

Wulfram Gerstner

EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL

from brain-style computing to neuromorphic computing

Objectives for today:

- application of three-factor rules
- local learning rules for hardware
- Spiking Neural Networks (SNN)
- neuromorphic chips
- reducing energy consumption

Previous slide. We continue our discussion from last week

Background reading:

Fremaux et al., 2013, PLOS Comput. Biol.
[doi:10.1371/journal.pcbi.1003024](https://doi.org/10.1371/journal.pcbi.1003024)

IBM research lab

[Bert Offrein et al., 2020, Prospects for photonic implementations of neuromorphic devices and systems, IEEE Xplore, <https://ieeexplore.ieee.org/abstract/document/9371915>](#)

LOIHI Chip (intel)

<https://en.wikichip.org/wiki/intel/loihi>

<https://download.intel.com/newsroom/2021/new-technologies/neuromorphic-computing-loihi-2-brief.pdf>

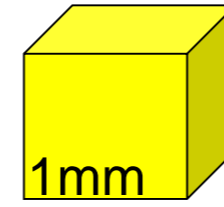
Implementation of Navigation task with Memristors

<https://doi.org/10.1038/s41467-023-37097-5>

Review: Neurons and Synapses form a big network



brain



1mm

1mm

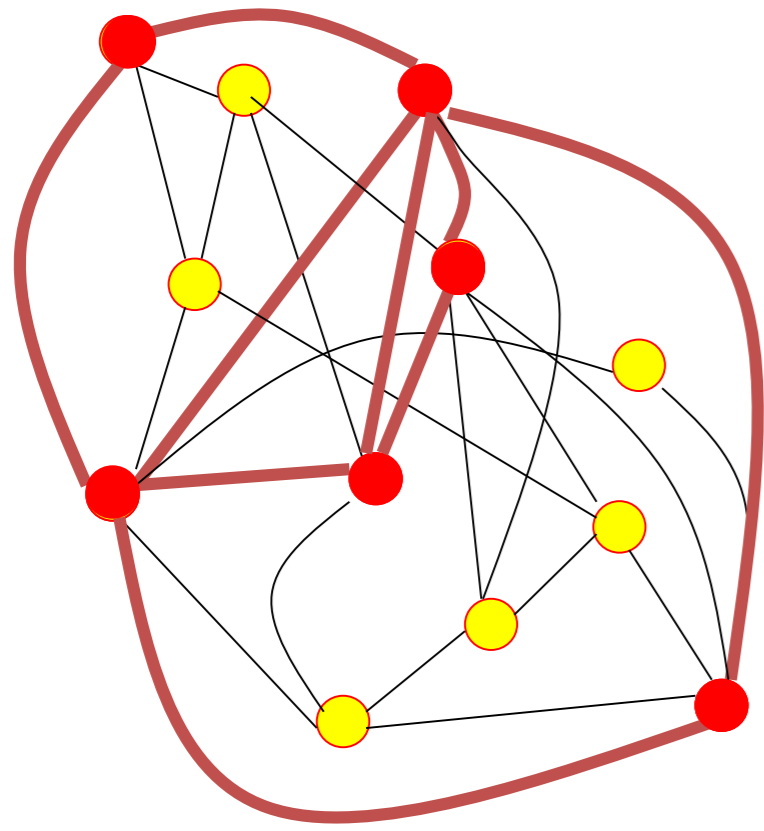
10 000 neurons

3 km of wire

10 billions neurons

10 000 connexions/neurons

non-von-Neumann
computing & hardware
'brain-style computing'



Distributed Architecture

memory in the connections

**No separation of
processing and memory**

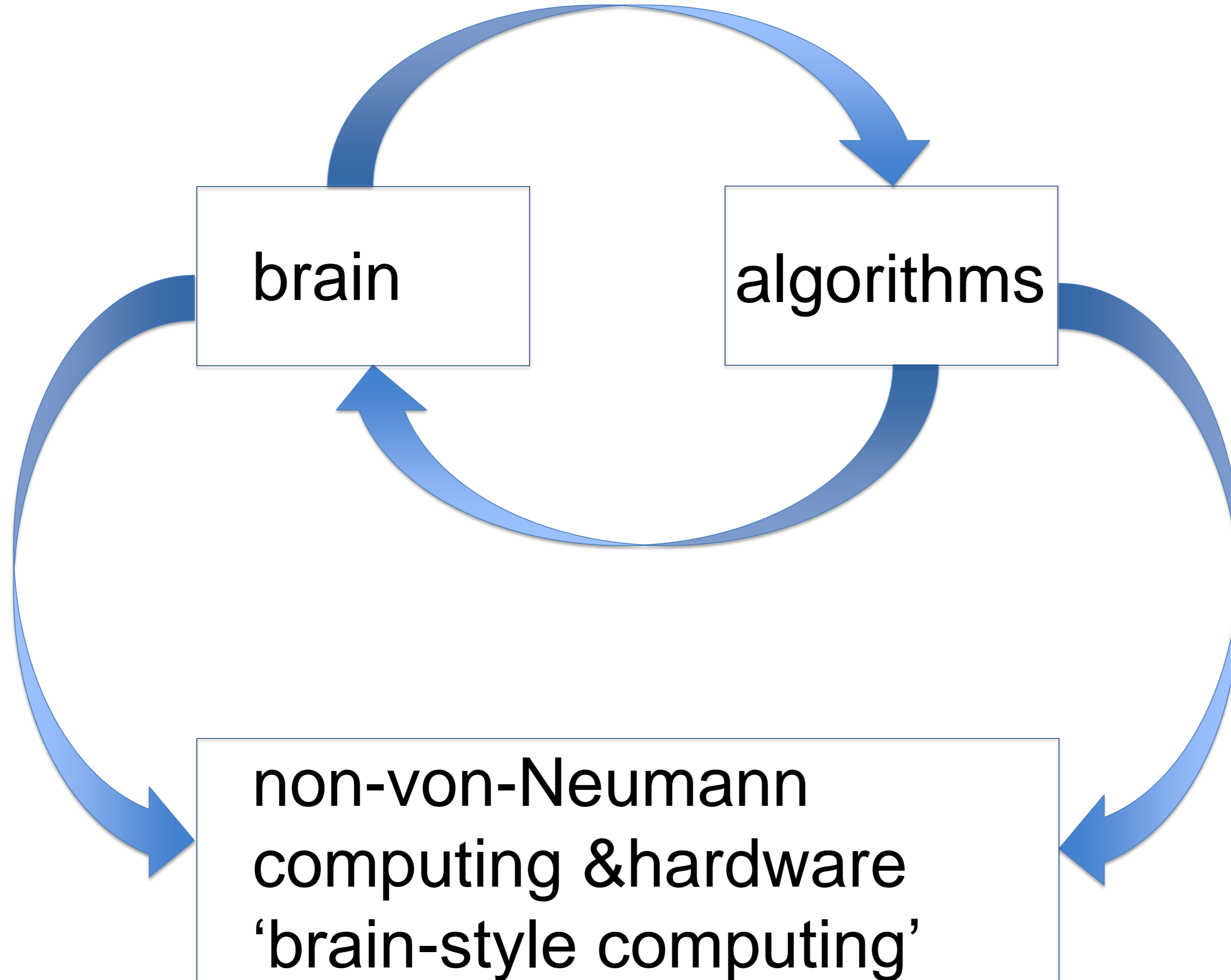
Previous slide. Review from previous lectures.

In the first lecture it was mentioned that the brain is radically different from the classical von-Neumann architecture that lead to our standard compute devices.

Particularly important differences are that the brain-style computing architecture is completely distributed, without centralized clock, no centralized controller and no separation of computing and memory.

We take in the following the learning rules of RL as a starting point of what this means and mention at the end novel hardware.

Review: Learning Rules of Reinforcement Learning



Previous slide. Review from previous lectures.

RL has two roots: optimization and Markov Decision Problems and Brain sciences.

We take in the following the learning rules of RL to see how they lead to alternative computing paradigms.

Review: Advantage Actor-Critic with Eligibility traces

Actor–Critic with Eligibility Traces (continuing), for estimating $\pi_{\theta} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \theta)$

Input: a differentiable state-value function parameterization $\hat{v}(s, \mathbf{w})$

Algorithm parameters: $\lambda^{\mathbf{w}} \in [0, 1]$, $\lambda^{\theta} \in [0, 1]$, $\alpha^{\mathbf{w}} > 0$, $\alpha^{\theta} > 0$

Initialize state-value weights $\mathbf{w} \in \mathbb{R}^d$ and policy parameter $\theta \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Initialize $S \in \mathcal{S}$ (e.g., to s_0)

$\mathbf{z}^{\mathbf{w}} \leftarrow \mathbf{0}$ (d -component eligibility trace vector)

$\mathbf{z}^{\theta} \leftarrow \mathbf{0}$ (d' -component eligibility trace vector)

Loop forever (for each time step):

$A \sim \pi(\cdot|S, \theta)$

Take action A , observe S', r

$\delta \leftarrow r + \gamma \hat{v}(S', \mathbf{w}) - \hat{v}(S, \mathbf{w})$

$\mathbf{z}^{\mathbf{w}} \leftarrow \lambda^{\mathbf{w}} \mathbf{z}^{\mathbf{w}} + \nabla \hat{v}(S, \mathbf{w})$

$\mathbf{z}^{\theta} \leftarrow \lambda^{\theta} \mathbf{z}^{\theta} + \nabla \ln \pi(A|S, \theta)$

$\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \mathbf{z}^{\mathbf{w}}$

$\theta \leftarrow \theta + \alpha^{\theta} \delta \mathbf{z}^{\theta}$

$S \leftarrow S'$

The algo for the update is the 'learning rule'.

*Adapted from
Sutton and Barto*

Previous slide. Review from previous lectures.

Red box:

Parameters in the advantage actor critic change proportional to

- The TD error δ
- Eligibility trace.

In turn, eligibility traces change proportional to

- The derivative of the value function for the critic
- The derivative of the log policy for the actor
- A decay term

In the example today eligibility traces are important.

Review: Learning Rules of Reinforcement Learning

Assume the transition to state x^{t+1} with a reward of r^{t+1} after taking action a^t at state x^t . The learning rule for the Advantage Actor-Critic with Eligibility traces is

**‘learning rule’
of Advantage
Actor-Critic
with eligibility trace**

$$\begin{aligned}\delta &\leftarrow r^{t+1} + \gamma \hat{v}_w(x^{t+1}) - \hat{v}_w(x^t) \\ z^w &\leftarrow \lambda^w z^w + \nabla_w \hat{v}_w(x^t) \\ z^\theta &\leftarrow \lambda^\theta z^\theta + \nabla_\theta \pi_\theta(a^t | x^t) \\ w &\leftarrow w + \alpha^w z^w \delta \\ \theta &\leftarrow \theta + \alpha^\theta z^\theta \delta\end{aligned}\tag{1}$$

- Learning rules of other ONLINE RL policy gradient models are special cases of (1).
- We take (1) as a starting point to discuss the relation with the brain and with hardware

Can such a **learning rule** be implemented in the brain?

Can such a **learning rule** be implemented in hardware?

Previous slide. Review from previous lecture.

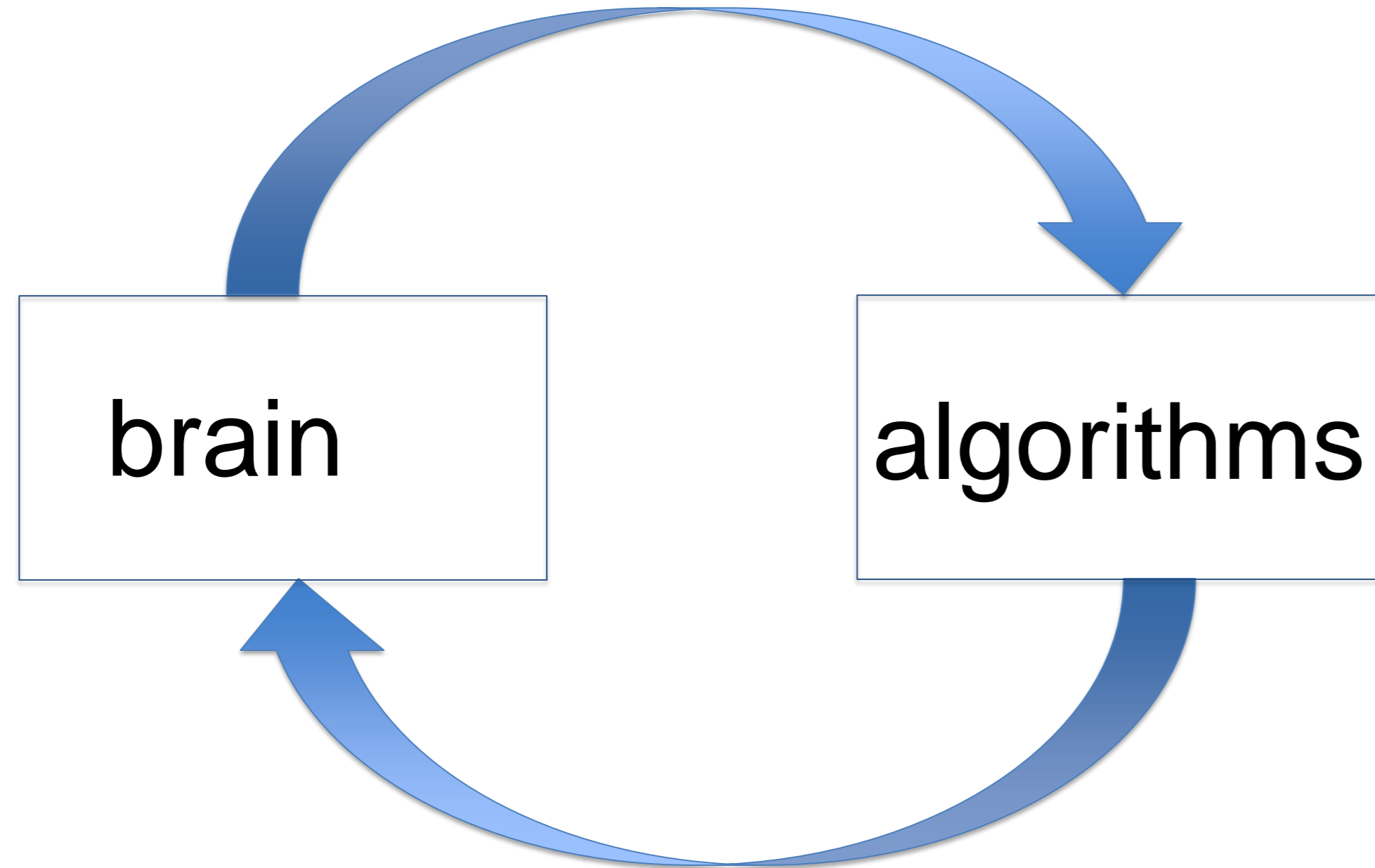
In the following we take the Advantage Actor Critic as our Reference Model.

As we have seen earlier, other Algorithms in the Family of Policy Gradients can be identified as special cases.

Last week we have seen how such a learning rule (update algorithm) be implemented in the brain.

In this lecture we ask: how could an implementation of the actor-critic look like in the brain? And in hardware?

Review: Learning Rules of Reinforcement Learning



Update of all eligibility traces

$$z_{lk} \leftarrow z_{lk} \lambda_z$$

$$z_{lk} \leftarrow z_{lk} + \frac{d}{dw_{lk}} \ln[\pi(a|s, w_{lk})]$$

Change of **all** weights

$$\Delta w_{lk} = \eta r_t z_{lk}$$

$$\Delta w_{lk} = \eta \delta_t z_{lk}$$

The learning rule of the (advantage) actor-critic or REINFORCE with eligibility traces are both compatible with three-factor rules

Updates proportional to the reward r or TD error δ_t

Previous slides.

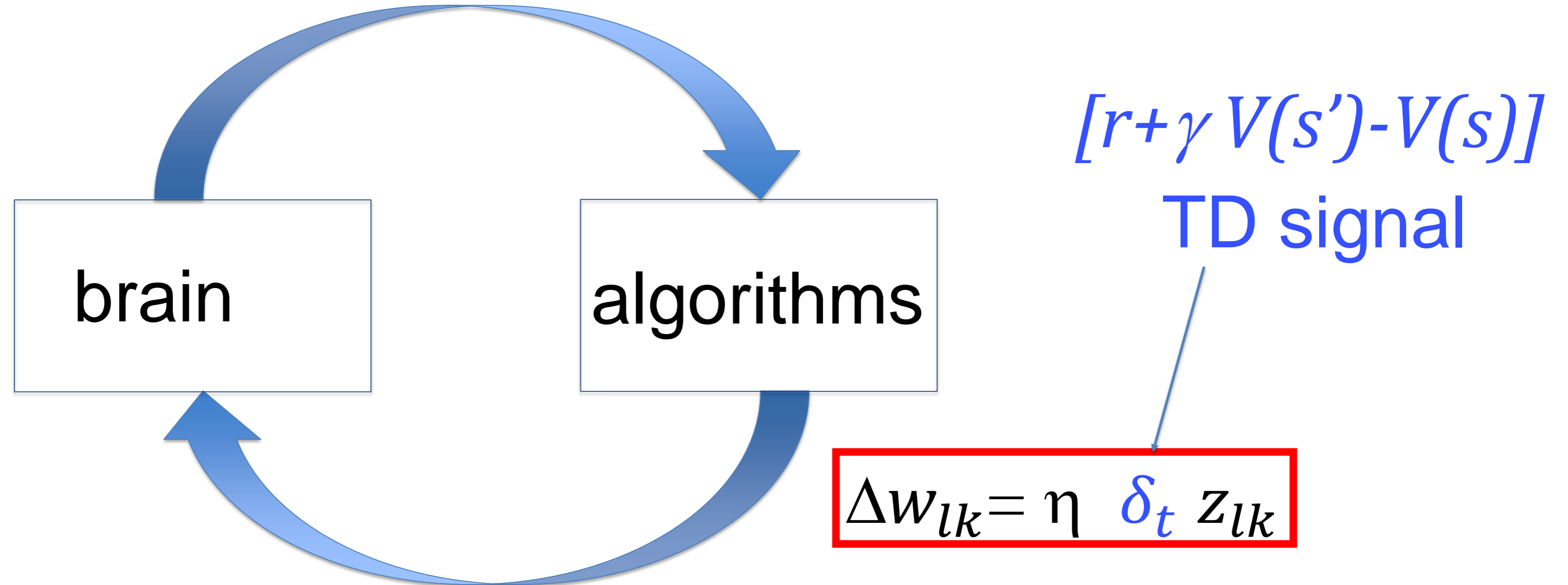
Review of algorithm with actor-critic architecture with eligibility traces.

The Advantage actor critic has parameter updates proportional to the TD error.

Reinforce/policy gradient has updates proportional to the momentary reward.

Apart from this difference, the overall structure of the two algorithms is very similar.

Review: Learning Rules of Reinforcement Learning



The learning rule of the **advantage** actor-critic with eligibility traces is consistent with a brain-like three-factor rule

Dopamine in the brain broadcasts TD signal!

Previous slide. Review

In the Advantage Actor Critic the learning signal (third factor) is the TD error.

It turns out that dopamine in the brain has a signature that is reminiscent of a TD signal (as shown last week). Dopamine can be seen as a (near-)global third factor.

Artificial Neural Networks and RL:

from brain-style computing to neuromorphic computing

Wulfram Gerstner

EPFL, Lausanne, Switzerland

1. Detour: Spiking Neural Networks (SNN)

Previous slide:

As we have seen in the previous lecture, neurons in the brain communicate by short pulses (spikes, also called action potentials).

How can we model Spiking Neural Networks (SNNs)?

The standard model of a single Spiking Neuron is the leaky integrate-and-fire model (LIF).

In the community of computational neuroscience it is usually written in continuous time.

For computer science applications, it is usually written in discrete time.

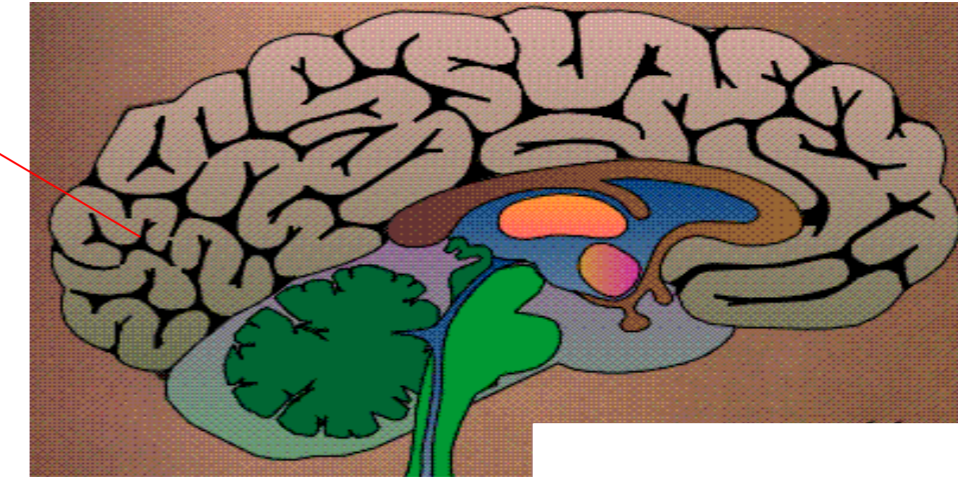
We show both versions. But first we review some biological information about spikes.

spike train recorded in the brain

electrode to measure a single neuron
in the brain of a mouse 'in vivo'.

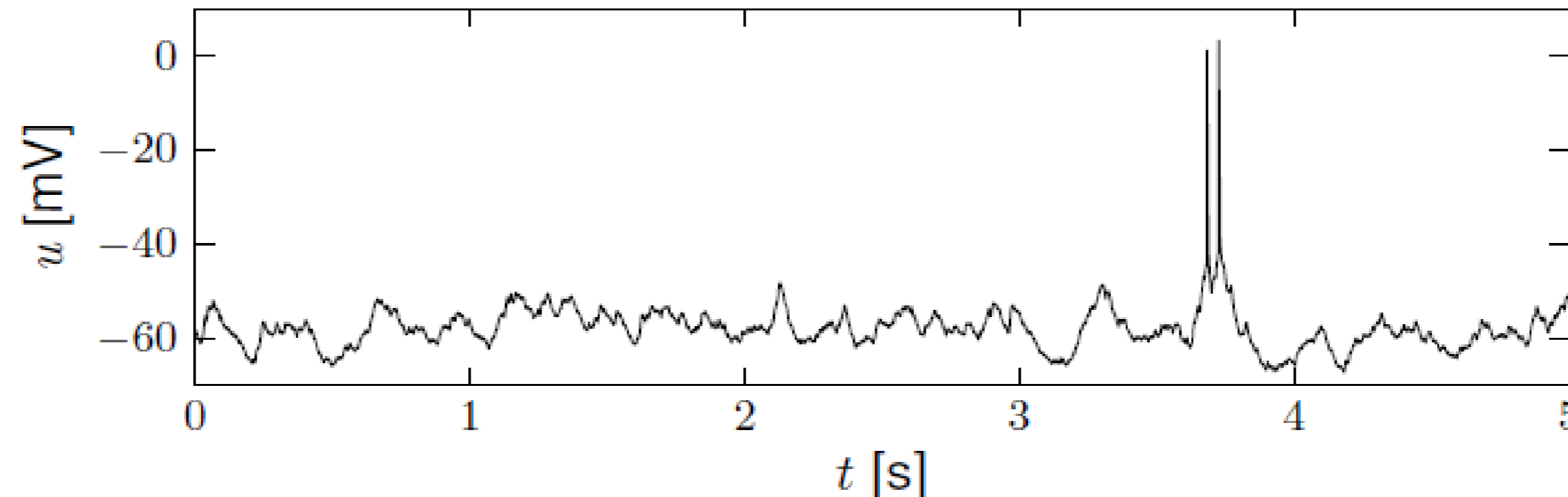
- *mouse has no specific task,*
- *no stimulus given*
- *spikes are 'rare'*

electrode



brain

awake mouse, cortex, freely whisking,



Lab of Prof. C. Petersen, EPFL *Crochet et al., 2011*

Previous slide.

