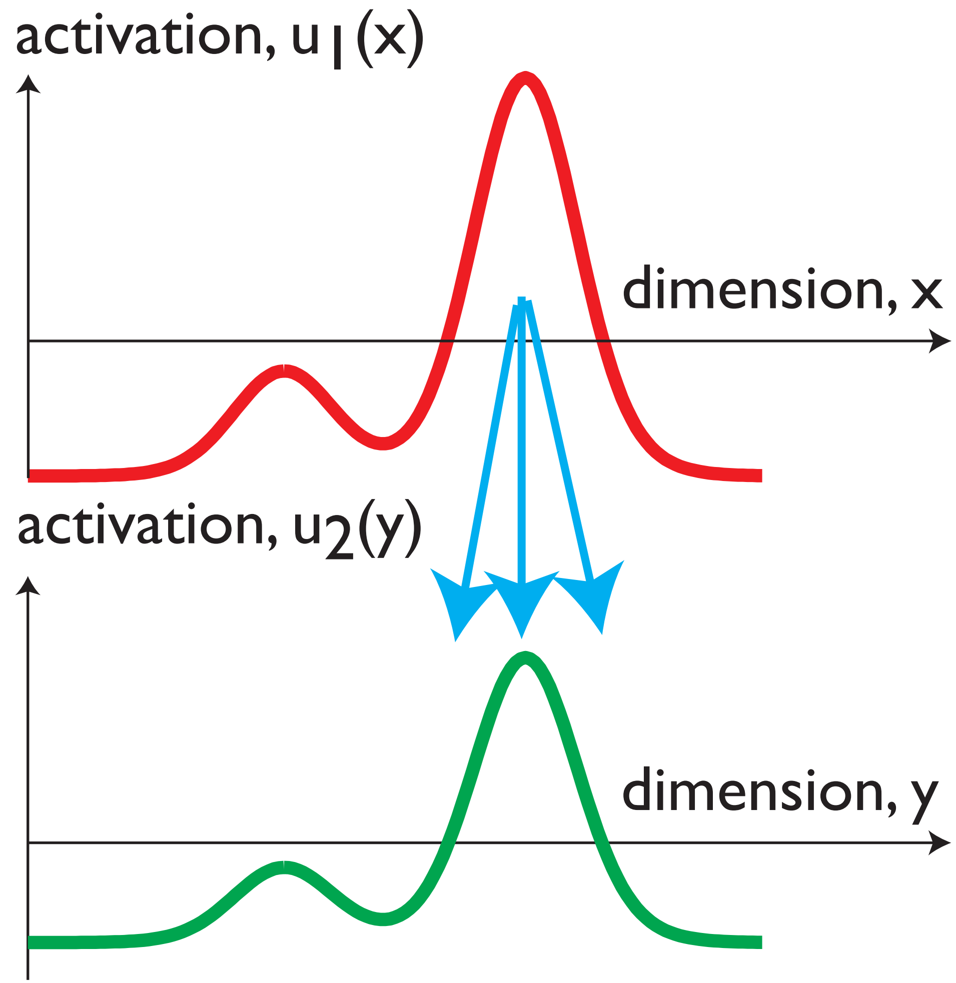
Memory trace as first-order Hebbian learning

- increase resting level at those field locations where and when supra-threshold activation is present
- ~the old "bias" unit in NN
- that does much more work here due to the boost-driven detection instability