Neurons have a voltage, sometimes called the membrane potential u .

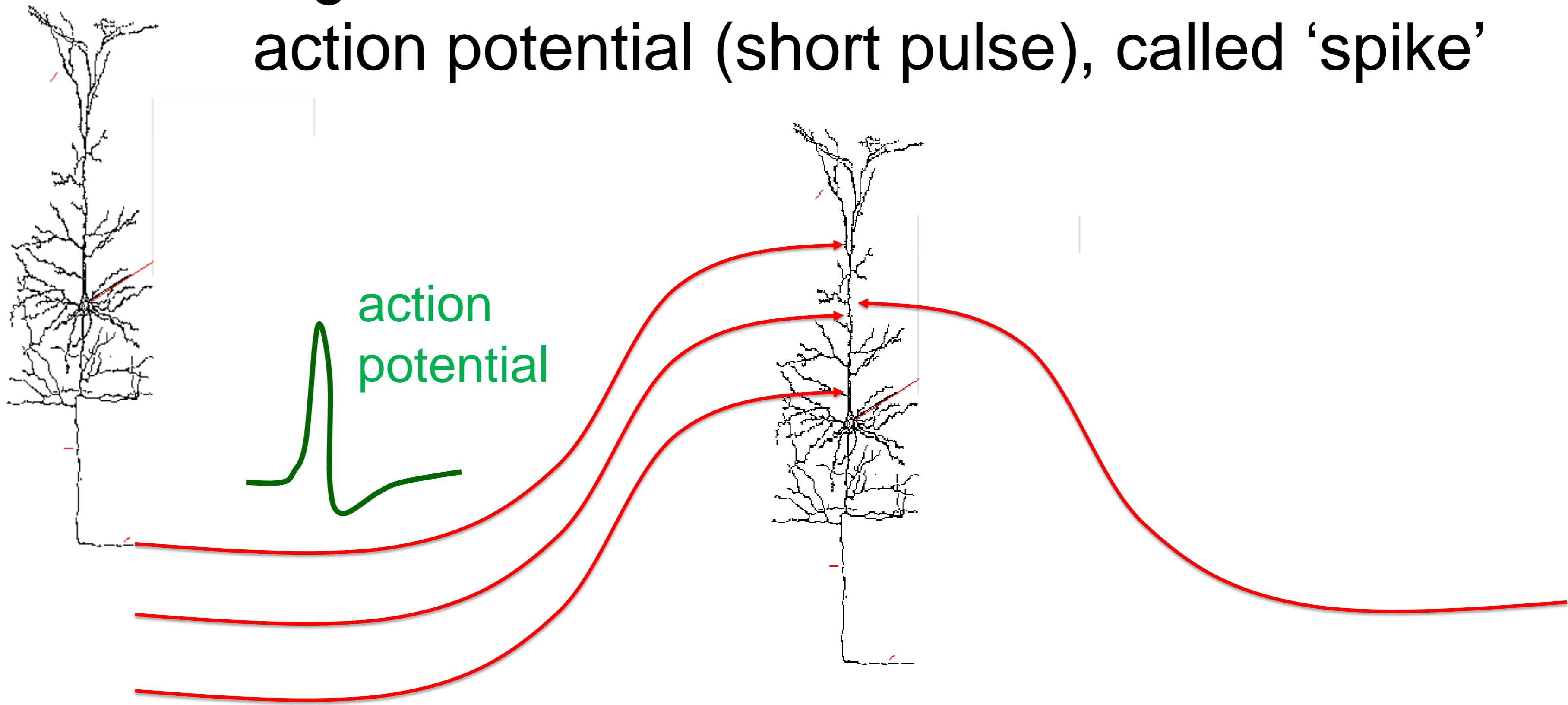
The image shows the voltage measurements of a single neuron over several seconds. We observe:

- 1) Occasionally there are spikes (large, but very short voltage peaks)
- 2) The interval between spikes is long (e.g., no spike for the first 3.5 seconds)

Review: The brain uses signal transmission by spikes

Signal:

action potential (short pulse), called 'spike'



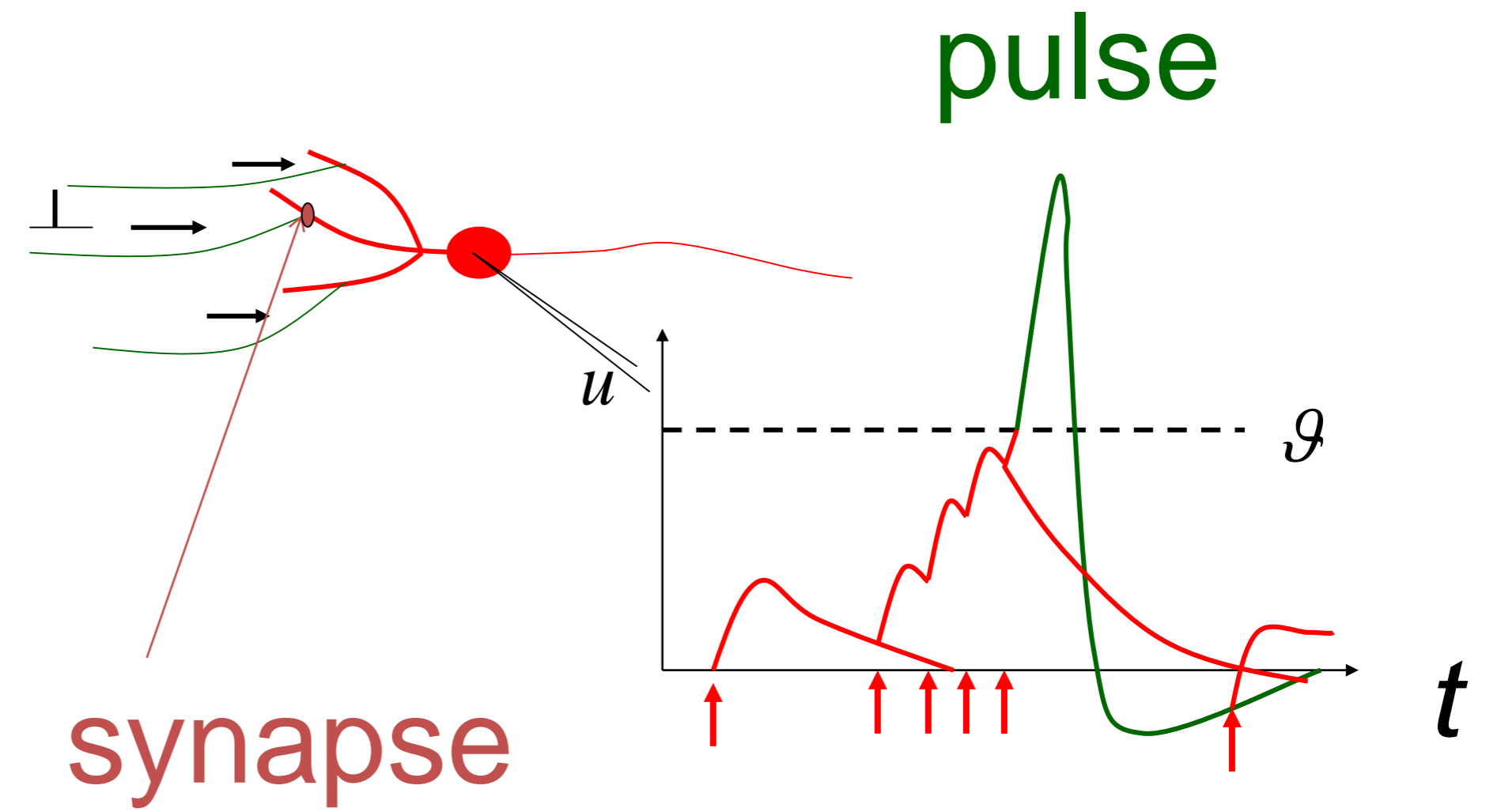
More than 1000 inputs

Previous slide. Review from Lecture 0

Signals are transmitted along wires (called axons). These wires branch out to make contacts with many other neurons.

Each neuron in cortex receives several thousands of wires from other neurons that end in 'synapses' (contact points) on the dendritic tree.

Review: neurons sum their inputs



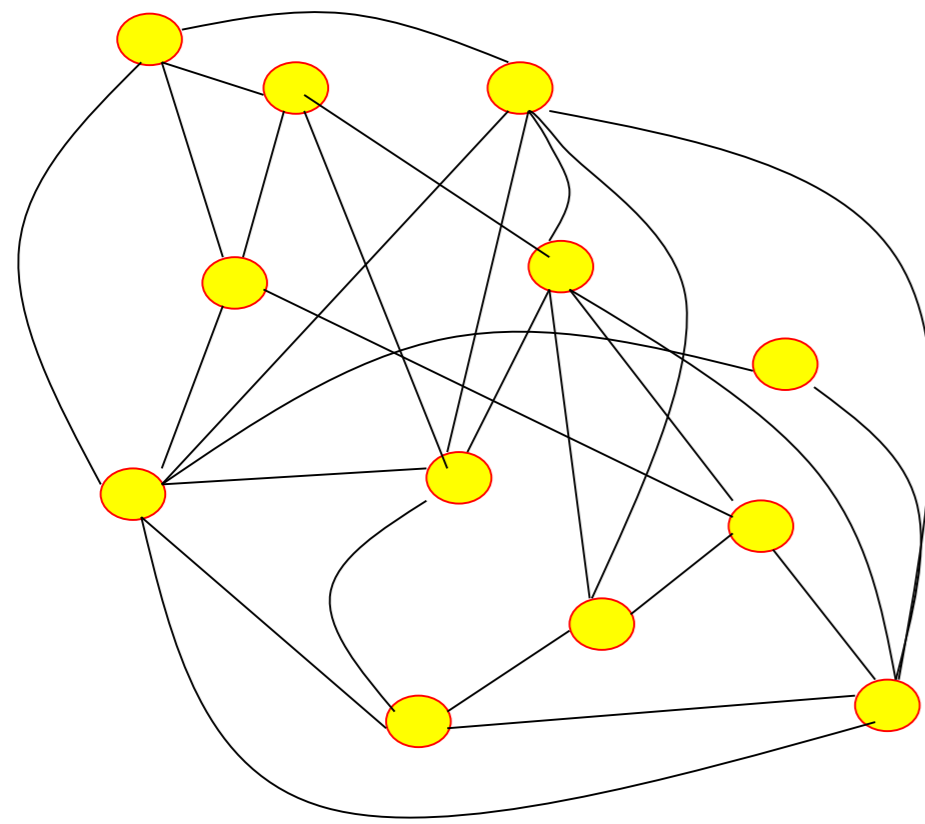
Previous slide. Review from Lecture 1

If a spike arrives at one of the synapses, it causes a measurable response in the receiving neuron.

If several spikes arrive shortly after each other onto the same receiving neuron, the responses add up.

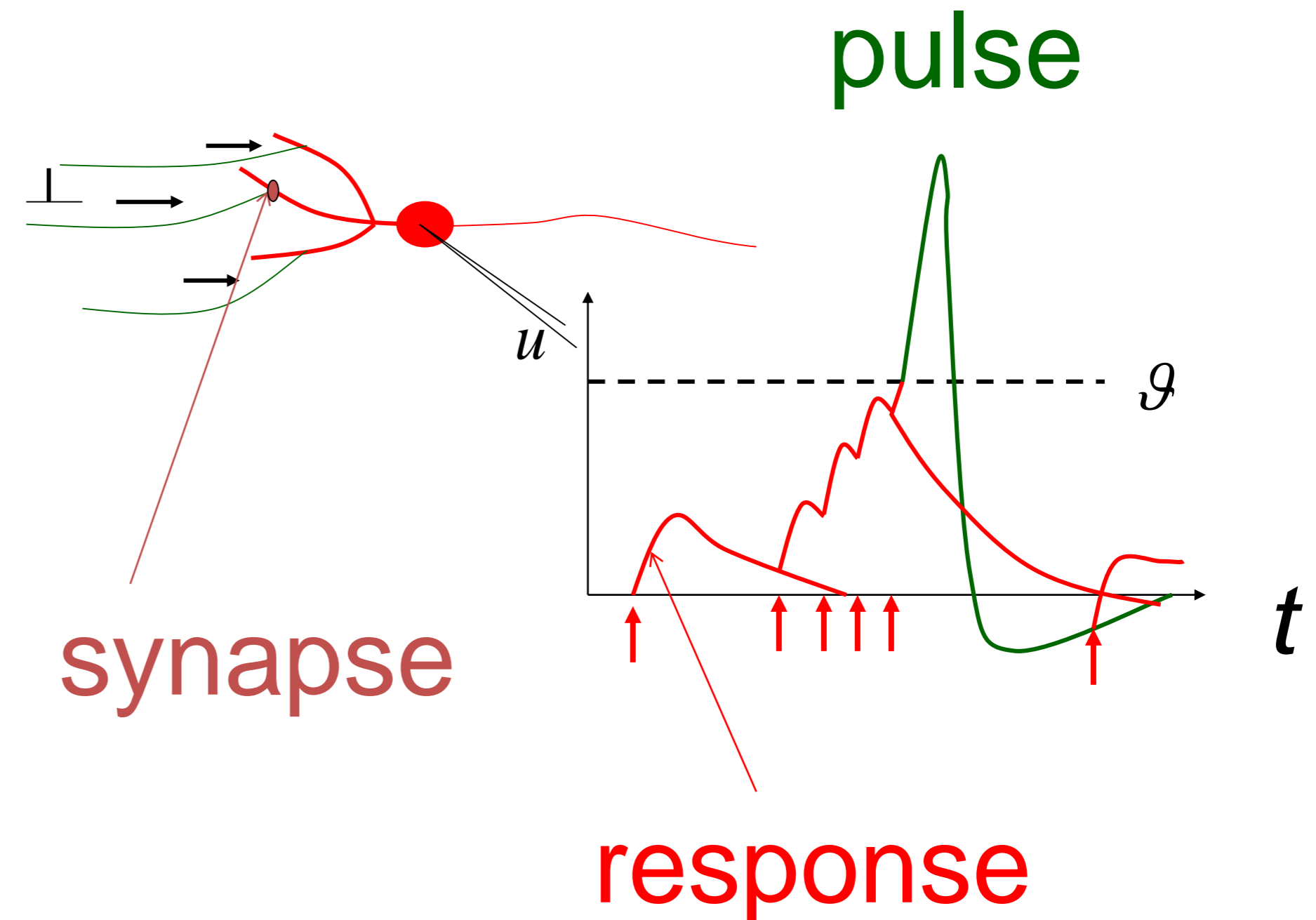
If the summed response reaches a threshold value, this neuron in turn sends out a spike to yet other neurons (and sometimes back to the neurons from which it received a spike).

Review: modeling of spiking neurons



- responses are added
- pulses created at threshold
- transmitted to other
- pulses are 'unitary events': shape of spike irrelevant

→ **Mathematical description:**
Integrate-and-fire neuron



Previous slide.

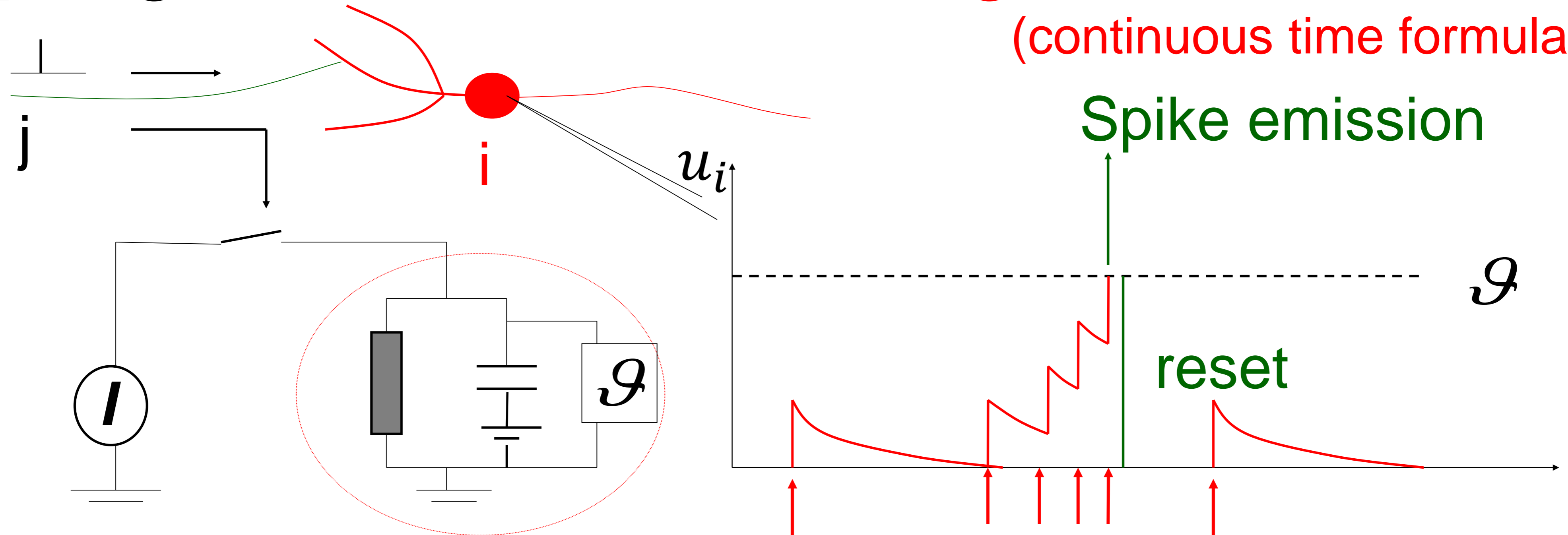
The fact that responses are added and then compared with a threshold is an aspect that is shared between real neurons, integrate-and-fire neurons, and artificial neurons in ANNs.

This is the essential ideal that we keep for the abstract mathematical model in the following.

Note that spikes are formal events – their duration can be reduced to zero. What matters is the fact whether a pulse is transmitted, yes or no.

Spiking Neural Network – Leaky Integrate-and-Fire Model

(continuous time formulation)



$$I_i(t) = \sum_j w_{ij} \delta(t - t_j^{pre})$$

$$\tau_m \frac{d}{dt} u_i = -u_i(t) + I_i(t)$$

linear

if $u_i(t) = \vartheta$ note spike+reset to $u(t) = 0$ **threshold**

Previous slide:

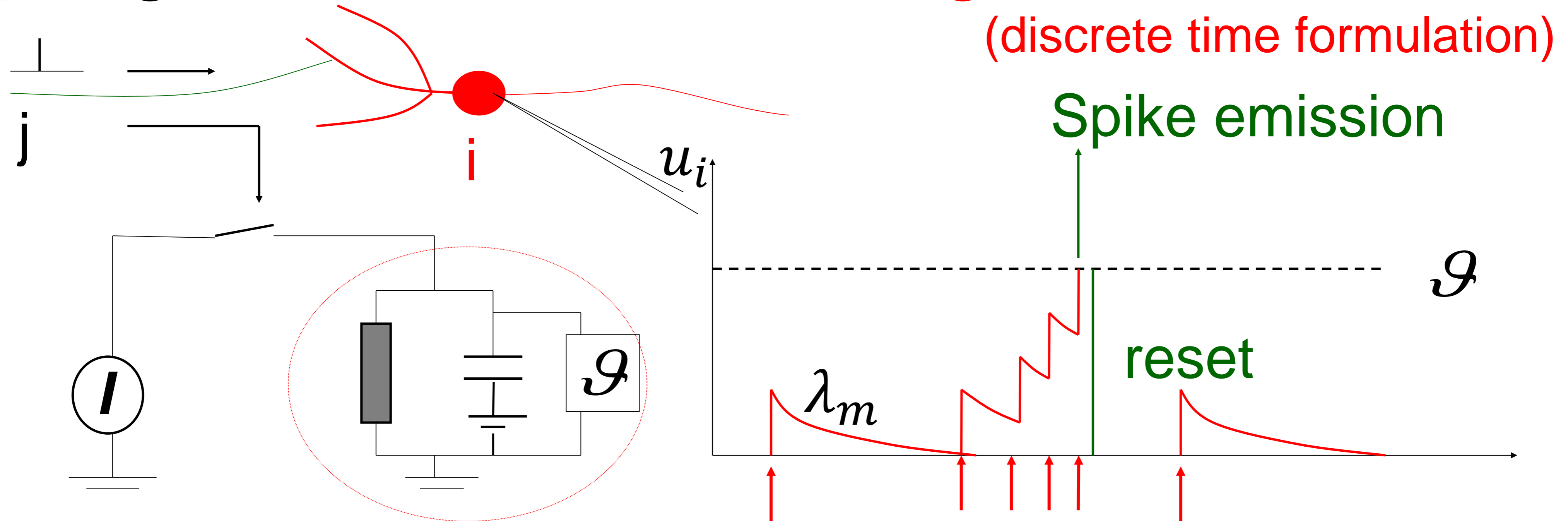
The Leaky integrate-and-fire model written in **continuous time** involves a LINEAR differential equation that can be interpreted as an electrical RC circuit charged by a current $I(t)$. This current $I(t)$ consists of short electrical pulses that present spike arrivals. The $\delta(t - t_j^{pre})$ denotes the Dirac delta function for each presynaptic spike arrival at times t_j^{pre} and w_{ij} are the weights. We can interpret w_{ij} as the charge delivered by the current pulse at time t_j^{pre} .

The linear equation is combined with a NONLINEAR FIRE-and-RESET process. If the variable u ('membrane potential of the neuron') reaches the threshold θ , then u is reset to zero.

Side Note: An electrical RC circuit consists of a capacitance C and a resistor R and has a time constant $\tau = RC$. Therefore after each short current pulse, the voltage (membrane potential) decays exponentially back to zero with time constant $\tau = RC$.