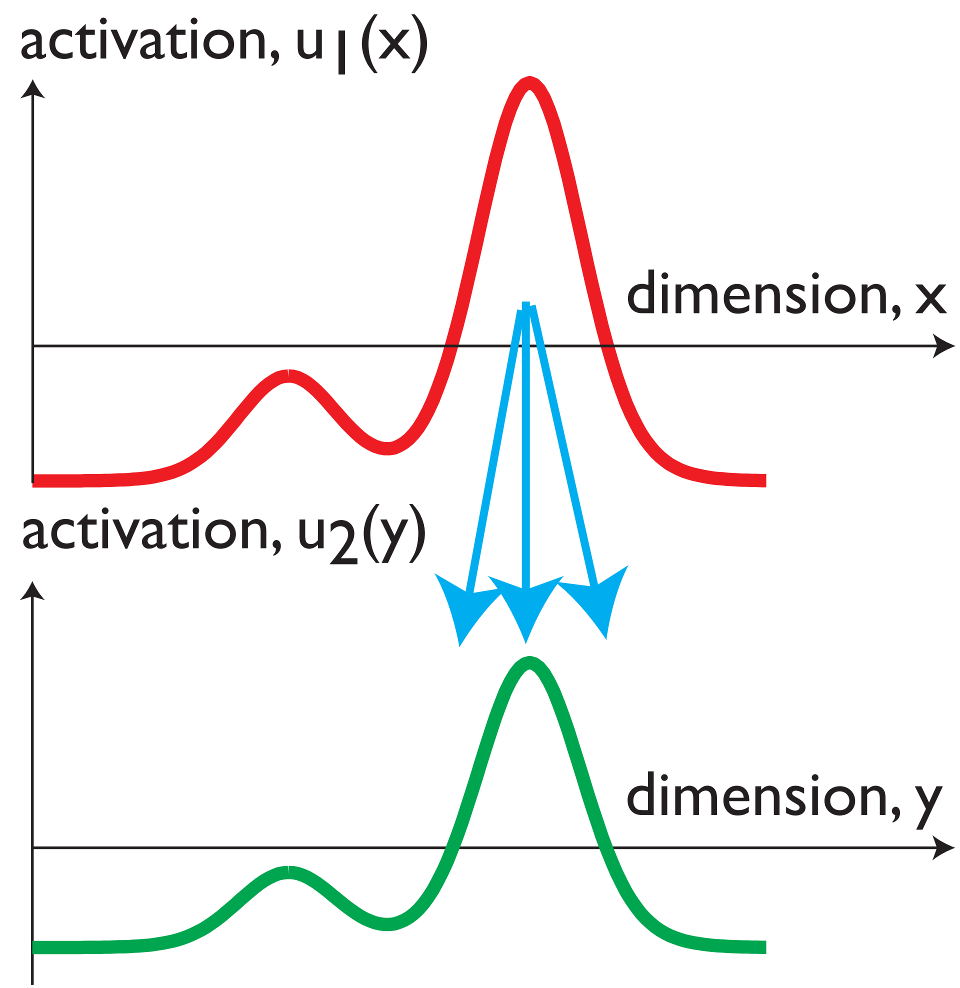
Regular second-order Hebbian learning

- projections among fields (or from sensory input to field) learns according to Hebb rule
  - strengthen input projection where supra-threshold activation in both fields are aligned



Regular second-order Hebbian learning

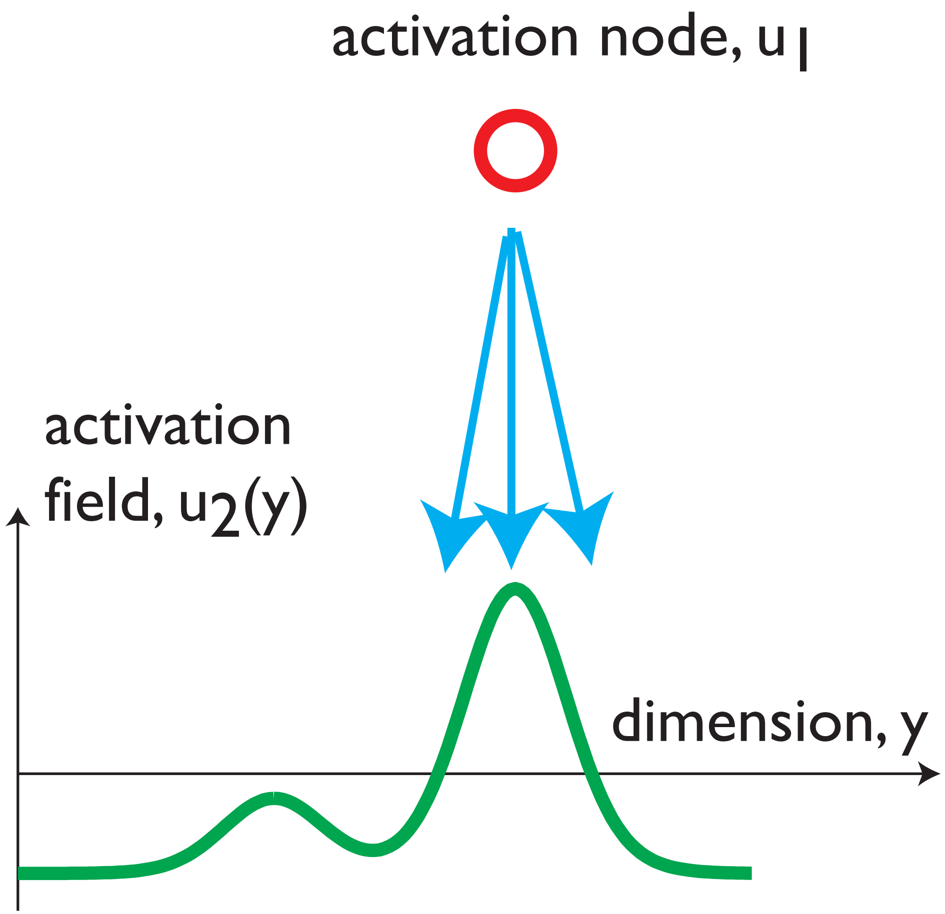
$$\tau \dot{W}(x, y, t) = \epsilon(t)\Big(- W(x, y, t) + f(u_1(x, t)) \times f(u_2(y, t))\Big)$$



[Sandamirskaya, 2014]

Regular second-order Hebbian learning

- used a lot in DFT for projections from zero-dimensional nodes to one-dimensional nodes
- or generally, from lower to higher dimensional field
- => concepts



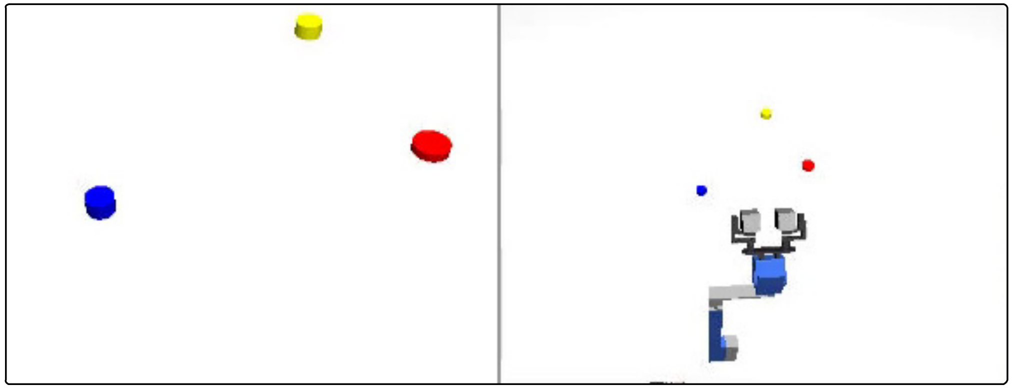
Autonomous learning

- Learning as change of neural dynamics (memory trace, Hebb) driven by ongoing activation patterns while system is "behaving"
- (rather than in a particular training regime in which parts of the architecture is "clamped" or in which error information is provided)