Spiking Neural Network – Leaky Integrate-and-Fire Model

(discrete time formulation)



discrete time steps

$$\Delta u_i = w_{ij} \quad \text{if } t = t_j^{pre}$$

$$u_i \leftarrow \lambda_m u_i$$

$$\text{if } u_i = \vartheta \quad u_i \leftarrow 0$$

linear, voltage jump

linear, decay with parameter λ_m

threshold \rightarrow fire+reset

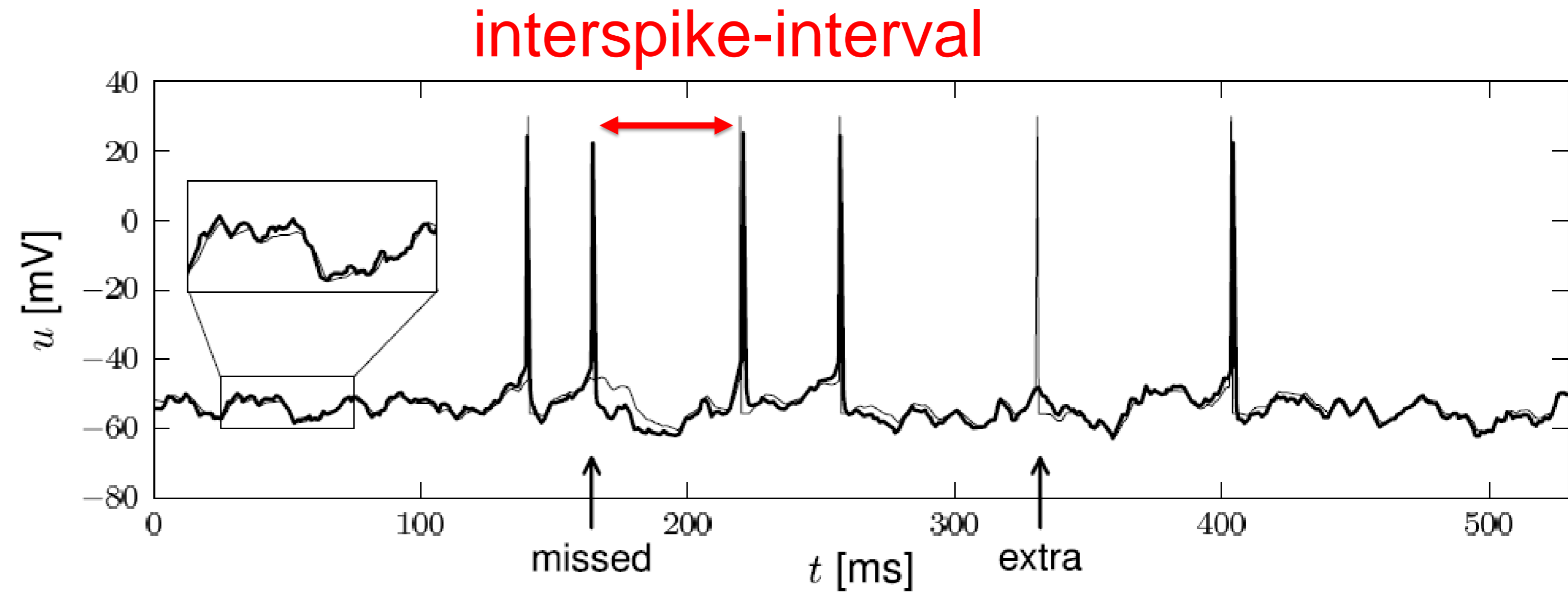
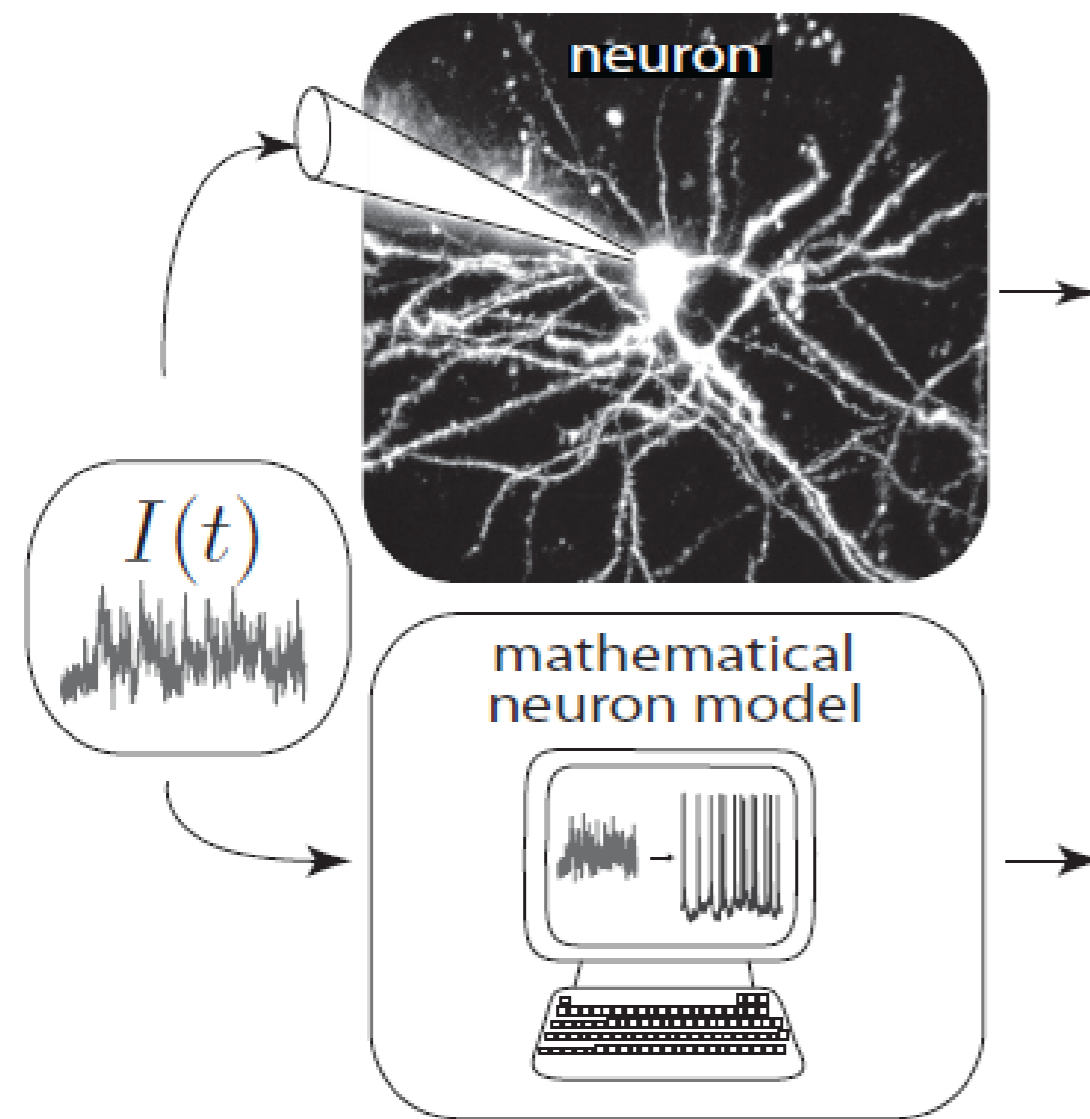
Previous slide:

The Leaky integrate-and-fire model written in discrete time (say time step $\Delta t = 1\text{ms}$) has two linear update steps:

- each presynaptic spike causes a jump of the voltage (membrane potential) by the synaptic weight w_{ij} .
- In each time step the membrane potential decays with a factor $\lambda_m < 1$. (Aside: If we compare with the previous equation in continuous time, we find that the factor is $\lambda_m = 1 - \left(\frac{\Delta t}{\tau_m}\right)$ where Δt is the time step.)

These linear update steps are combined with a NONLINEAR FIRE-and-RESET process. If the variable u ('membrane potential of the neuron') reaches the threshold θ , then u is reset to zero.

How good are integrate-and-fire models?



- **real neuron: thick line**
duration of spike about 1 ms
- **integrate-and-fire model: thin line**
duration of spike reduced to 0ms
followed by reset of u

Previous slide:

We can compare the membrane potential (voltage u) of a real neuron with that of an integrate-and-fire neuron for time-dependent input current. The same current drives either the real neuron, or the integrate-and-fire neuron.

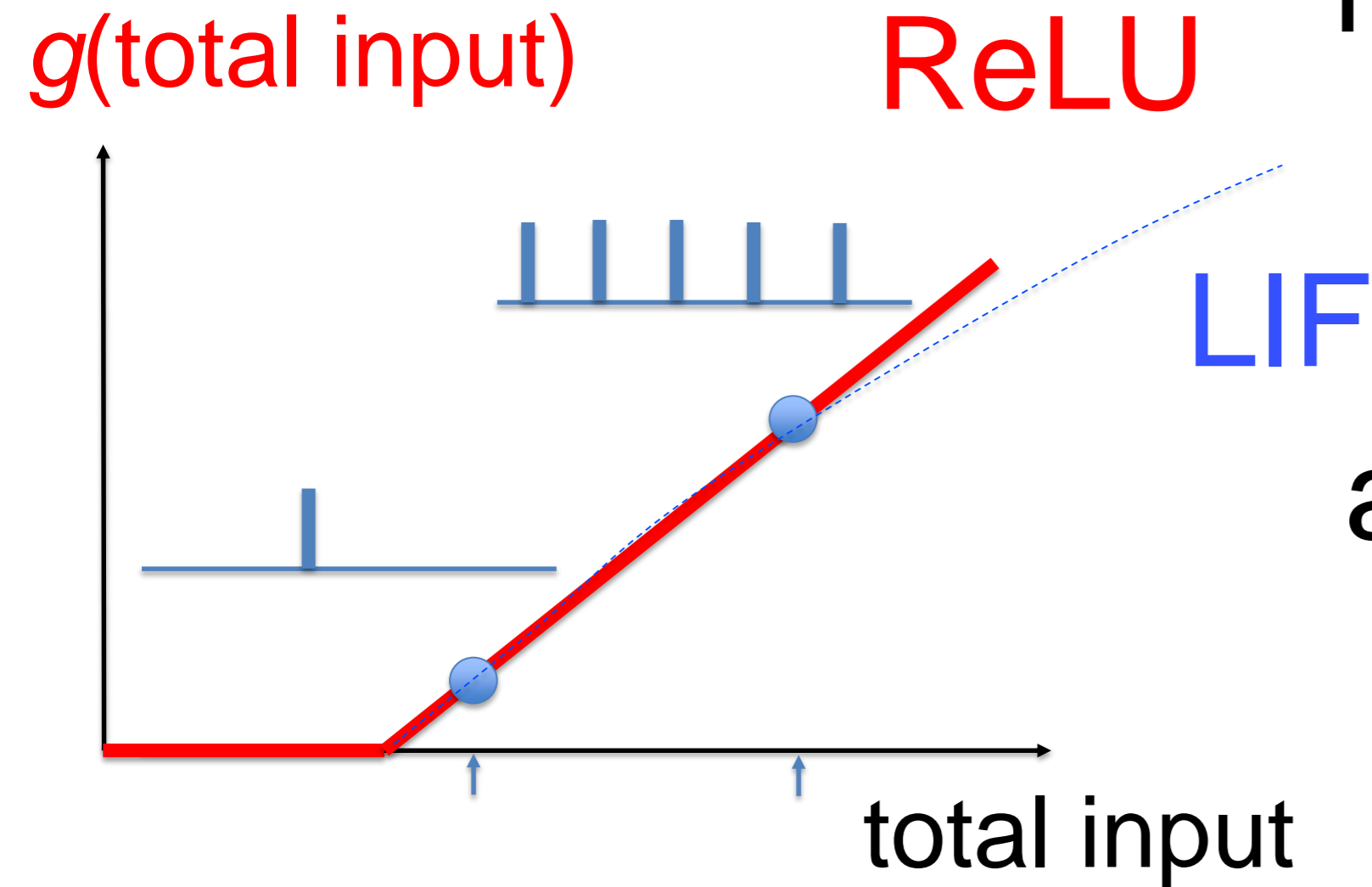
After optimization of parameters, the similarity is striking.

The LIF misses a few spikes, and also puts a few extra spikes.

For the LIF spike times are defined by the threshold crossing. The spike time is marked for visualization by a short vertical bar. After the spike the membrane potential is reset to a lower value.

Review: Modeling of artificial neurons

forget spikes: continuous activity x
forget time: discrete updates



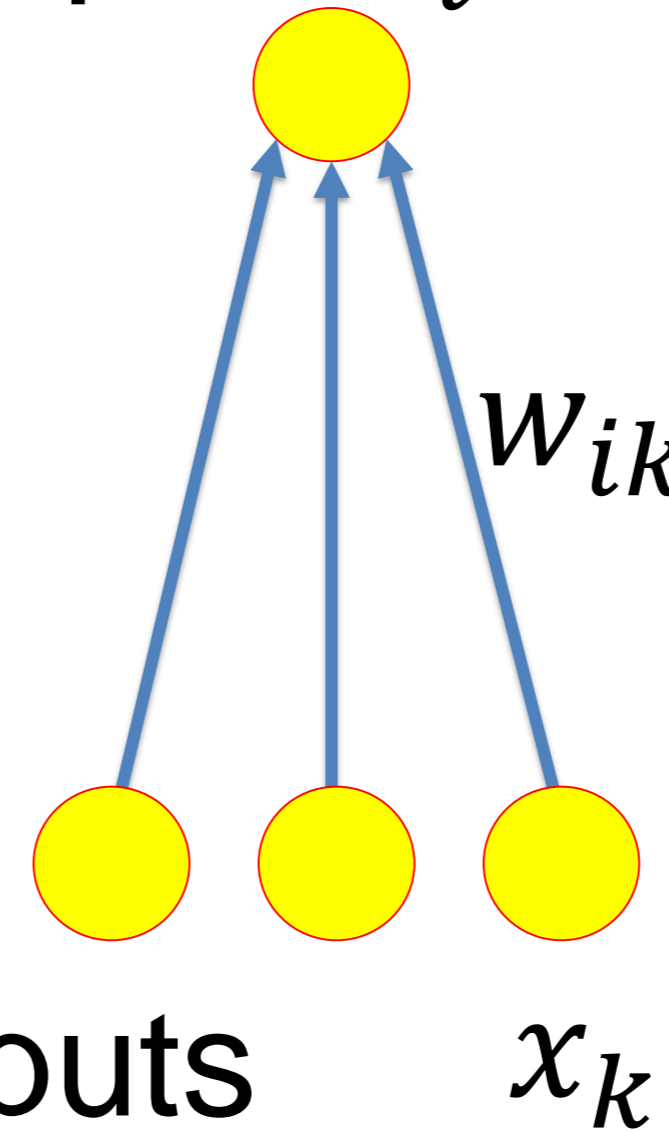
nonlinearity/threshold

$$\text{activity of output } x_i = g \left(\sum_k w_{ik} x_k \right)$$

output $g = \text{spikes per second}$
(firing rate)

leaky integrate-and-fire (LIF)

activity of inputs



weights =
adaptive
parameters

Previous slide.

The activity of inputs (or input neurons) is denoted by x_k

The weight of a synapse is denoted by w_{ik}

The nonlinearity (or threshold function) is denoted by g (could be a ReLU)

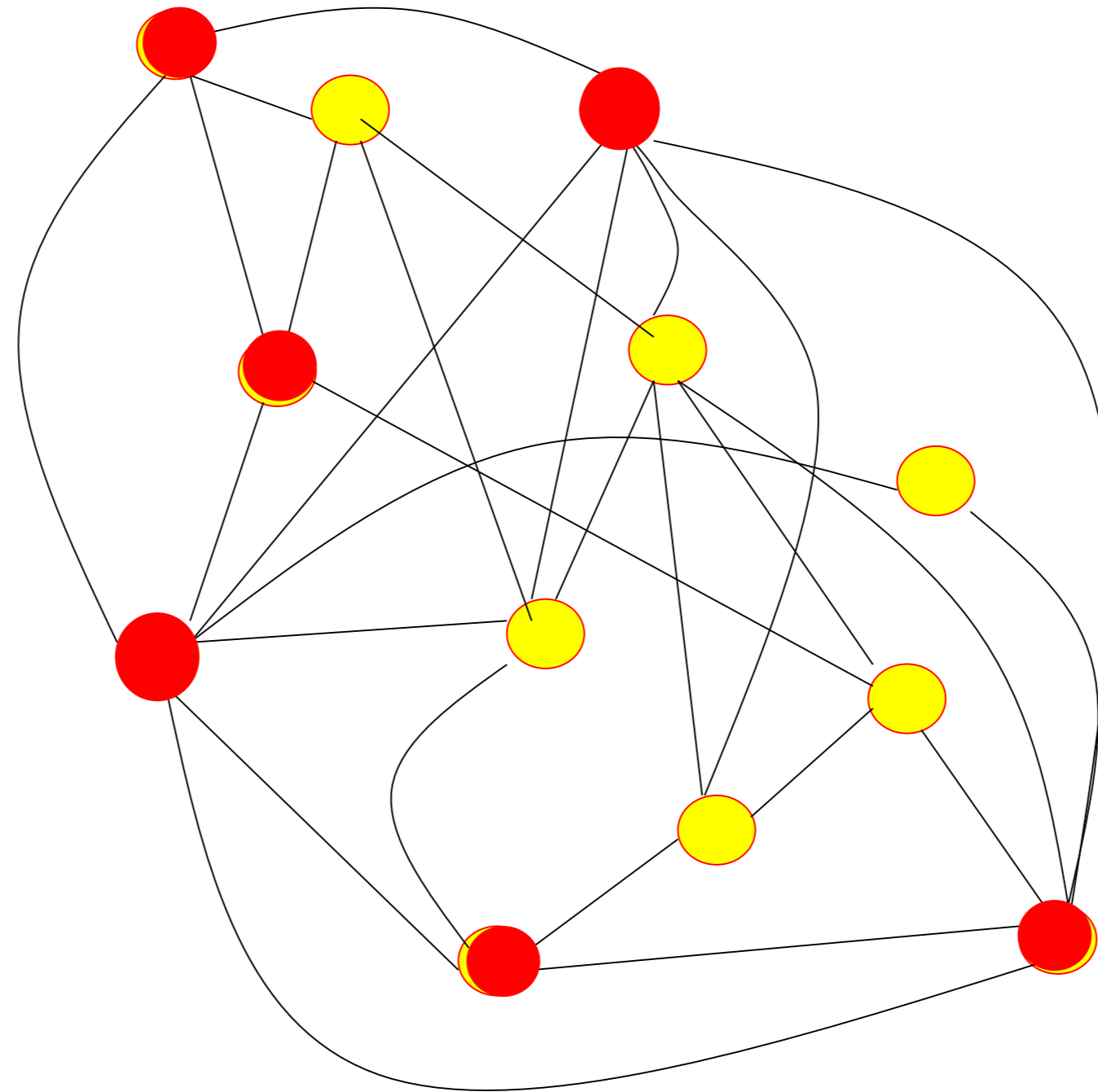
The output of the receiving neuron is given by $x_i = g \left(\sum_k w_{ik} x_k \right)$

The output for a leaky integrate-and-fire (LIF) neurons can be defined as the firing rate (number of spikes persecond).

The function g (firing rate for constant input) of a LIF is very similar to that of a ReLU.

The brain is a large recurrent network of spiking neurons

- Active neuron = spike emission



- spikes are rare events
- only events are transmitted → low bandwidth

Previous slide:

This slide has already been shown in the very first week. In a spiking neural network, most neurons are most of the time silent. Spikes are rare events.

This is exploited in spiking hardware.

Spiking Neural Networks

- spikes are rare binary events (yes/no)
- only events are transmitted → low bandwidth

Typical time scales in the brain:

- | | |
|--|--------------|
| - spike transmission time | 1ms |
| - spike duration | 1ms |
| - rise time of postsynaptic potential | 1ms |
| - decay time of postsynaptic potential | 10ms |
| - interspike interval (active neurons) | 50ms |
| - interspike interval (resting state) | 1000ms |
| - eligibility trace | 1000ms |
| - update of synapses | 10000ms= 10s |
| - decay of synapses | >10 000s |

Note:
Hardware could have a speed-up factor > 10 000, but respect the relative time scales.

Previous slide:

Neuronal dynamics occurs on different time scales.

The fast processes are in the range of 1ms.

However, most neurons emit most of the times no spikes, since interspike intervals are in the range of 30ms to 3s. Hence spikes are rare events.

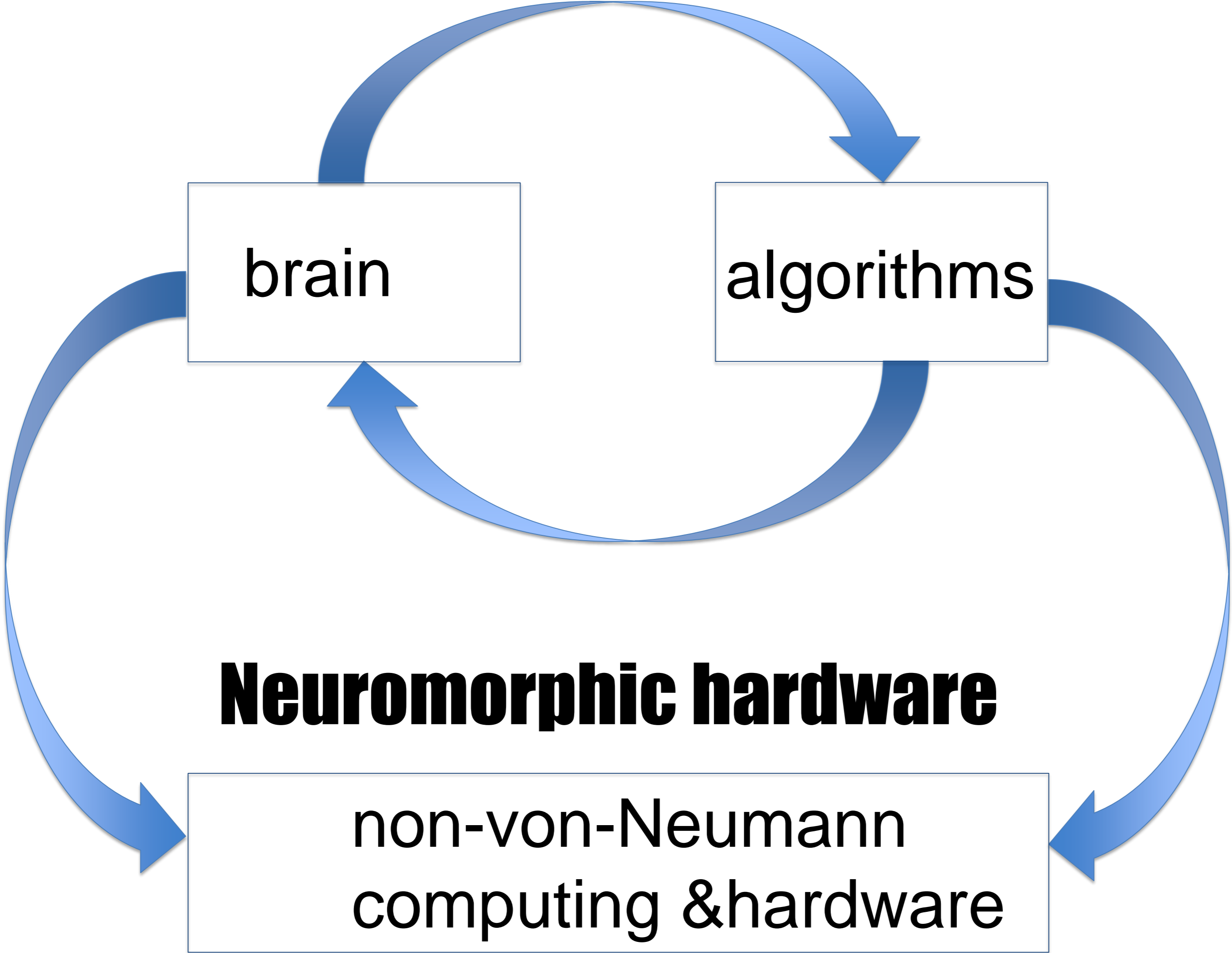
Consequence: transmission lines are rarely used.

But: the costly process is the transmission of the binary signal (spike) to thousands of other neurons.

Therefore we can save energy if only a few spikes are transmitted.

This can potentially lead to a large reduction of energy consumption.

Local Learning Rules, Spiking Neurons,



Previous slide:

The lecture last week covered the relation between learning rules used by the brain and those implemented in modern reinforcement learning algorithms.