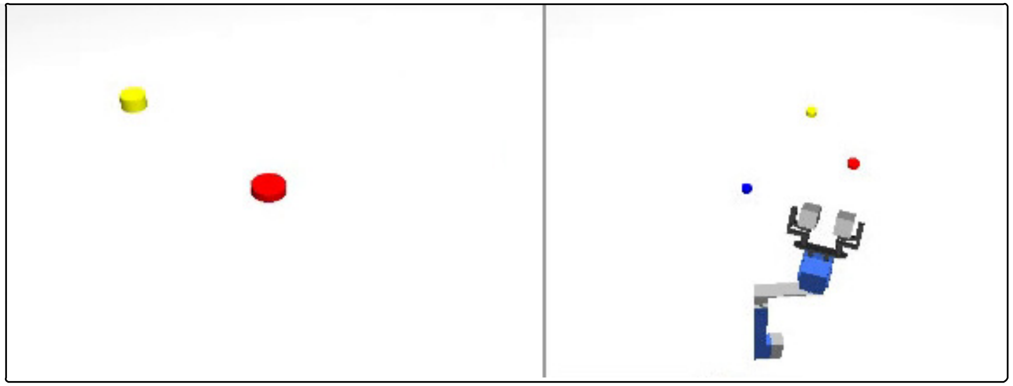
Example: learning to look

- have "retinal" coordinates of a visual target
- need motor command to move fovea onto the visual target

Snapshot 1: before looking

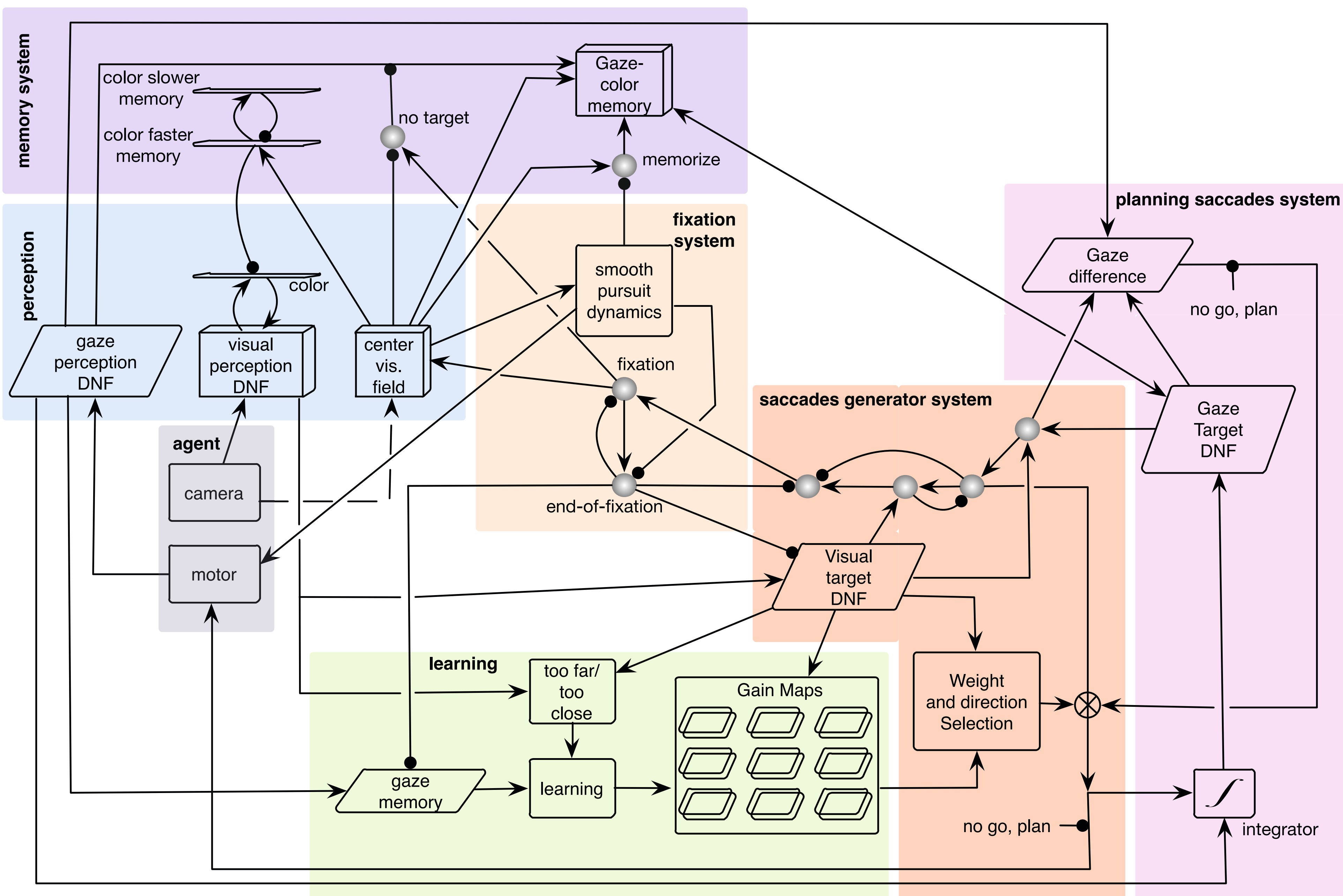


Snapshot 2: after gaze shift 1

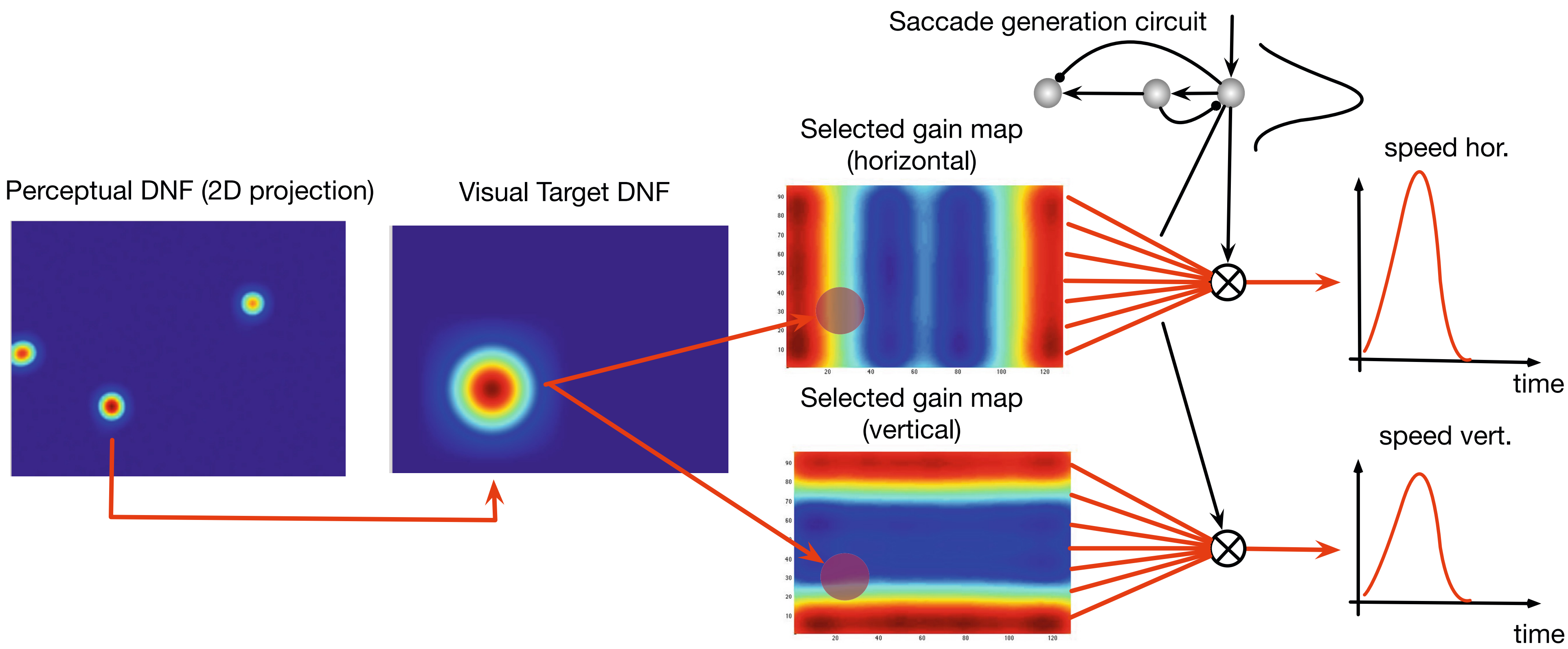