This lecture will make the link to recent developments in modern neuromorphic computing architectures that are completely different than the class model of von-Neumann computing architectures.

One aspect is that these hardware approaches explore potential advantages of Spiking Neural Networks.

Another aspect is that they rely on local learning rules, in particular three-factor rules.

A third aspect is that they could potentially reduce energy consumption.

This lecture today provides an outlook onto current developments for specialized, bio-inspired chips that will eventually use much less energy than conventional chips. The category of chips is often called neuromorphic chips since they take inspiration from biological principles in neuroscience.

In particular, they use communication with spiking neurons and local learning rules.

Review: Policy Gradient as three-factor rule

parameter = weight W_{ij}

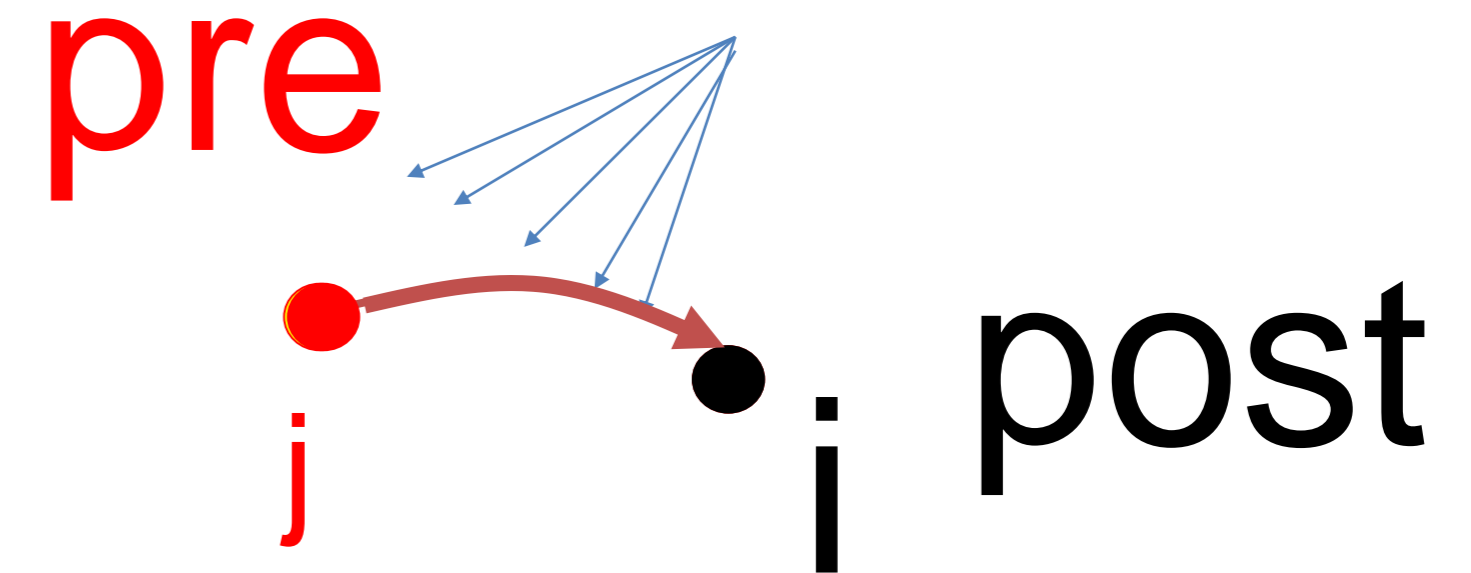
variable = eligibility Z_{ij}

pre- = sending neuron

post = receiving neuron

Stimulus

success



Change depends on pre and post

Two factors: eligibility trace proportional to **post** times **pre**

$$\Delta Z_{ij} = \overbrace{[a_i^t - \langle a_i(\vec{x}) \rangle]} \text{post} \cdot \text{pre} - \lambda_z Z_{ij}$$

Third factor: TD-error (**success**)

$$\Delta w_{ij} = \eta \delta_T Z_{ij}$$

synapse=
connection

postsynaptic factor is

'activity - expected activity'

linear decay with

parameter λ_z

Previous slide:

We have found that the update of weights of the actor in the **output layer** can be written as a three-factor learning rule:

- The presynaptic factor and the postsynaptic factor define the eligibility trace and there select the connection weight that is updated (pre and post are the two local factors).
- The third factor is global (independent of neuronal indices) and signals success.
- Success can be reward or the TD error (for the advantage actor-critic).

Review: Coincidence detection rule of STDP

$$\Delta w_{ij} \propto F(pre, post, SUCCESS)$$

Success signal:
TD error

success

Eligibility trace:

$$\Delta z_{ij} > 0 \text{ if 'STDP - condition'}$$

pre
j

post
i

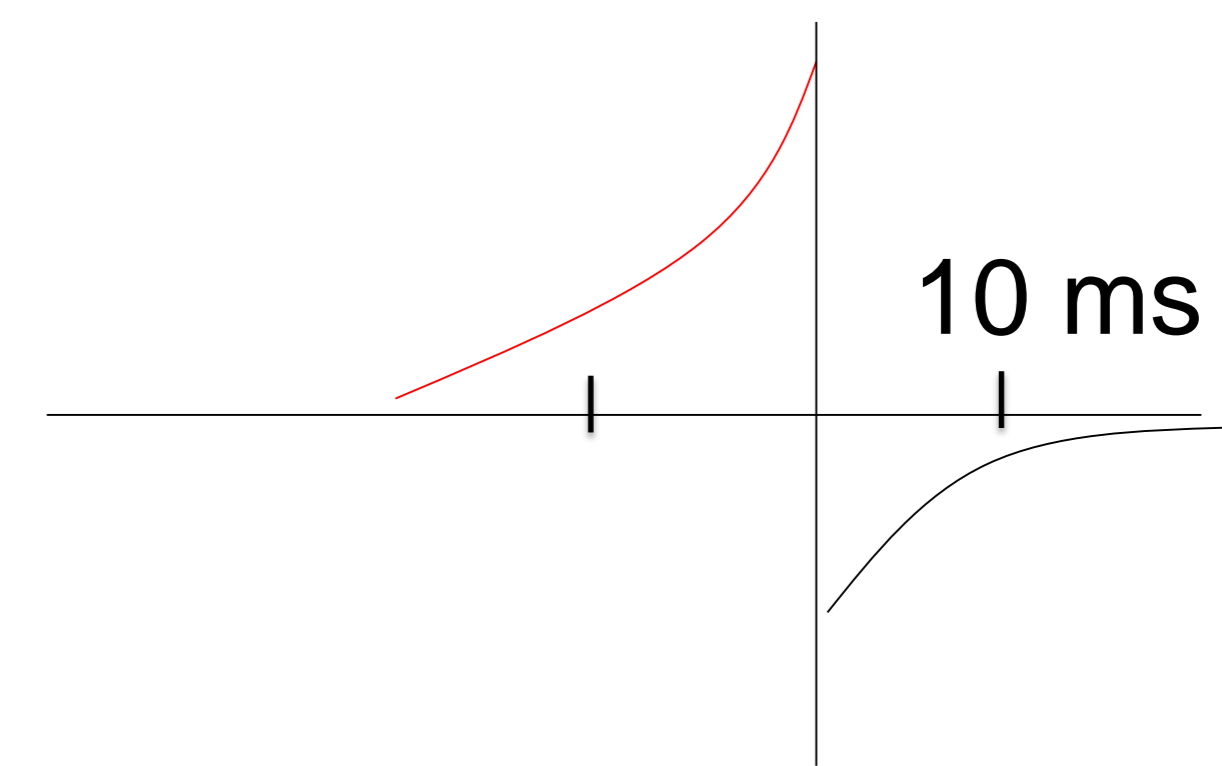
Weight

$$\Delta w_{ij} = z_{ij} S$$

Success signal

STDP condition

Hebb rule/eligibility trace



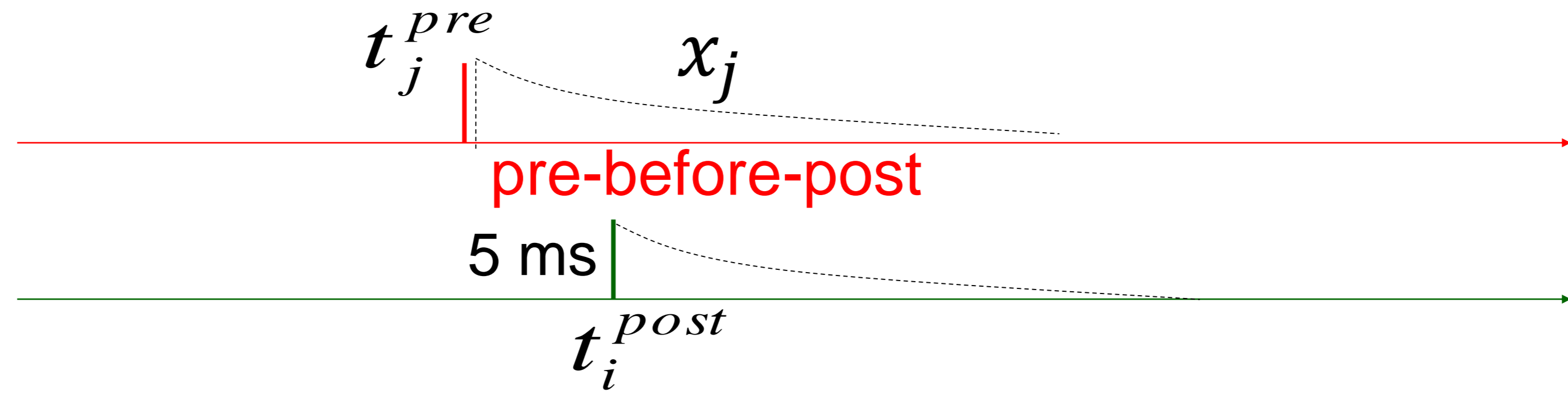
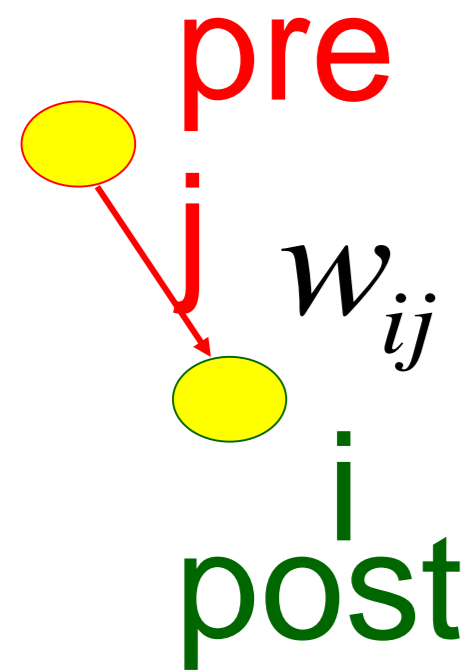
Xie and Seung 2003, Izhikevich, 2007; Florian, 2007; Legenstein et al., 2008, Fremaux et al. 2010, 2013

Previous slide:

A specific biologically plausible three-factor rule with eligibility traces would be the following:

- Spike-Timing-Dependent Plasticity (STDP) picks up coincidences between pre and postsynaptic spikes on a time scale of 10 milliseconds. STDP is hence a spike-based version of Hebbian learning.
- If furthermore the success signal arrives within one second, then the weight is updated.

'traces' for STDP: how to implement Hebb with spikes



Simple STDP model

(Gerstner et al. 1996,
Song-Miller-Abbott 2000, etc)

STDP condition

- (i) Trace left by presynaptic spike (discrete time steps of 1ms):

$$\Delta x_j = 1 \quad \text{if} \quad t = t_j^{pre}$$

$$x_j \leftarrow \lambda_+ x_j \quad \text{decays over 10ms}$$

- (ii) Update of eligibility trace at moment of postsynaptic spike

$$\Delta z_{ij} = x_j \quad \text{if} \quad t = t_i^{post}$$

$$z_{ij} \leftarrow \lambda_z z_{ij} \quad \text{decays over 1000ms}$$

- (iii) Update of weights prop to eligibility trace and Success S

$$\Delta w_{ij} = z_{ij} S \quad \text{decays never (or over days)}$$

pre-before-post

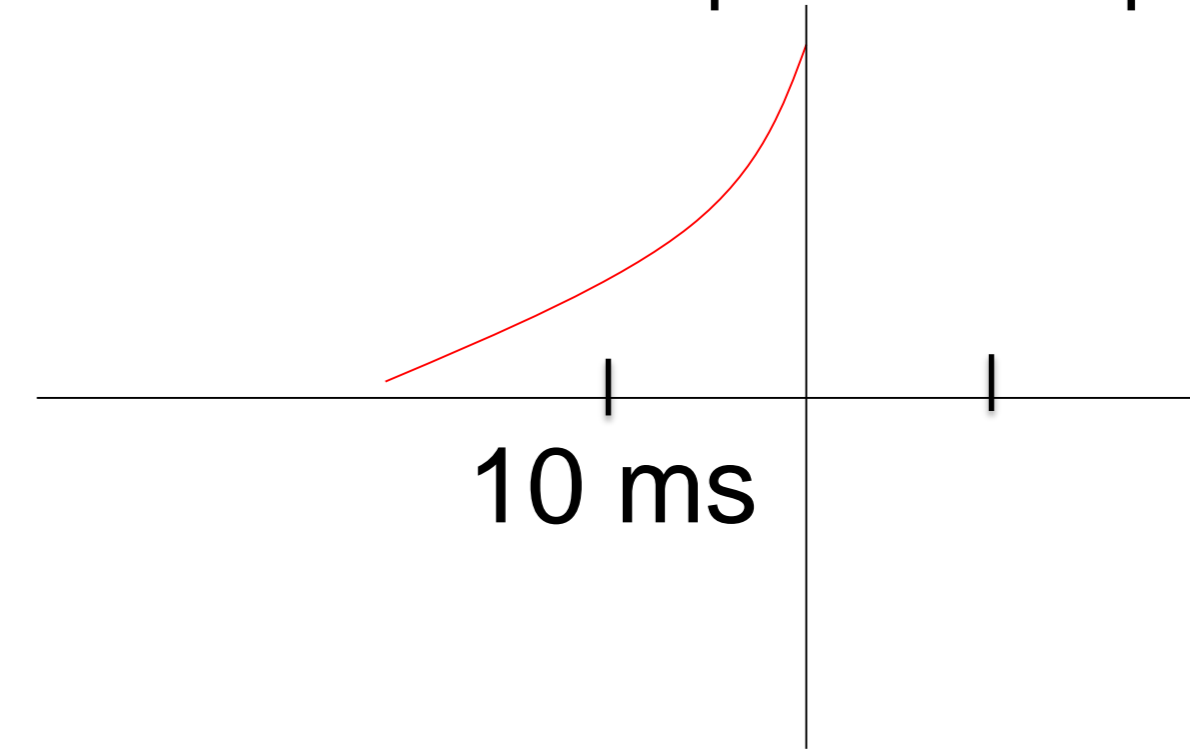
10 ms

Previous slide:

$$\tau_+ = 1/\lambda_+$$

- For example, the **pre-before-post 'HEBB' condition** can be implemented by saying that the presynaptic spike leaves an **exponential trace** (decaying with a time constant $\tau_+ = 1/\lambda_+$ of 10 millisecond); if the postsynaptic spike arrives a few milliseconds afterwards, it sets an eligibility trace that is proportional to the value of the **presynaptic trace**.
- The eligibility trace decays on slower time scale (time scale $\tau=1/\lambda_z= 1$ second).
- If the success signal arrives within one second, then the weight is updated.
- We can consider a special case (all time scales are the same discrete time step):
 - If (i) the eligibility trace has a time constant of $\tau_+ = 1/\lambda_+ 10\text{ms}$
 - (ii) the Hebbian STDP condition is one-sided with a time scale of 10ms
 - (iii) the discrete time step is 10ms,then the three-factor STDP is very similar to the three-factor policy gradient rule
- A two-sided STDP condition (plus for pre-before post and minus for post-before-pre) can be implemented by stating that the postsynaptic spike leaves another trace (postsynaptic trace) which leads to a negative updated of the eligibility trace at the moment of the next presynaptic spike arrival.

STDP condition: pre-before post



Review: Three-factor rules with eligibility trace

Three-factor rule defines a framework

x_j = activity-trace left by of presynaptic neuron

φ_i = activity-trace left by of postsynaptic neuron

Step 1: co-activation sets eligibility trace

$$\Delta z_{ij} = \eta f(\varphi_i) g(x_j)$$

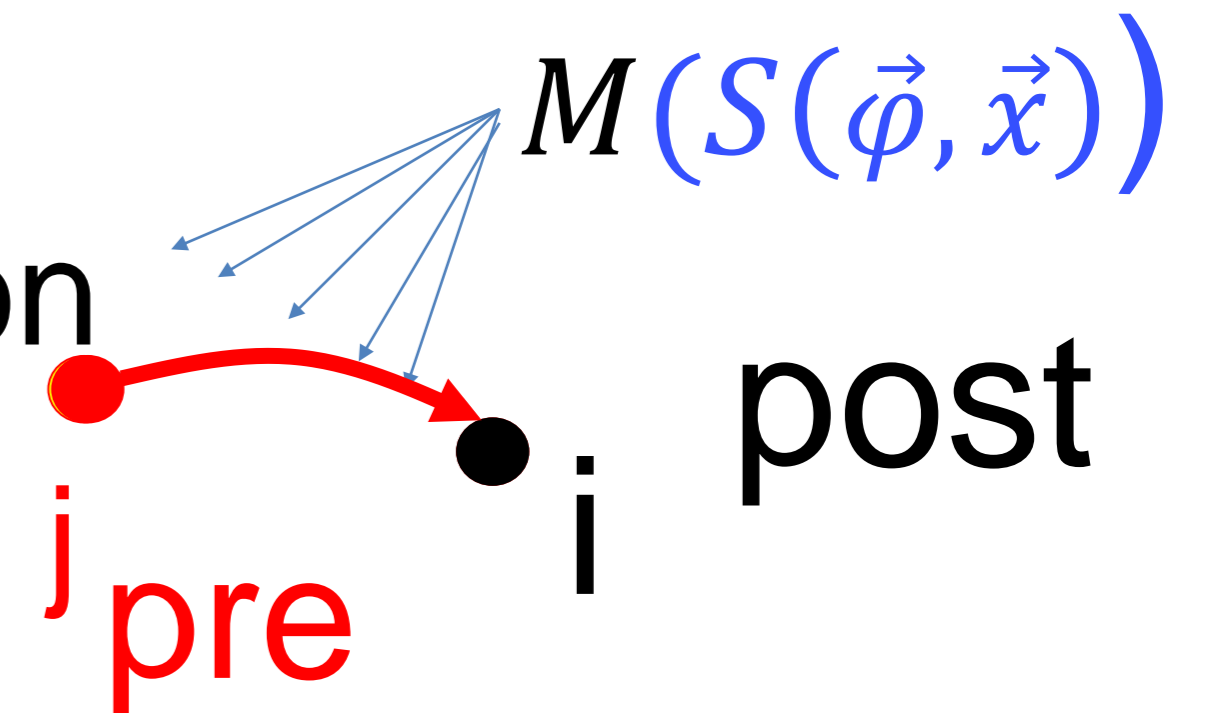
Step 2: eligibility trace decays over time

$$z_{ij} \leftarrow \lambda_z z_{ij}$$

Step 3: eligibility trace translated into weight change

$$\Delta w_{ij} = \eta M(S(\vec{\varphi}, \vec{x})) z_{ij}$$

Success signal



Previous slide:

There are many different Hebbian rules or STDP rules. Similarly, there is not a single three-factor rule. Rather three-factor rules are a framework formulated as follows:

- The trace left by presynaptic activity contributes some nonlinear factor $g(x_j)$
- The trace left by postsynaptic activity contributes some nonlinear factor $f(\varphi_i)$
- The eligibility trace e_{ij} is changed proportional to the two factors **f times g**
- The eligibility trace decays by a factor λ_z corresponding to a time scale of about one second
- Weights are updated proportional to eligibility trace **e_{ij} times M** with a modulator M that is a nonlinear function of the success S . The modulator is the 'third factor' in the update rule.
- The modulator M adjusts not only the learning speed but also the direction of change. In other words, the sign of the update (increase/decrease) depends on the sign of M .


Three-factor Learning Rules and Spiking Neurons



[] STDP is an acronym for Spike-Timing-Dependent Plasticity




[] Spikes are pulses that last less than $10\mu\text{s}$



[] A spike of a presynaptic neuron that arrives 5ms before a postsynaptic one sets leaves a 'trace' for a few ms

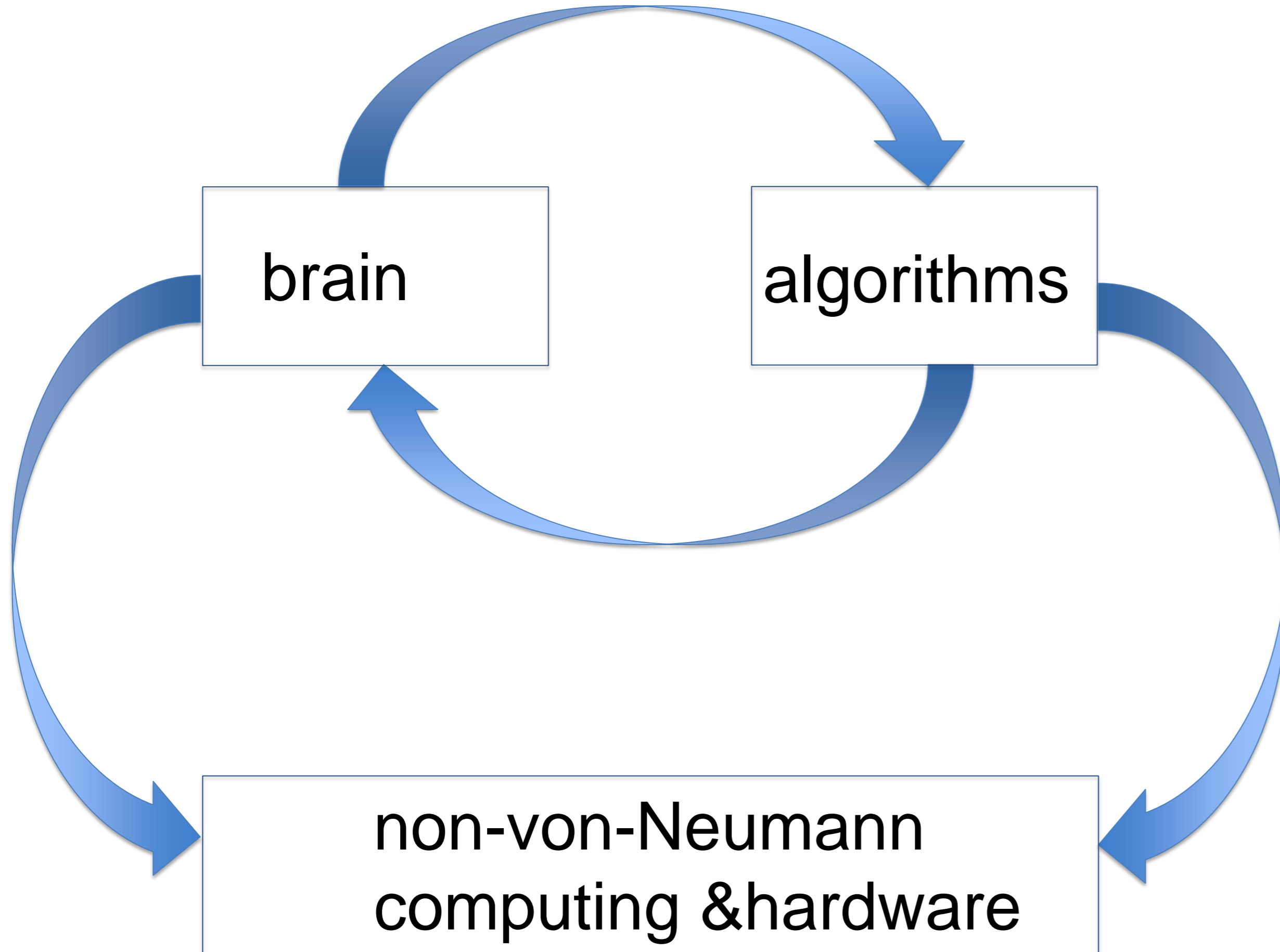


[] The eligibility trace of three-factor rules lasts for at least 10 minutes



[] Neurons in the brain exhibit interspike-intervals that are rarely longer than 50ms

Three-factor Learning Rules and Spiking Neurons



Summary:

- Neurons communicate by short pulses
- Pulses last 1ms
- Pulses are rare events
- A pulse timing pre-before-post (within 20ms) sets an eligibility trace
- The eligibility trace decay over 1s
- Dopamine, a global neuromodulator, sends a TD signal

The 80-percent question again:

[] Today, up to here, for the 3-factor framework and Spiking Neural Networks have the feeling that I understood at least 80 percent of the material

Previous slide:

After this introduction to spiking neurons, and review of three-factor rules, we make a small detour to an application that you have seen already at several occasions.