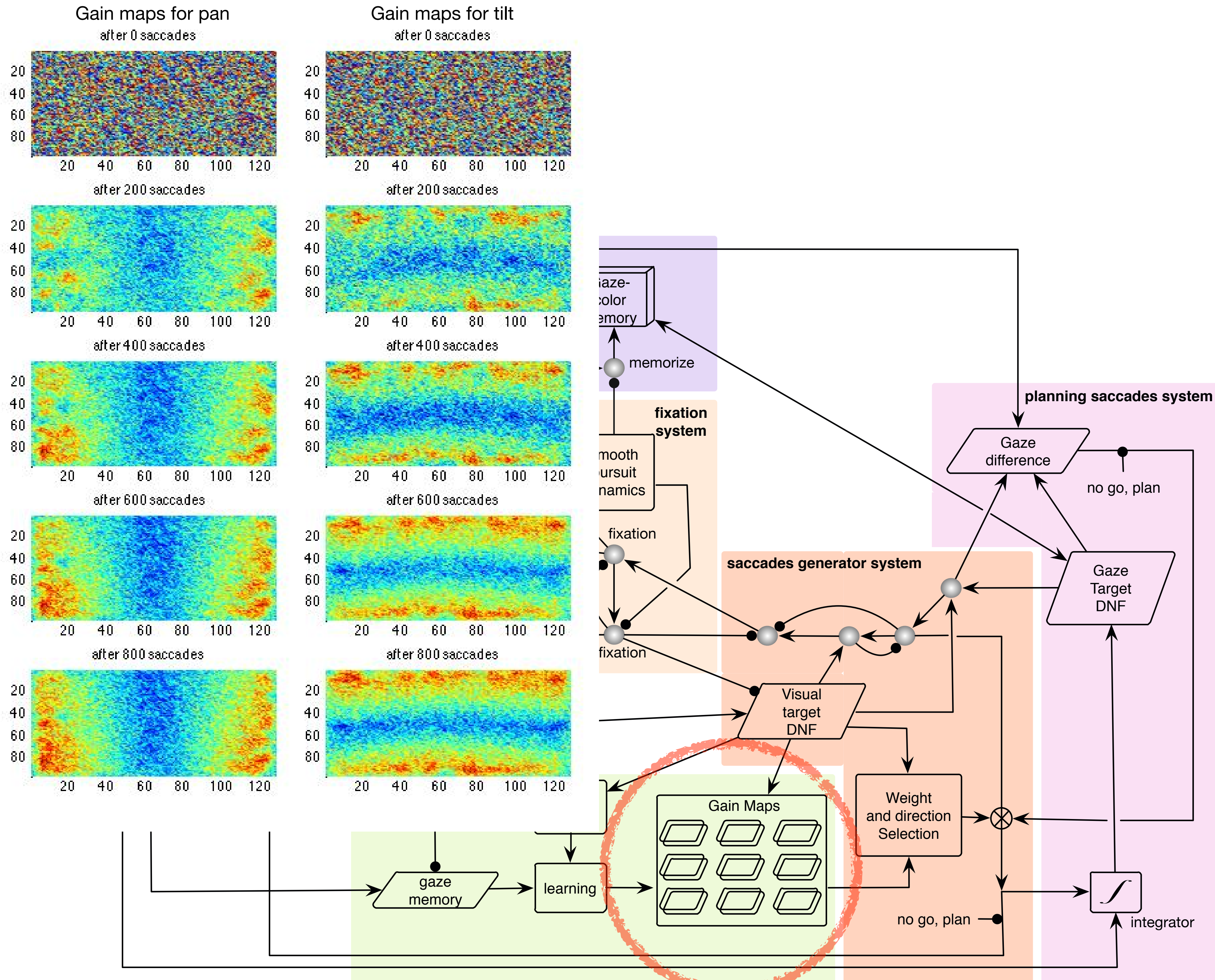
[Sandamirskaya, Storck: Artificial Neural Networks, Springer 2015]