And then we are prepared to look at the first hardware implementation.

Wulfram Gerstner

EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL: **from brain-style computing to neuromorphic computing**

1. Detour: Spiking Neural Networks (SNN)
2. **Example: Navigation in a Maze (Model Study)**
 - What is the task?
 - How are 'states' represented?
 - How are 'actions' represented?
 - How is the 'learning rule' represented

Previous slide.

We said that the three factor rule, dopamine, TD signals, value functions now all fit together. Let's apply this to the problem of navigation in a maze.

For biological plausibility we have to consider:

- Representation of states
- Representation of actions
- Representation of TD signal and learning rule

Example of modeling for biology: Navigation

- What is the task?
- How are 'states' represented?
- How are 'actions' represented?
- How is the 'learning rule' represented

→ Use three-factor rule as a framework for learning

Big Question: Do we need an 'exact' actor-critic update rule?

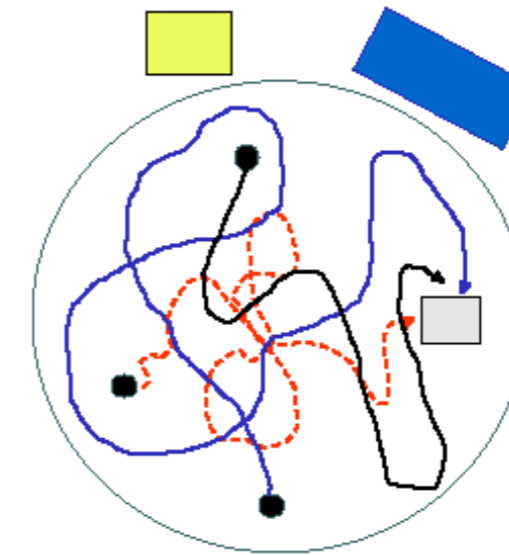
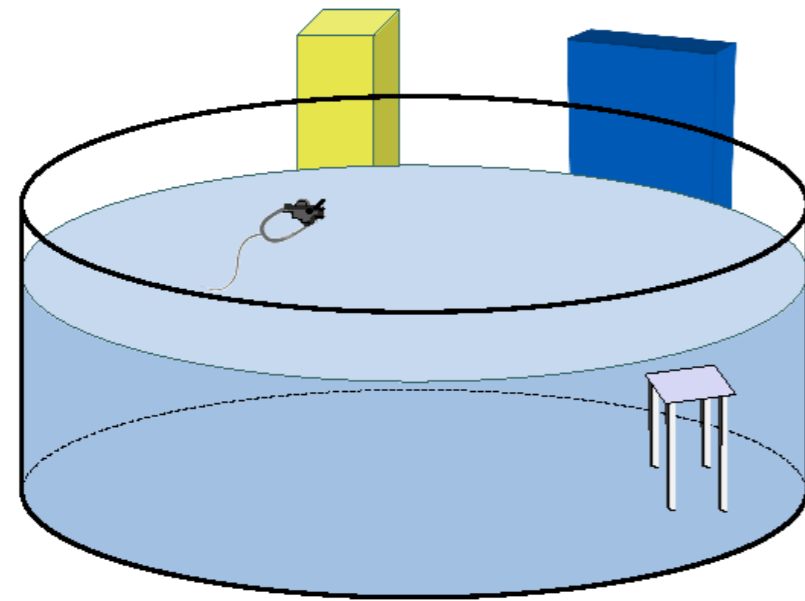
- can we replace softmax by something else?
- can we replace condition for 'coincidence' pre-post by something else?
- can we replace TD-error by something else?
- can we work with spiking neurons?

Previous slide.

Can we work with spiking neurons and solve a biological RL task using components that look 'plausible'?

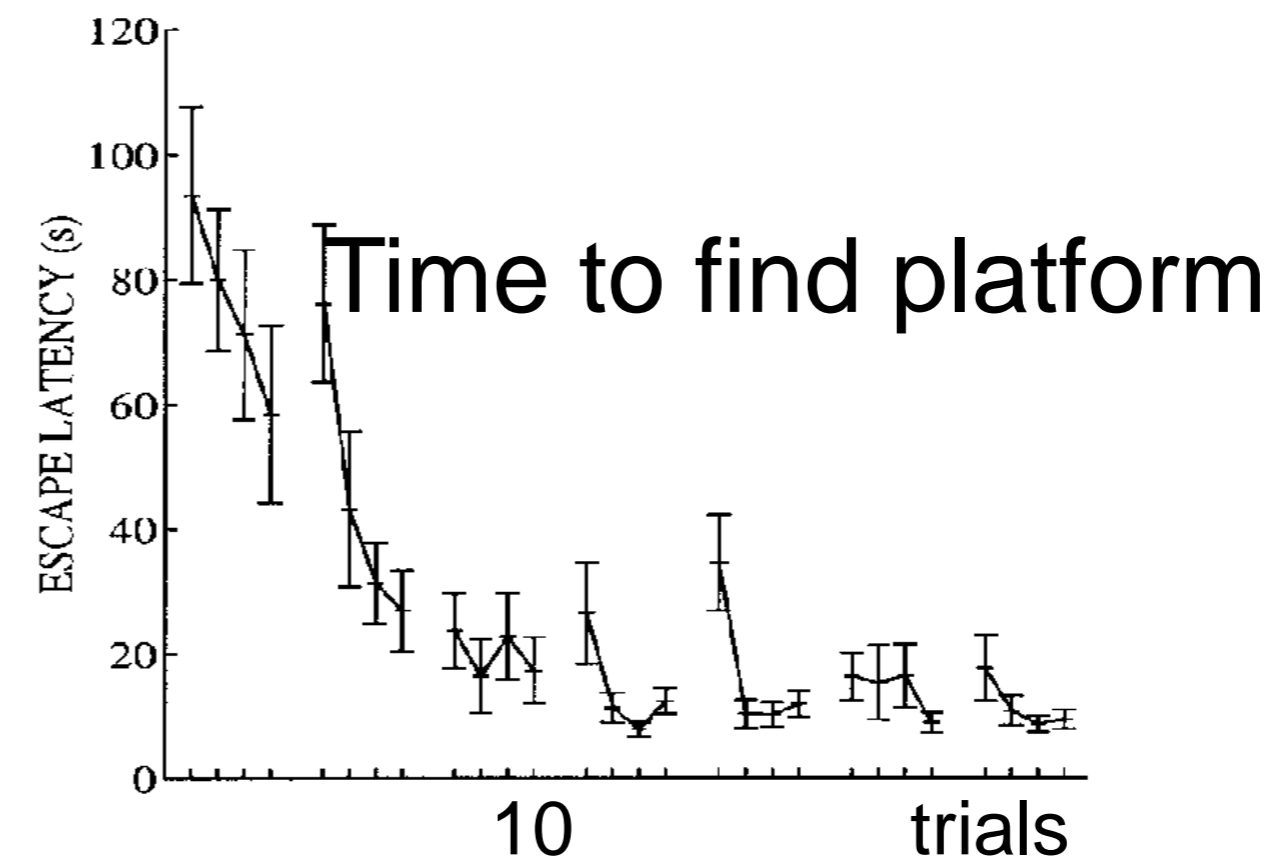
Review: TASK = conditioning in the Morris Water Maze

Morris Water Maze



Rats learn to find the hidden platform

(Because they like to get out of the cold water)



Foster, Morris, Dayan 2000

Previous slide.

Behavioral experiment in the Morris Water Maze.

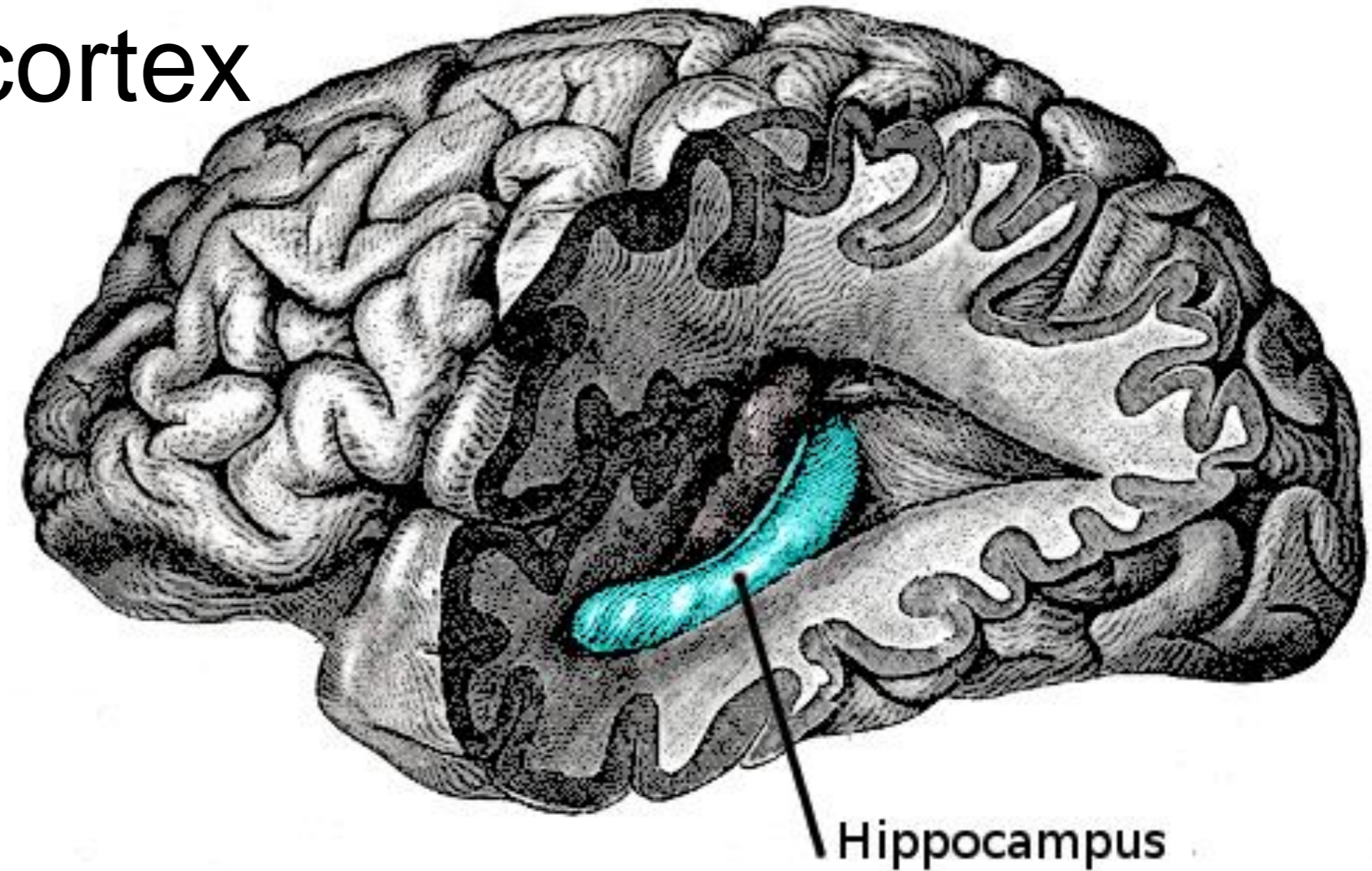
The water is milky so that the platform is visible.

After a few trials the rat swims directly to the platform

Review: Representation of momentary state: hippocampus

Hippocampus

- Sits below/part of temporal cortex
- Involved in memory
- Involved in spatial memory



Spatial memory:

knowing where you are,

knowing how to navigate in an environment

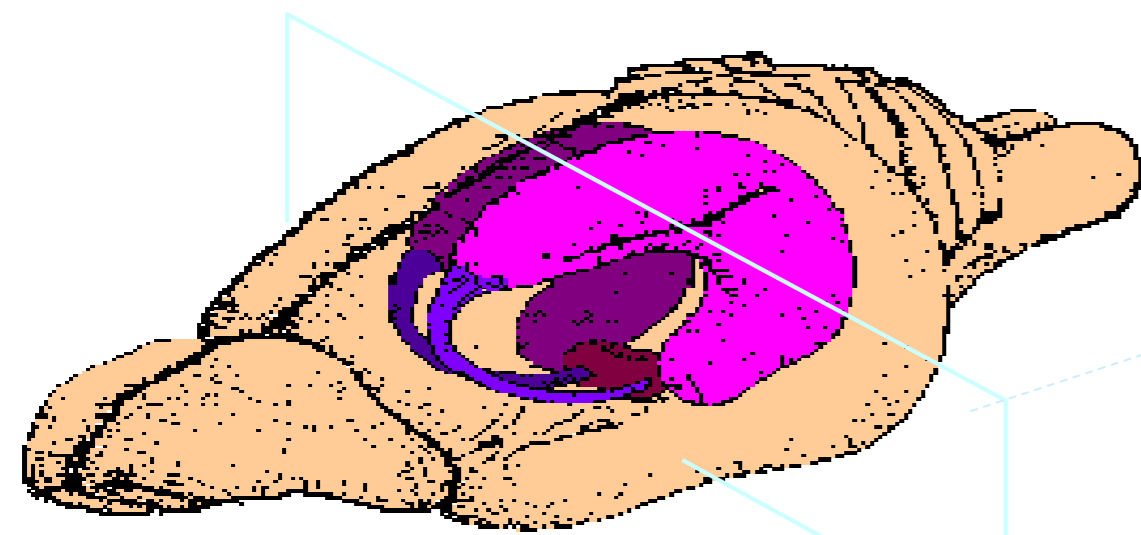
fig: Wikipedia

[Henry Gray](#) (1918) *Anatomy of the Human Body*

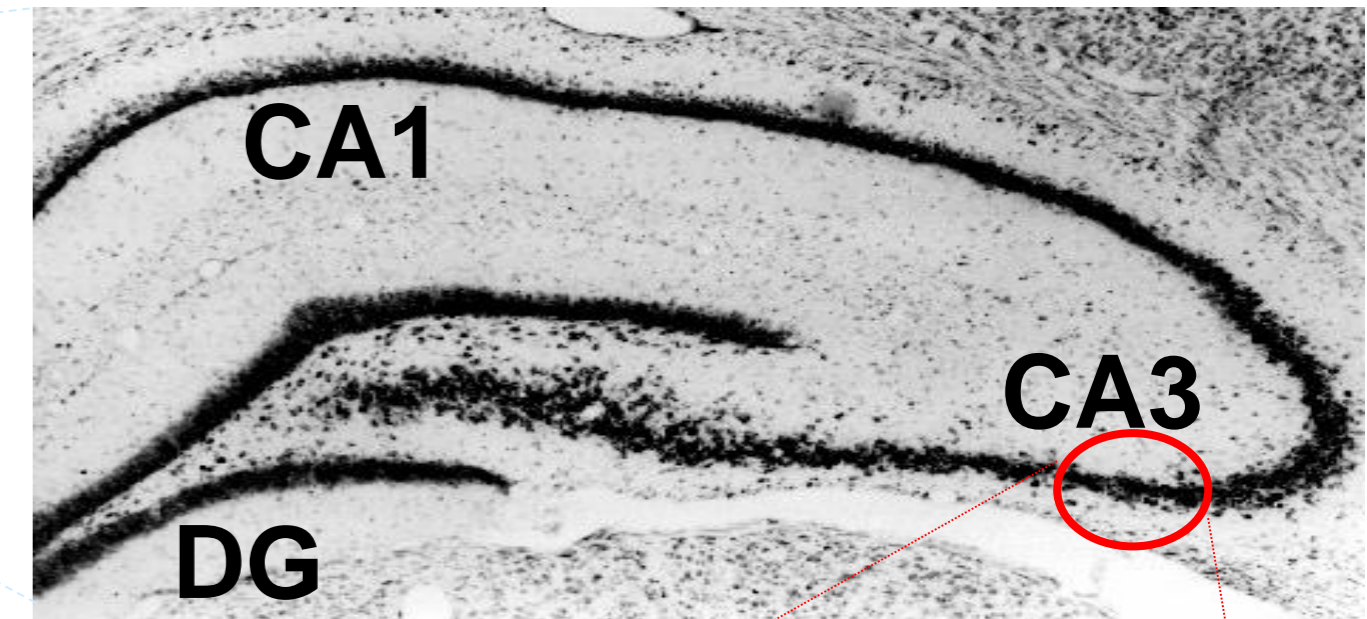
Previous slide.

the problem of navigation needs the spatical representation of the hippocampus.

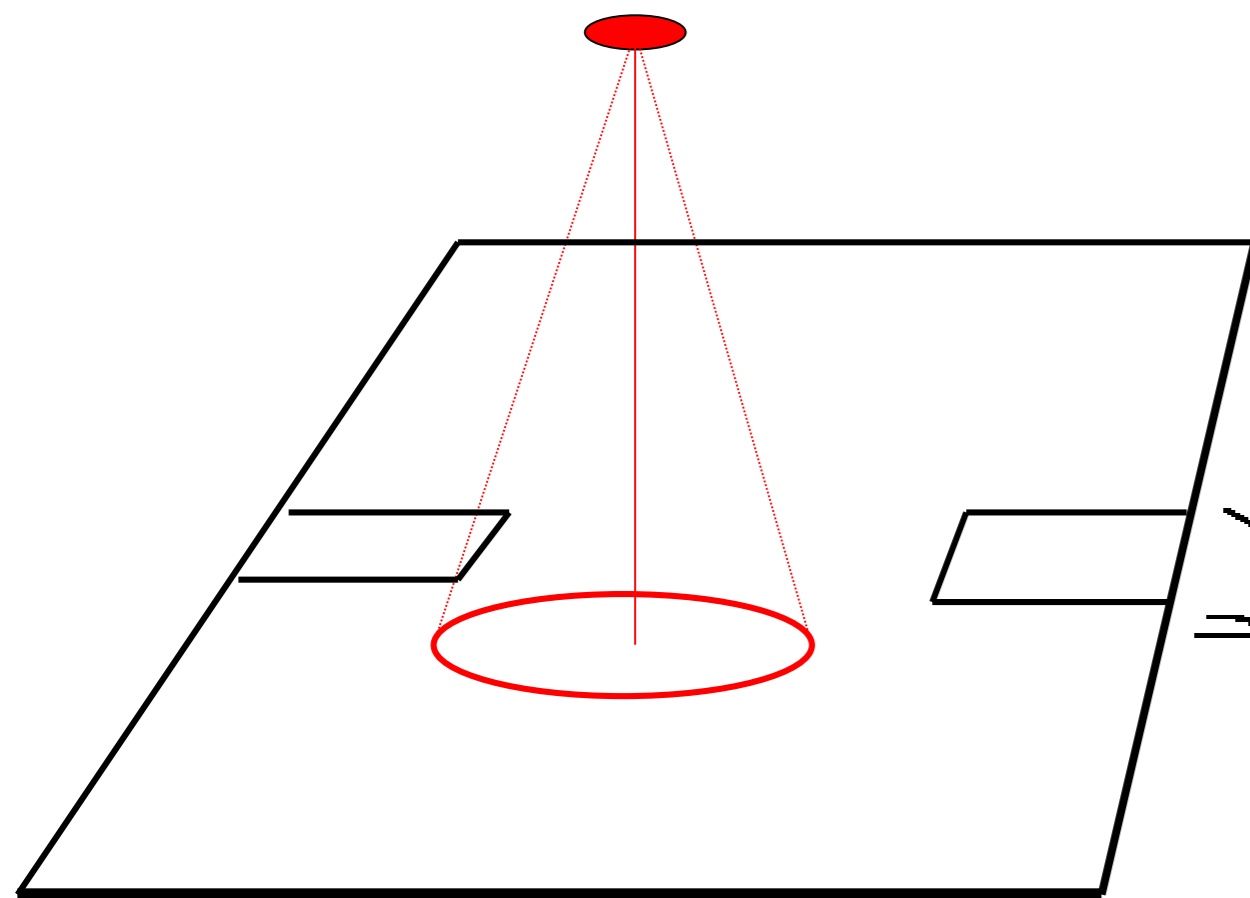
6. Representation of states: Place cells in rat hippocampus



rat brain

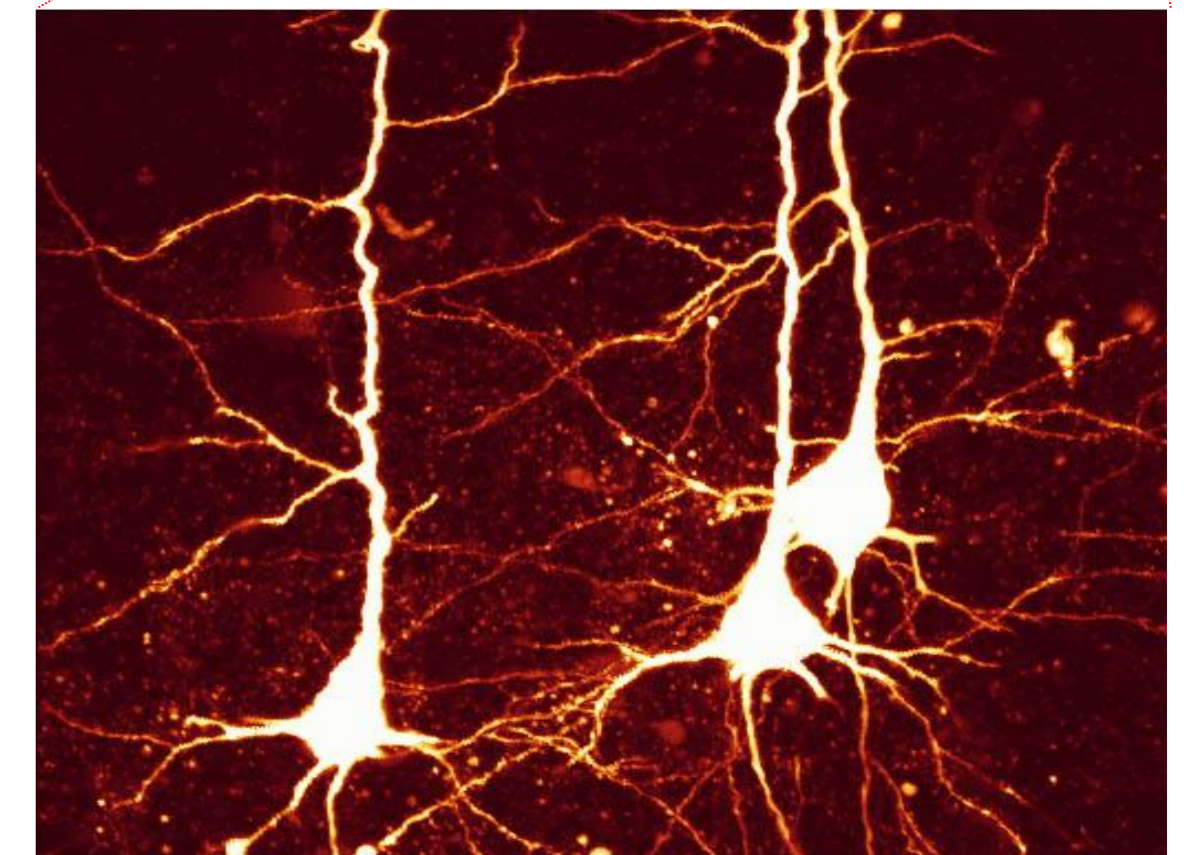
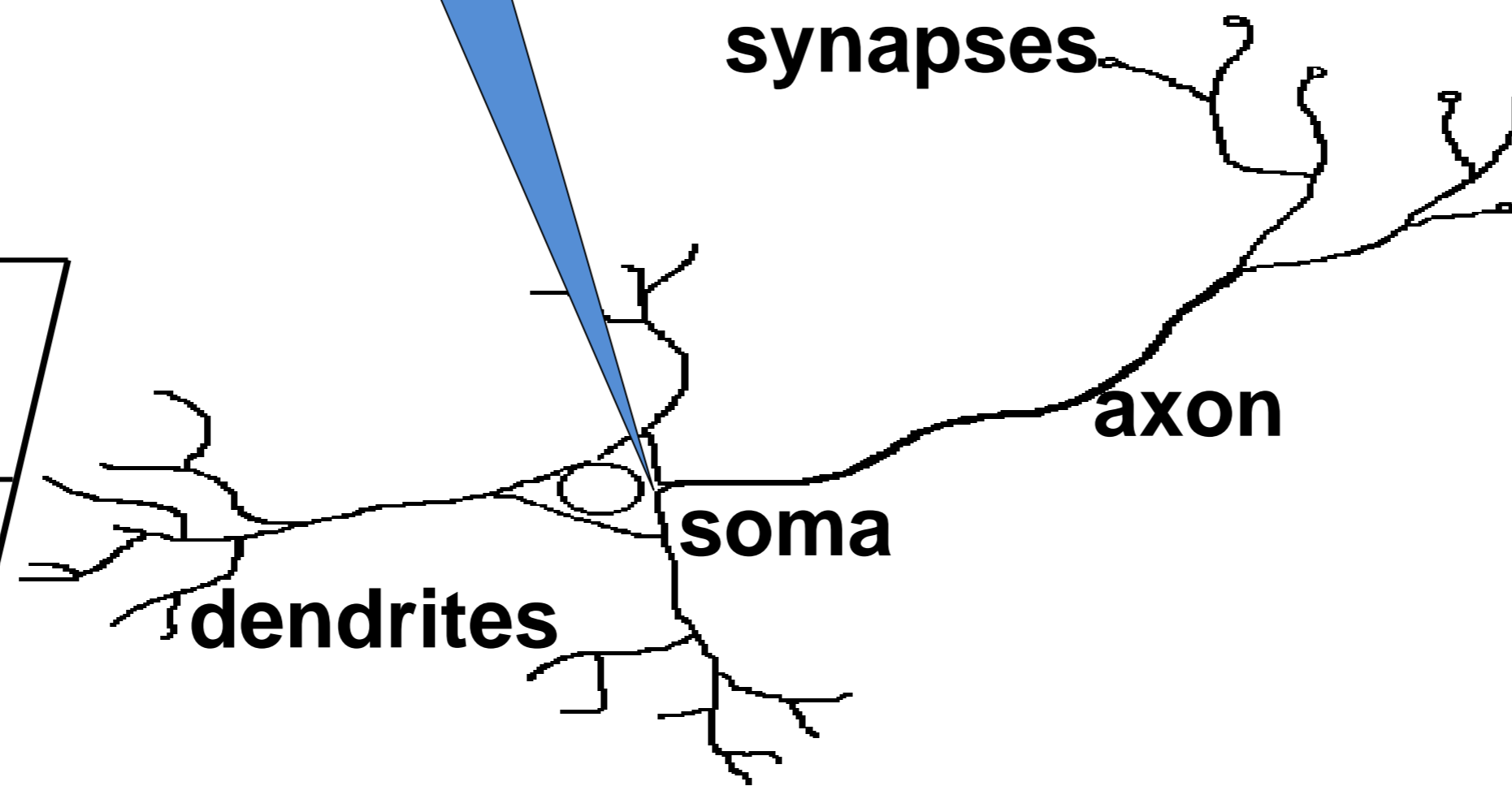


Place fields



electrode

synapses



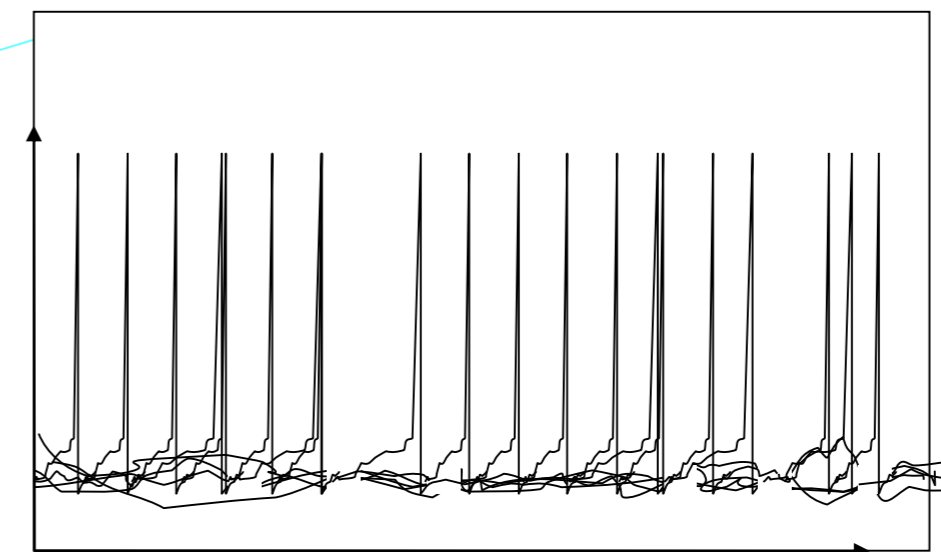
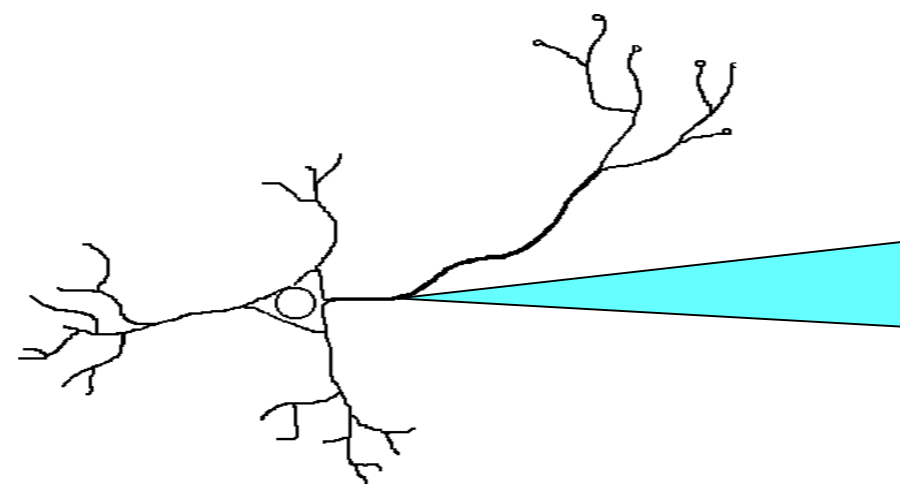
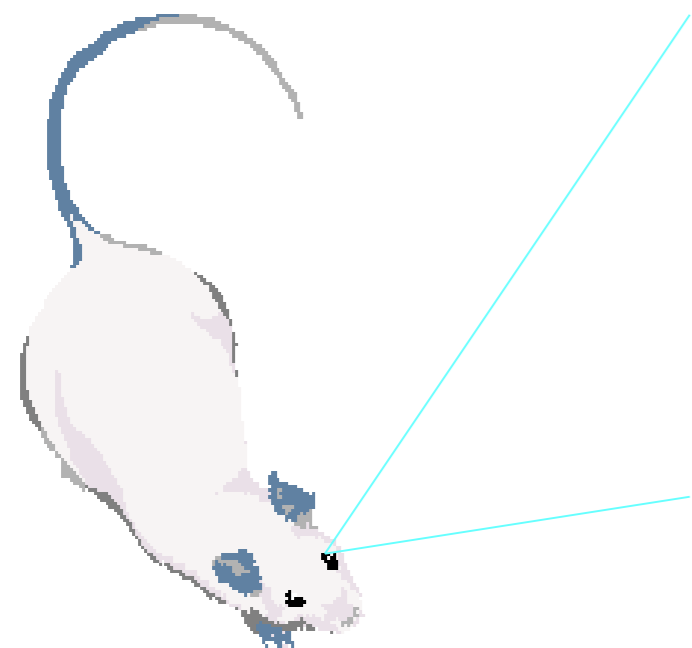
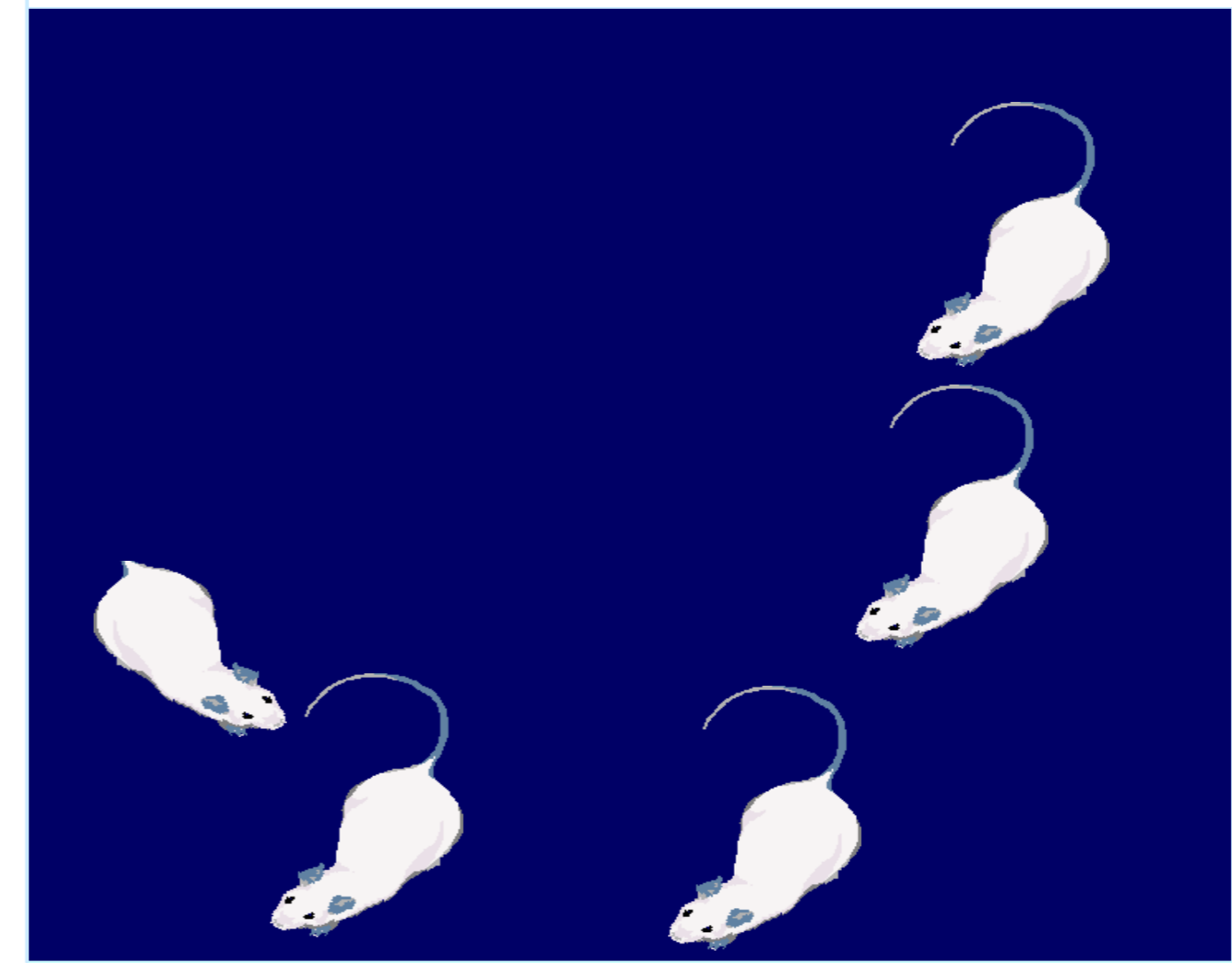
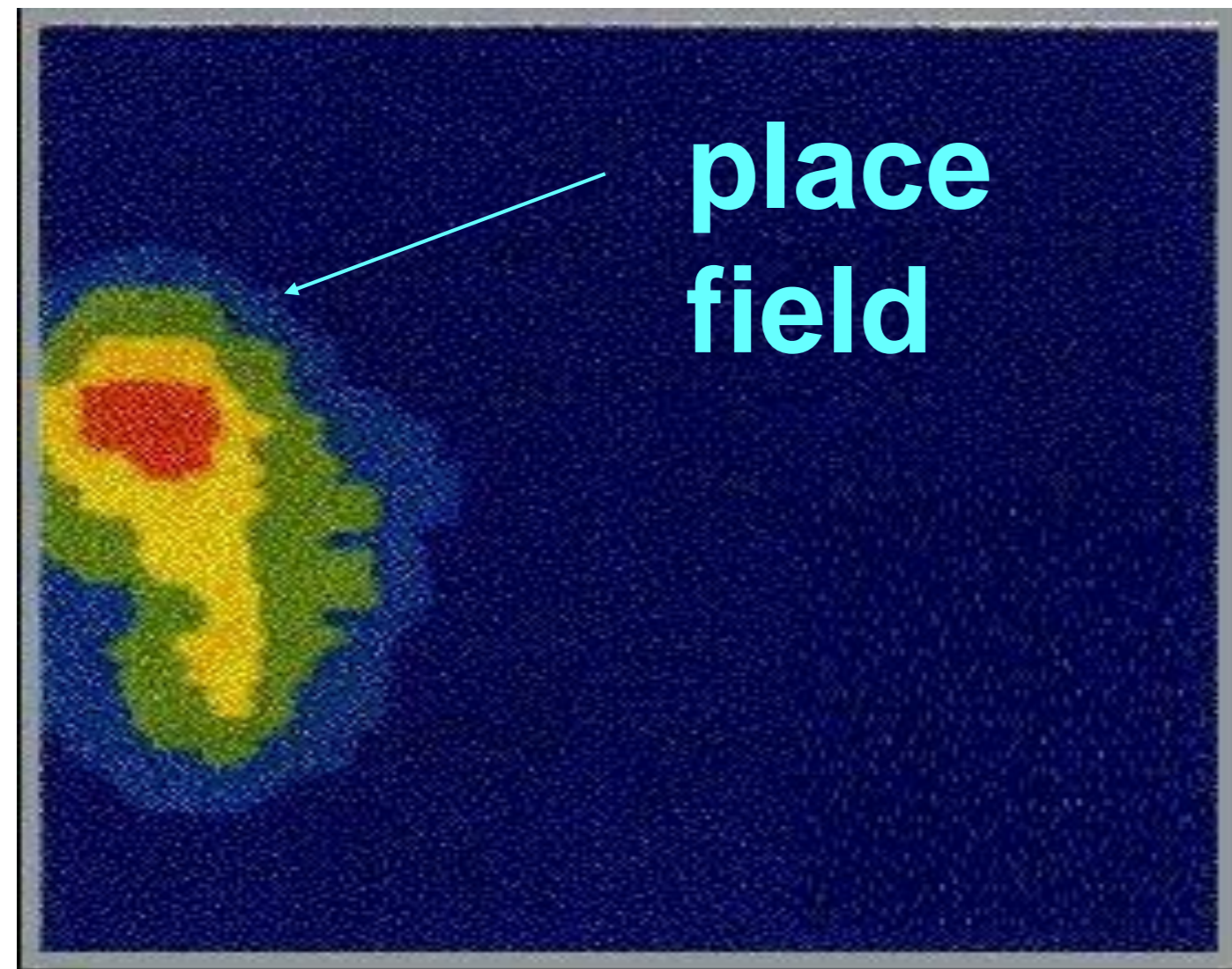
pyramidal cells

Previous slide.

the hippocampus of rodents (rats or mice) looks somewhat different to that of humans. Importantly, cells in hippocampus of rodents respond only in a small region of the environment. For this reason they are called place cells. The small region is called the place field of the cell.

6. Representation of states: Hippocampal place cells

Main property: encoding the animal's location



Previous slide.

Left: experimentally measured place field of a single cell in hippocampus.

Right: computer animation of place field

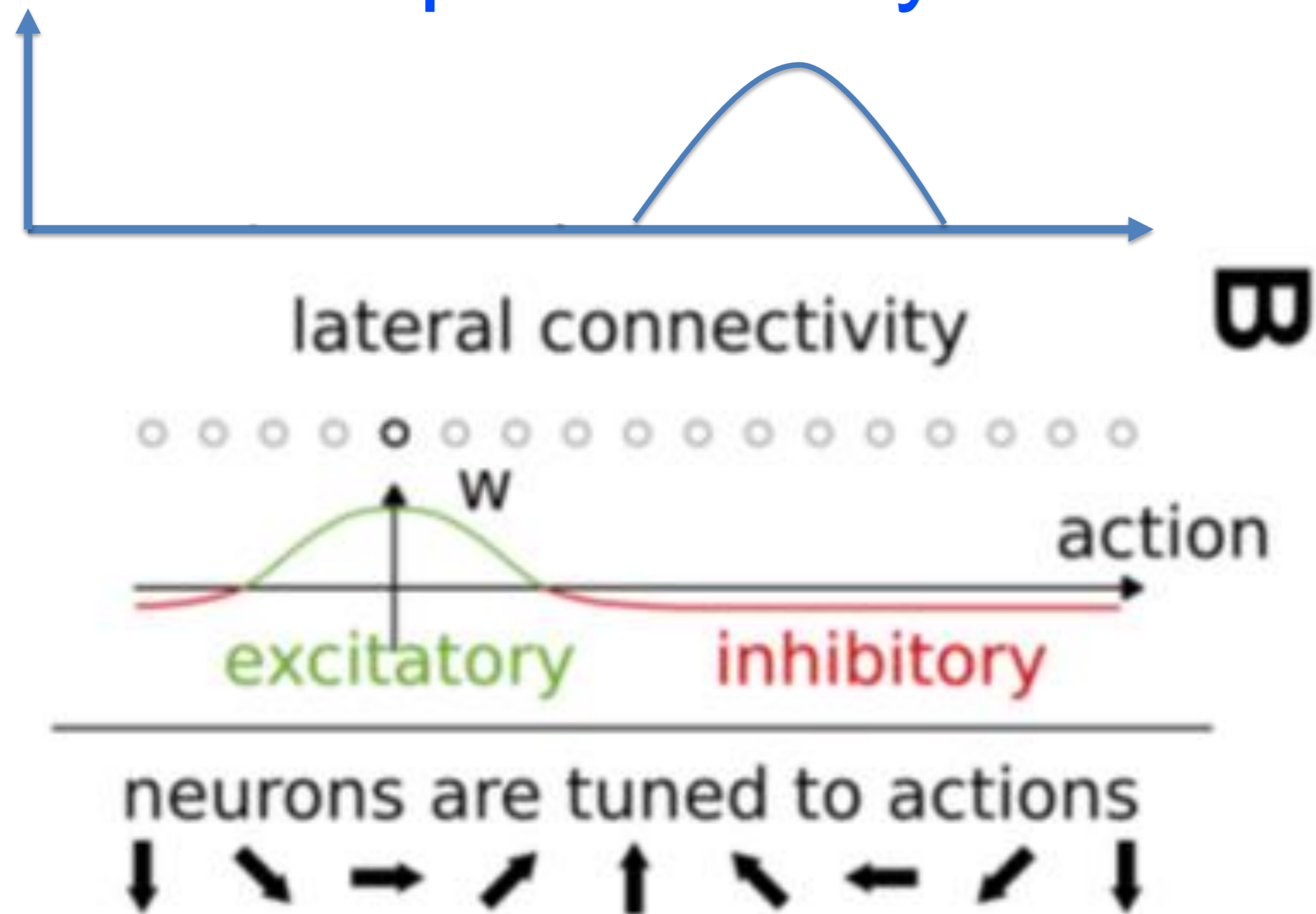
6. Representation of actions: Ring of spiking actor neurons

Mexican-hat interaction :

- Local excitation
- Long-range inhibition

$$w_{ij} = F(|i - j|)$$

→ Bump of activity at arbitrary location.



Previous slide.

Mexican-hat is a widely used interaction scheme.

Neighboring neurons (within some distance) excite each other, while far-away neurons inhibit each other. The connectivity pattern is translation invariant.

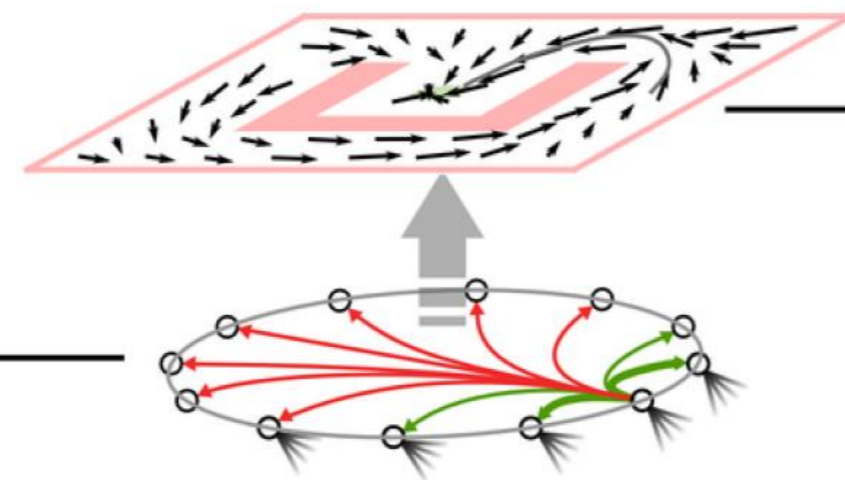
If the excitatory interactions are strong enough, then a localized group of neurons fires with high activity while all other neurons are inactivity. Importantly, the bump of activity can sit at an arbitrary location. The location can be influenced by input from a previous layer of neurons.

In our application, the location of the bump indicates the momentary action of movement.