process infrastructure to organize looking and learning



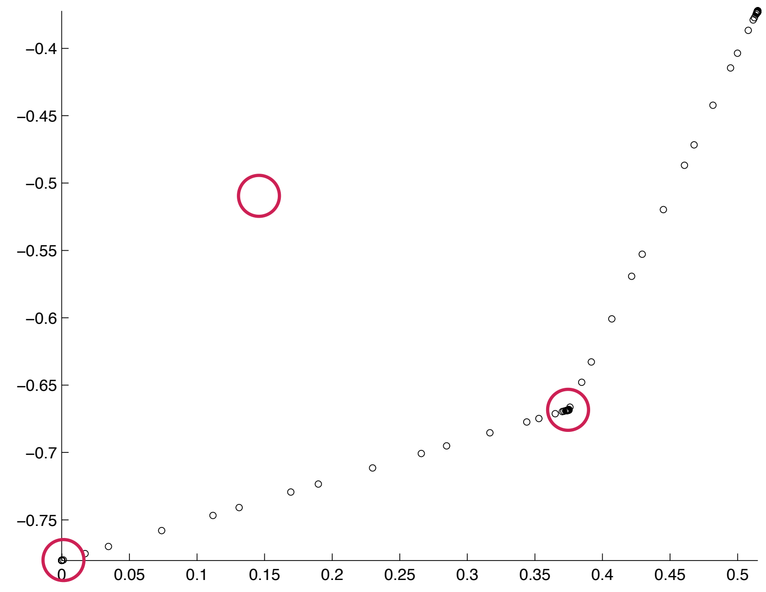
core element of learning: a (steerable) map from the "retina" to motor commands



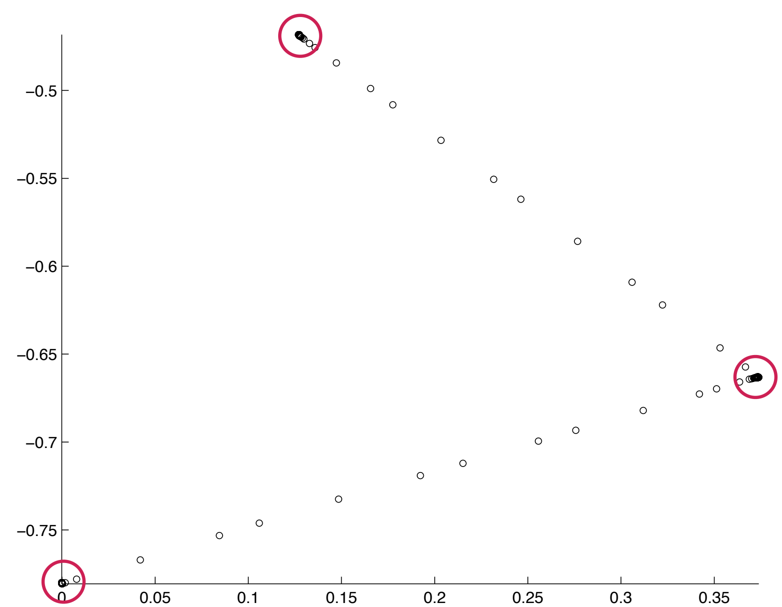


Gain maps for pan
Gain maps for tilt
after 0 saccades
after 200 saccades
after 400 saccades
after 600 saccades
after 800 saccades
20
40
60
80
100
120
planning saccades system
Gaze difference
no go, plan
Gaze Target DNF
fixation system
memorize
mooth
ursuit
namics
fixation
saccades generator system
Visual target DNF
Weight and direction Selection
Gain Maps
gaze memory
learning
no go, plan
integrator

- during learning a transition from gaze memory in retinal to gaze memory in body/scene coordinates



(a) Retina-memory saccades ('young model').



(b) Motor-memory saccades ('older model').

Autonomous learning

- … requires a lot of process structure
  - remembering the visual representation to bridge the temporal gap and compute error signals
  - remembering the motor command
  - autonomously organizing the update and storage of such information ..