6. Representation of actions: Ring of spiking actor neurons

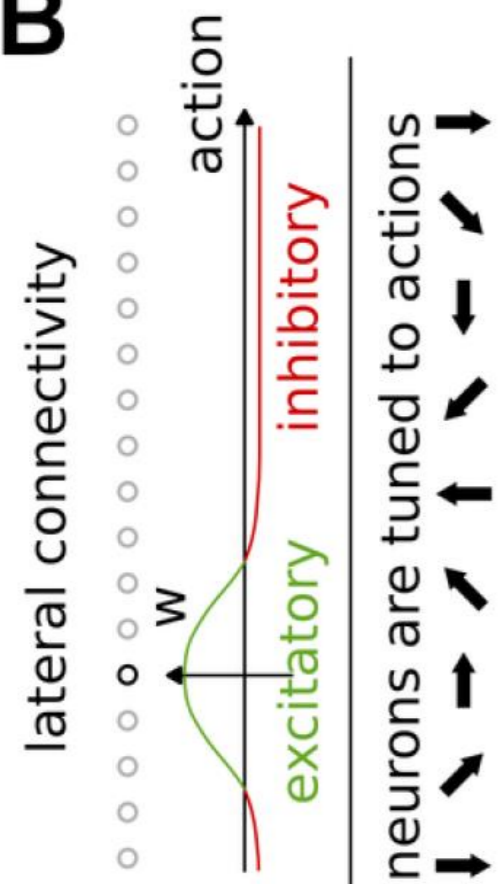
Note: no need to formally define a softmax function

A

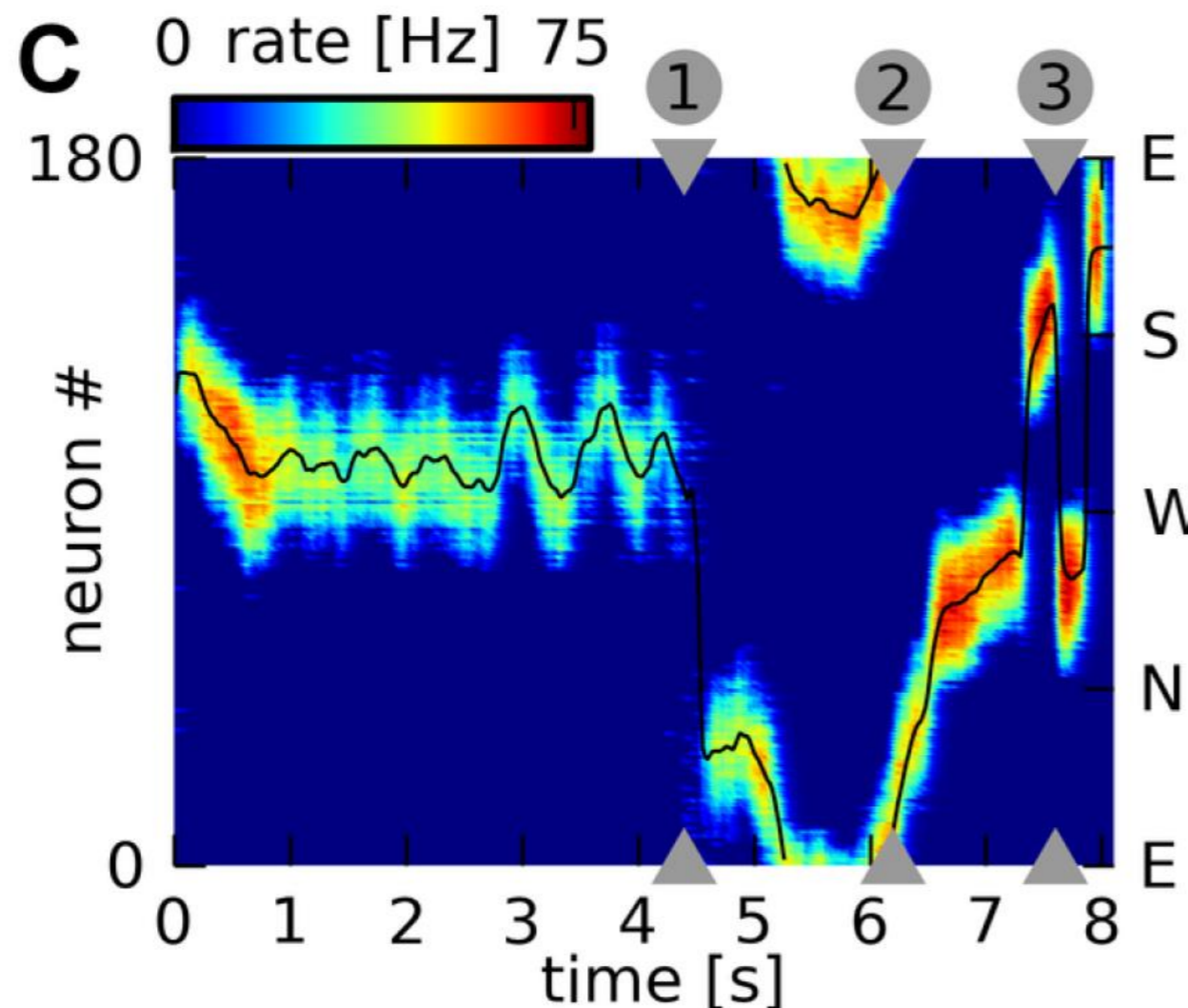


- Local excitation
- Long-range inhibition
- Not a formal softmax
- Could be a model of action selection in striatum

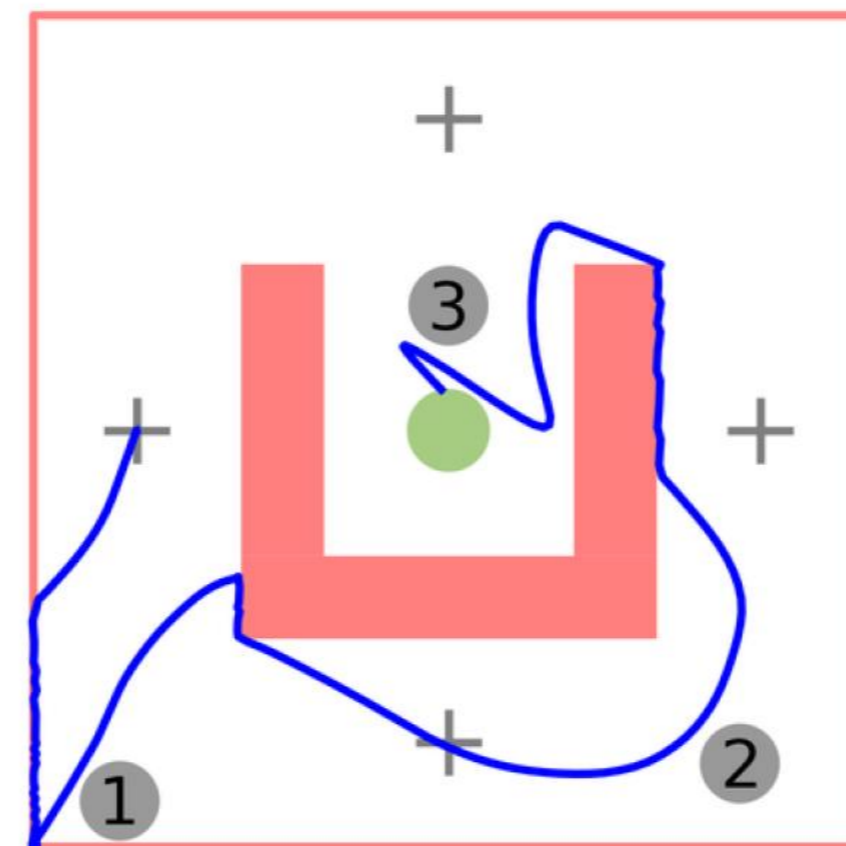
B



C



D



6. Ring of actor neurons

Actor neurons (previous slide).

A: A ring of actor neurons with lateral connectivity (bottom, green: excitatory, red: inhibitory) embodies the agent's policy (top).

B: Lateral connectivity. Each neuron codes for a distinct motion direction.

Neurons form excitatory synapses to similarly tuned neurons and inhibitory synapses to other neurons.

C: Activity of actor neurons during an example trial. The activity of the neurons (vertical axis) is shown as a color map against time (horizontal axis). The lateral connectivity ensures that there is a single bump of activity at every moment in time. The black line shows the direction of motion (right axis; arrows in panel B) chosen as a result of the neural activity.

D: Maze trajectory corresponding to the trial shown in C. The numbered position markers match the times marked in C.

6. Representation of Learning rule: Spikes + Eligibility trace

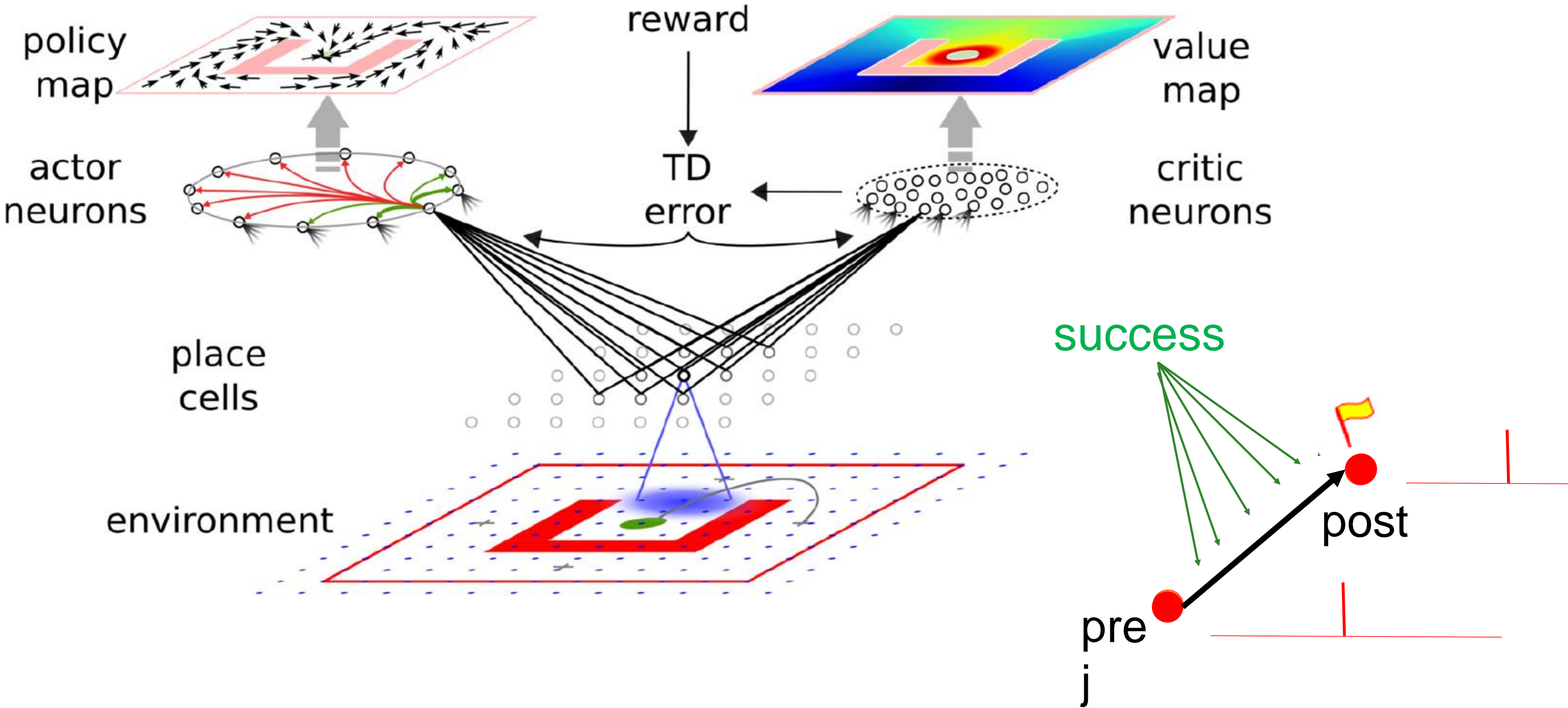


Figure 1. Navigation task and actor-critic network. From bottom to top: the simulated agent evolves in a maze environment, until it finds the reward area (green disk), avoiding obstacles (red). Place cells maintain a representation of the position of the agent through their tuning curves. Blue shadow: example tuning curve of one place cell (black); blue dots: tuning curves centers of other place cells. Right: a pool of critic neurons encode the expected future reward (value map, top right) at the agent's current position. The change in the predicted value is compared to the actual reward, leading to the temporal difference (TD) error. The TD error signal is broadcast to the synapses as part of the learning rule. Left: a ring of actor neurons with global inhibition and local excitation code for the direction taken by the agent. Their choices depending on the agent's position embody a policy map (top left).

doi:10.1371/journal.pcbi.1003024.g001

6. Learning rule: Three-factor STDP for reward-based learning

$$\frac{dw_{ij}}{dt} = F(w_{ij}; \text{PRE}_j, \text{POST}_i, 3rd)$$

pre-post-coincidence

$$\tau_z \frac{dz_{ij}}{dt} = -z_{ij} + STDP(t_i^f - t_j^f)$$

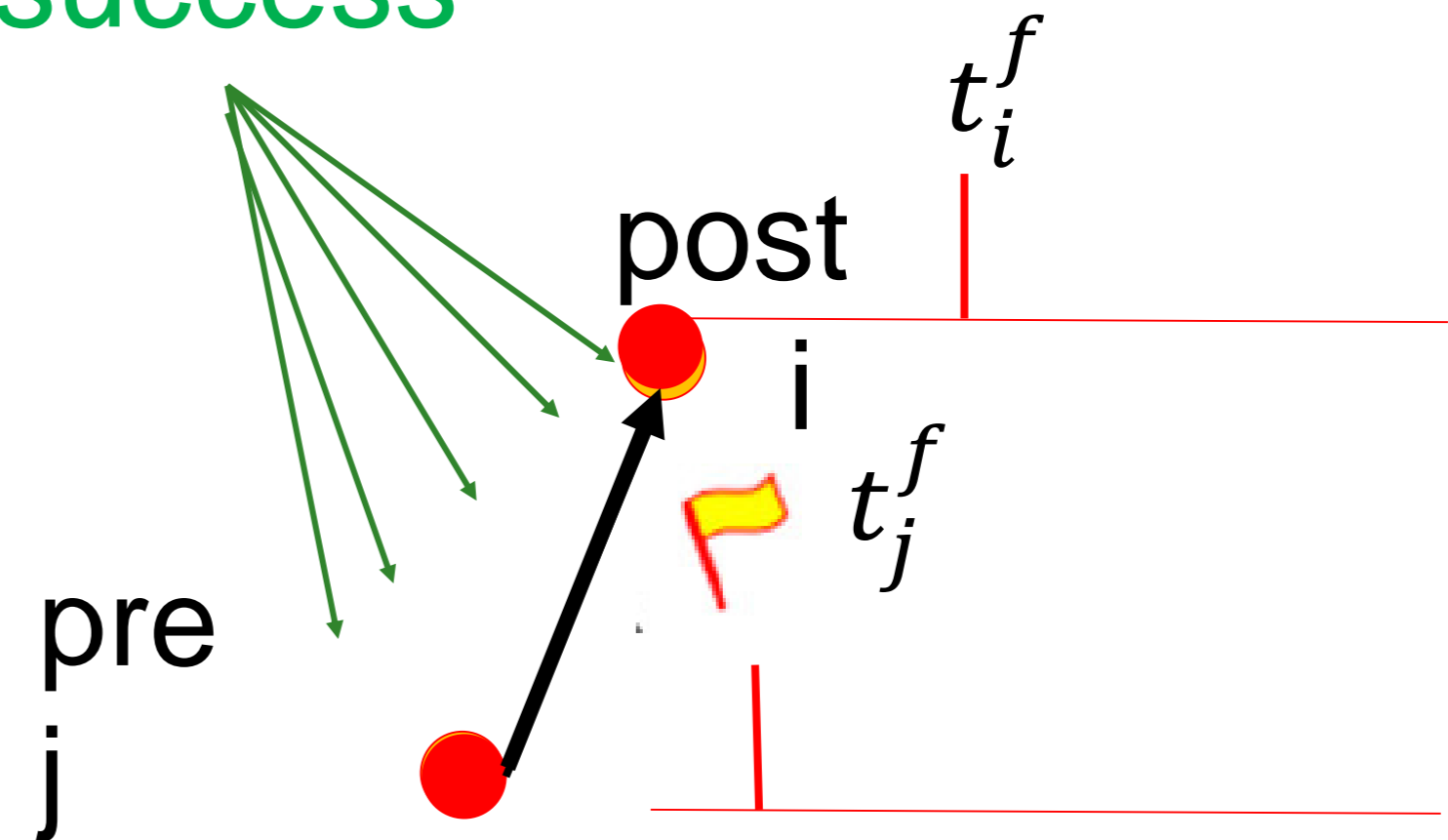
$$\frac{dw_{ij}}{dt} = z_{ij} \cdot S(t)$$

Success signal

Hebb rule/eligibility trace

Success signal:
TD error

success



1s

Xie and Seung 2003, Izhikevich, 2007; Florian, 2007; Legenstein et al., 2008, Fremaux et al. 2010, 2013

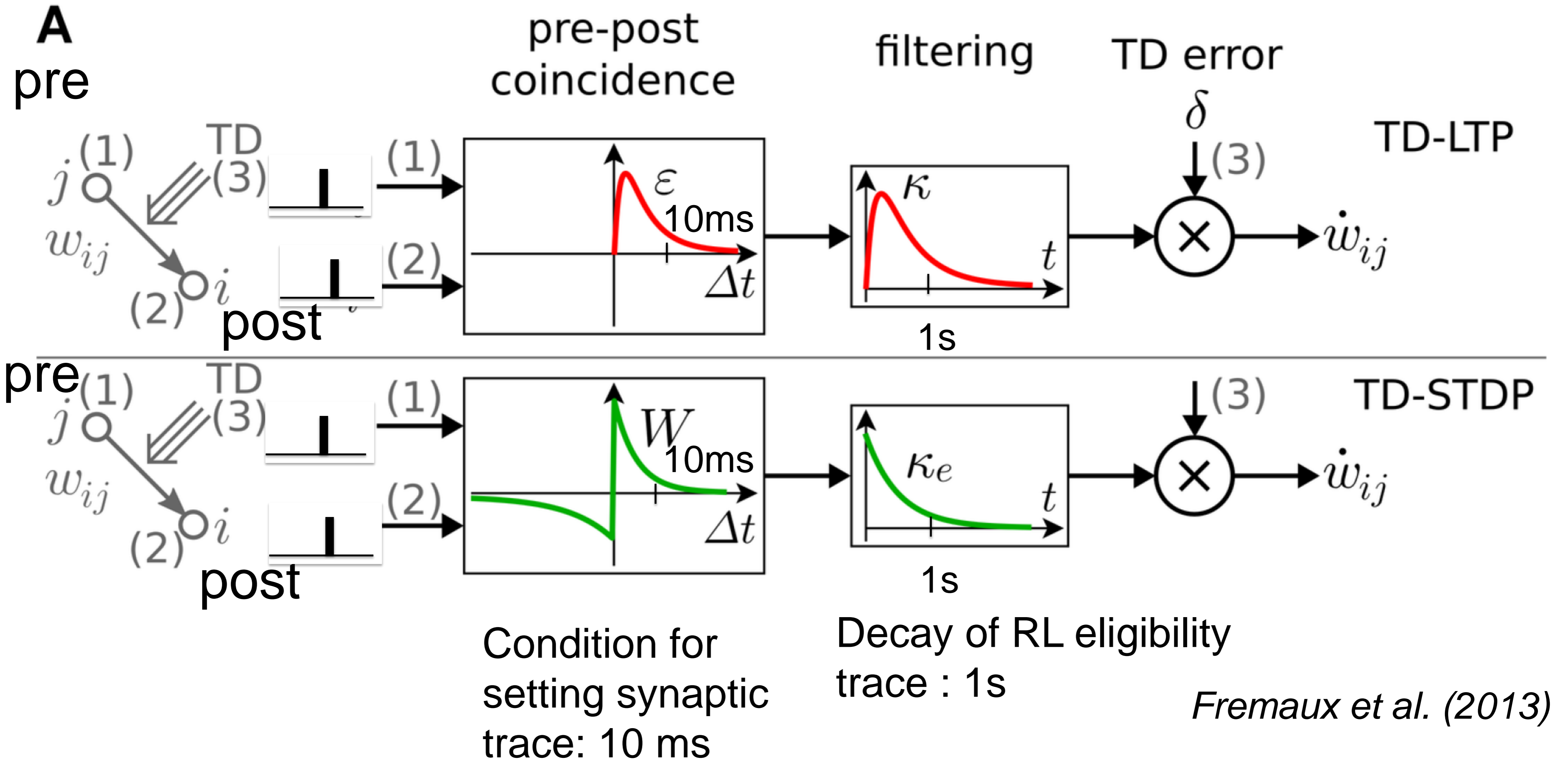
6. Learning rule with TD in Actor-Critic for spiking neurons

Learning rule with three factors (previous slide) based on spikes

1. In biology, neurons communicate by spikes (short electrical pulses).
2. Synaptic changes depend on the relative timing of the spikes of the sending (pre) and the receiving (post) neuron: Spike-Timing-Dependent Plasticity (STDP). Strong changes occur only if pre- and postsynaptic spikes coincide within ± 20 ms.
3. STDP is used to set the eligibility trace. The eligibility trace decays on a much slower time scale of 1s.
4. A success signal is necessary to transform the eligibility trace into an actual weight change.

Therefore weights increase if a success signal occurs within roughly one second after a coincident activity of pre- and postsynaptic neuron.

6. Two variants of spike-based three-factor Learning rules



Fremaux et al. (2013)

6. Learning rule with TD in Actor-Critic for spiking neurons

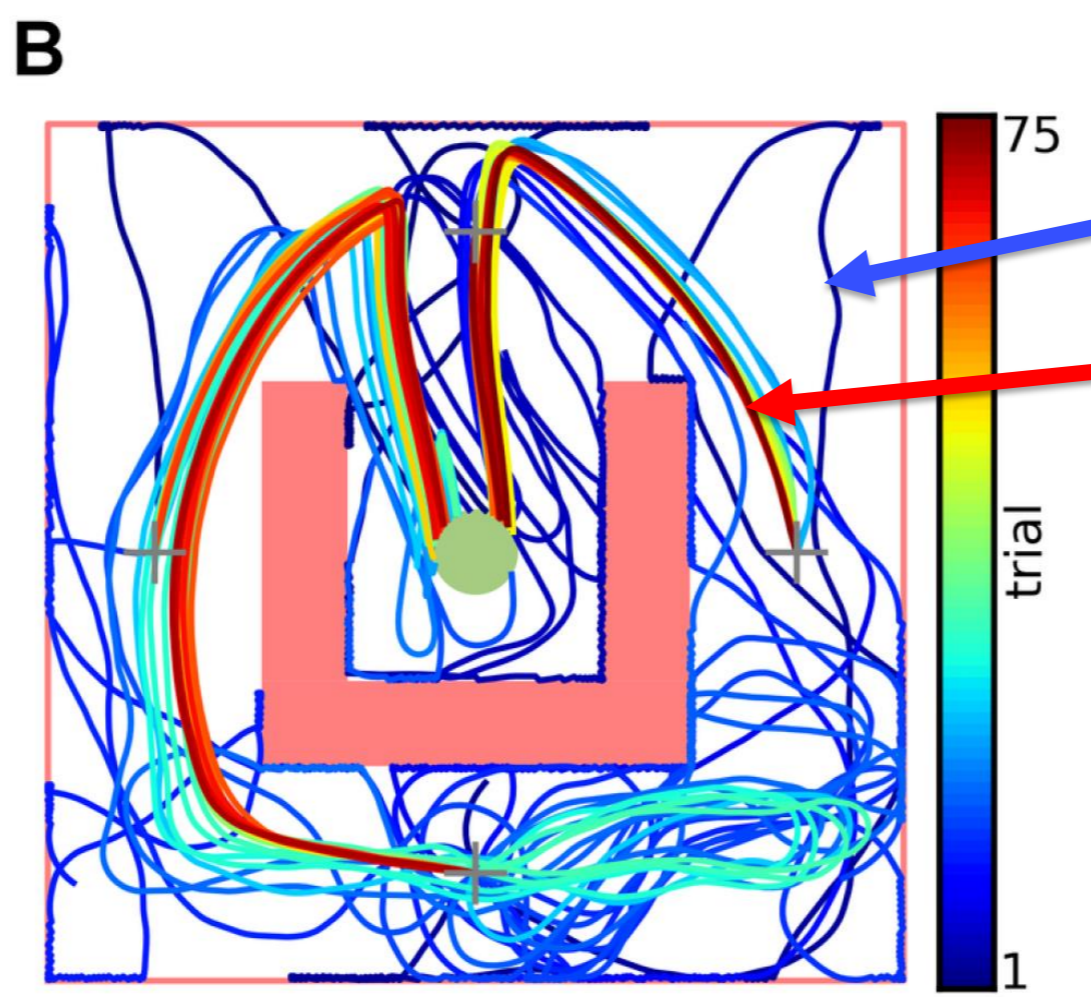
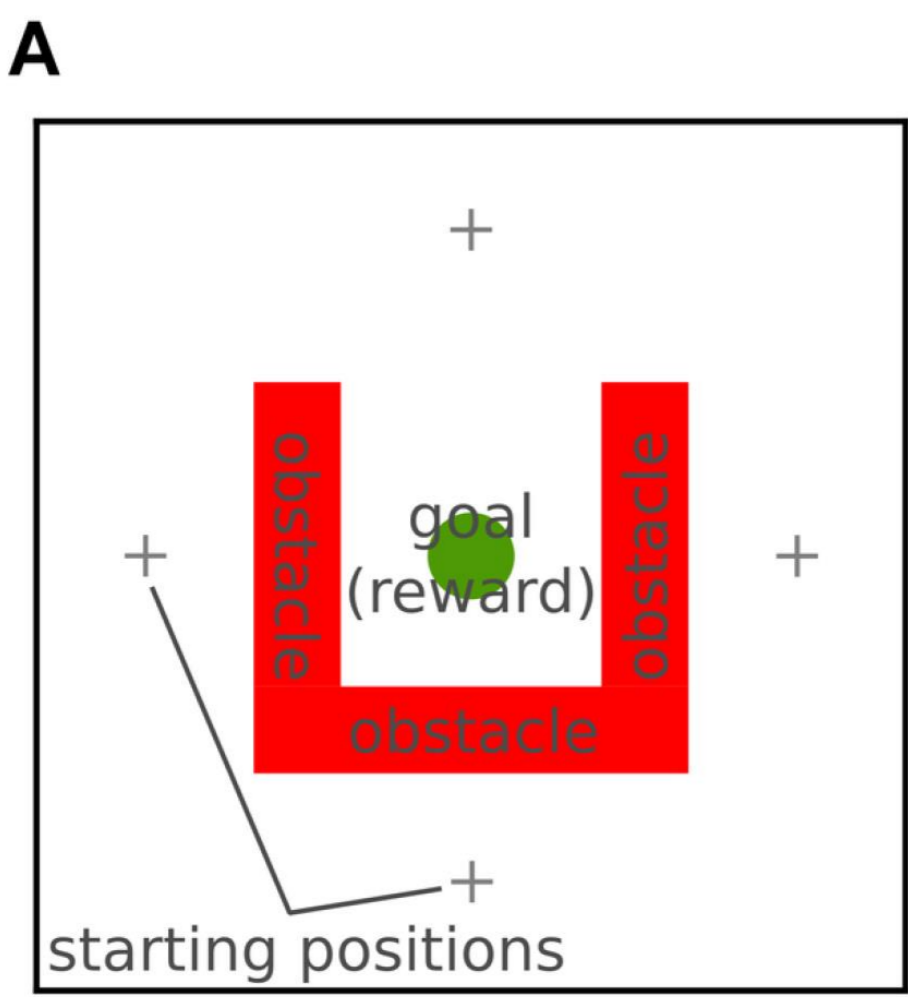
A: Learning rule with three factors (previous slide). We consider two different variants

Top: TD-LTP is the learning rule resulting from policy gradient. It works by passing the presynaptic spike train (factor 1) and the postsynaptic spike train (factor 2) through a coincidence window ε . Spikes are counted as coincident if the postsynaptic spike occurs within after a few ms of a presynaptic spike. The result of the pre-post coincidence measure is low-pass-filtered by passing it through a kernel (which yields the eligibility trace, decaying of 1s), and then multiplied by the TD error $\delta(t)$ (factor 3) to yield the learning rule which controls the change of the synaptic weight w_{ij} .

Bottom: TD-STDP is closer to biology and consists of a TD-modulated variant of STDP. The main difference with TD-LTP is the presence of a post-before-pre component in the coincidence window. As before, coincidences with 10ms set the eligibility trace

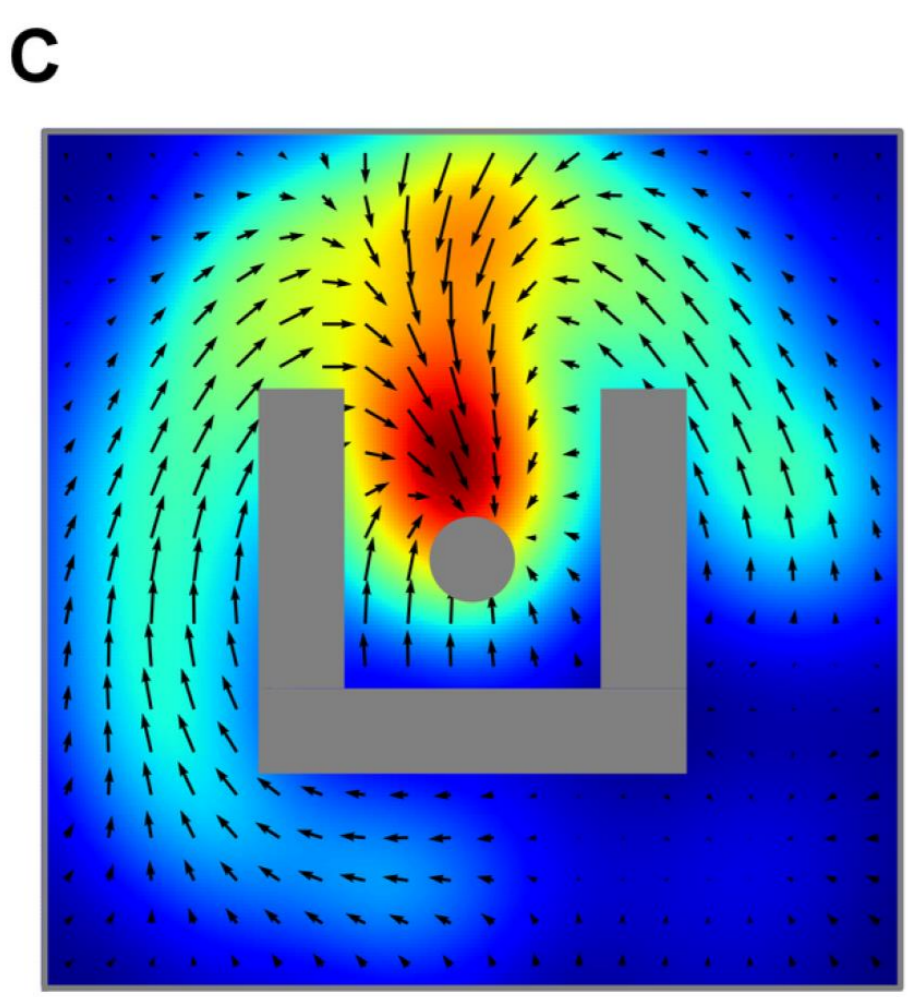
Fremaux et al. (2013)

6. Maze Navigation with TD in Actor-Critic with spiking neurons

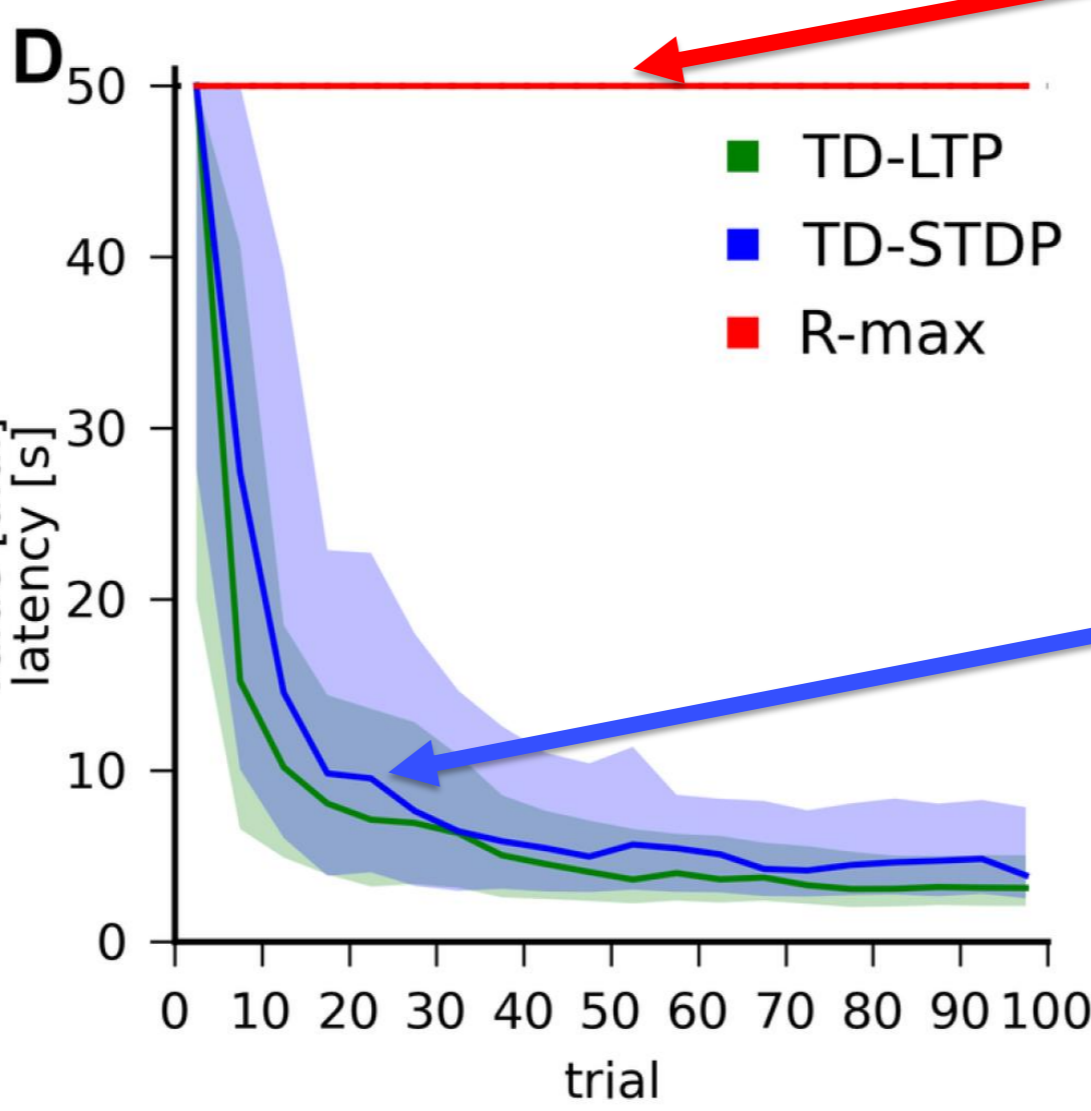


early trial

Late trial



value map



R-max:

Policy gradient without the critic. The goal was never found within 50s.

TD-STDP:

After 25 trials, the goal was found within 20s.

6. Maze Navigation with TD in Actor-Critic with spiking neurons

Maze navigation learning task. Both TD rules (TD-LTP and TD-STDP) work equally well. Hence, details of how the eligibility trace is set do not matter.

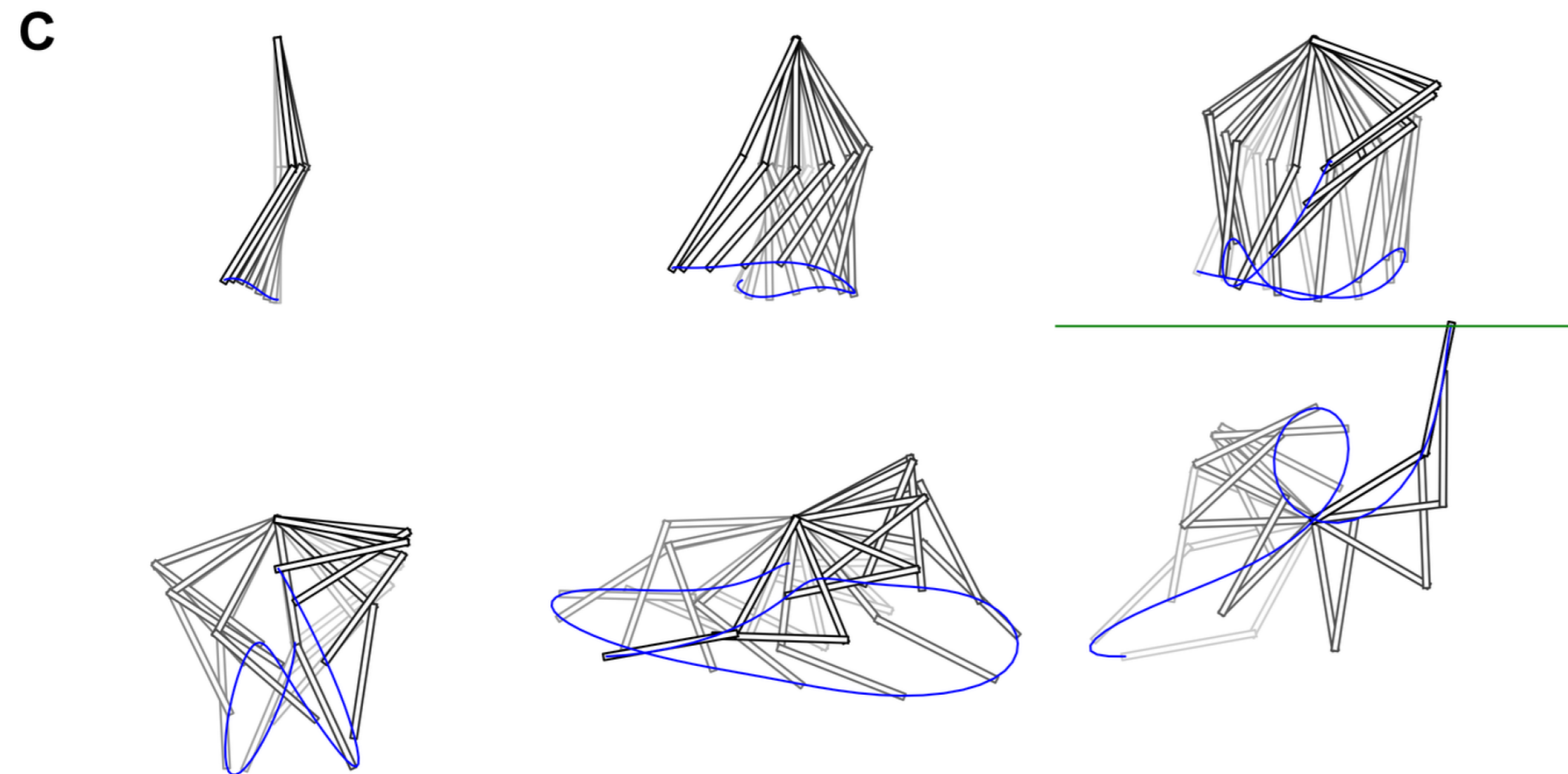
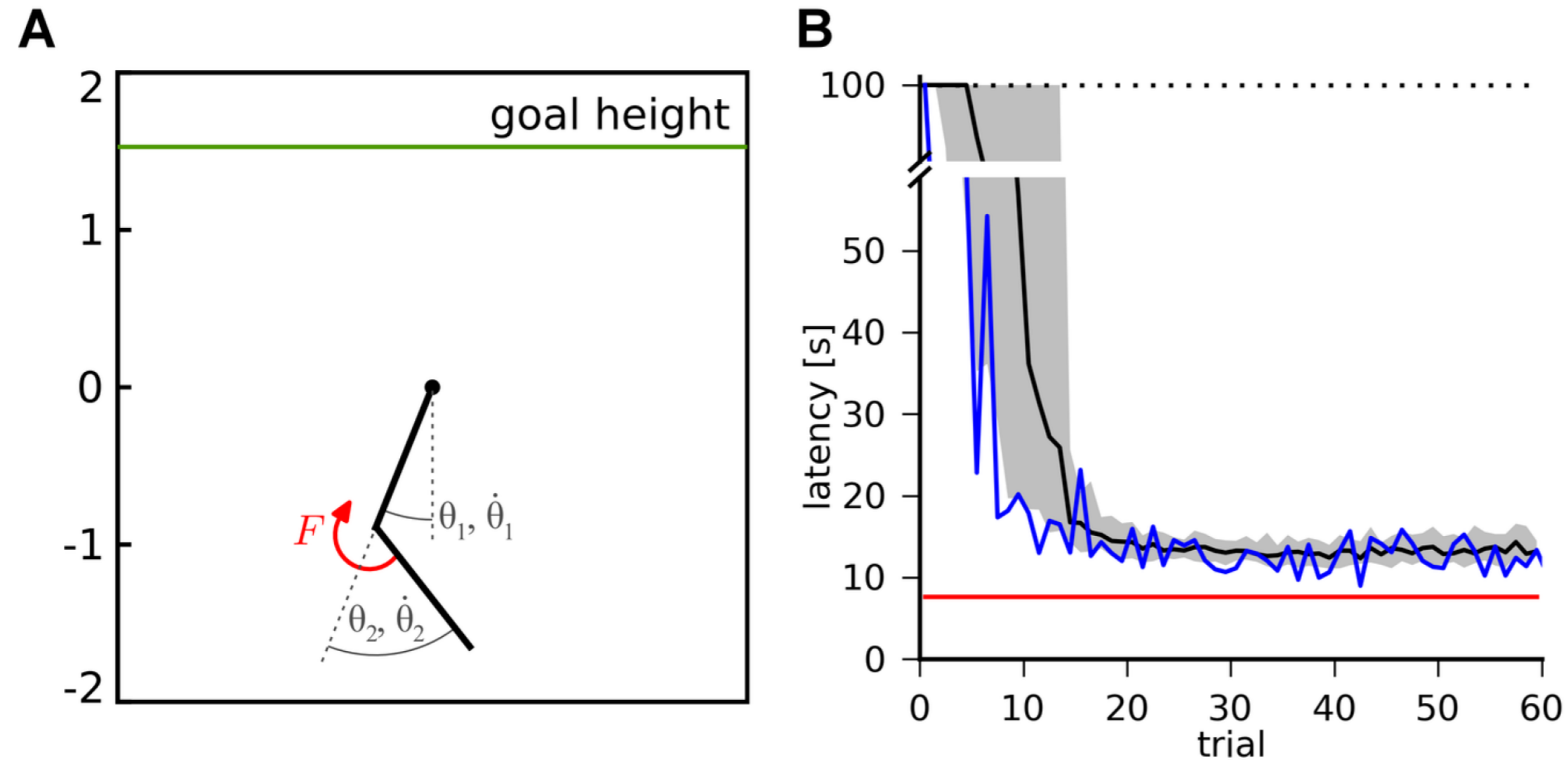
A: The maze consists of a square enclosure, with a circular goal area (green) in the center. A U-shaped obstacle (red) makes the task harder by forcing turns on trajectories from three out of the four possible starting locations (crosses).

B: Color-coded trajectories of an example TD-LTP agent during the first 75 simulated trials. Early trials (blue) are spent exploring the maze and the obstacles, while later trials (green to red) exploit stereotypical behavior.

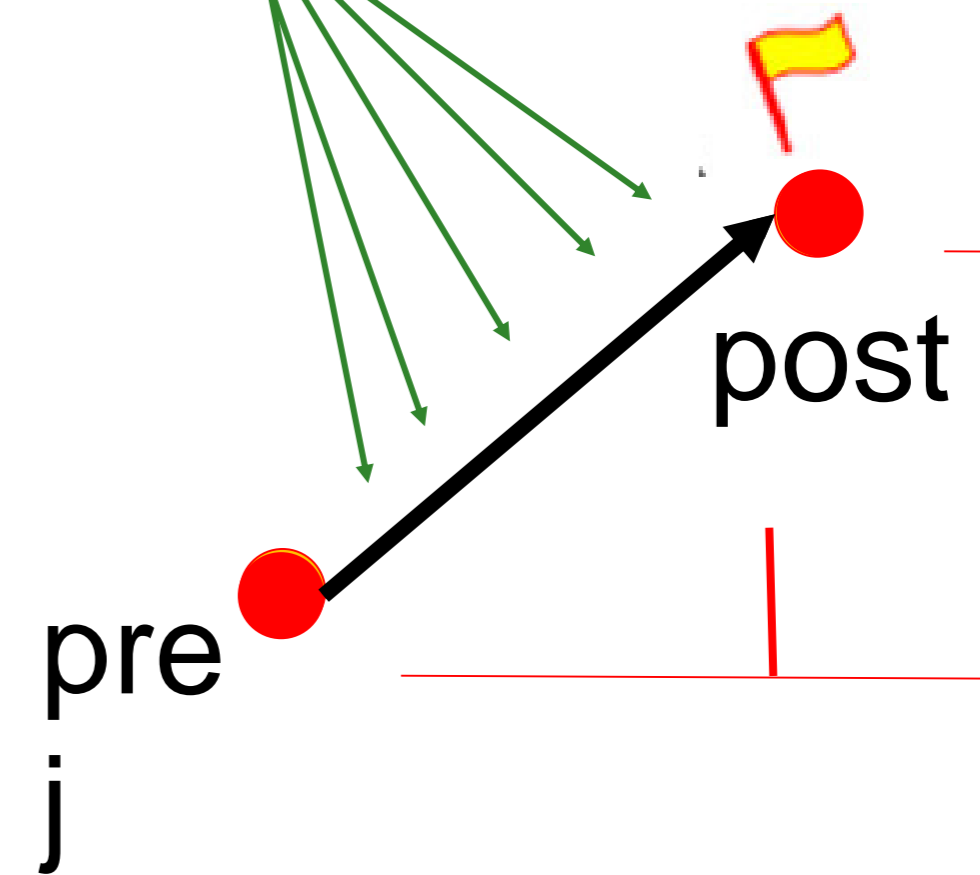
C: Value map (color map) and policy (vector field) represented by the synaptic weights of the agent of panel B after 2000s simulated seconds.

D: Goal reaching latency of agents using different learning rules. Latencies of N=100 simulated agents per learning rule. The solid lines shows the median shaded area represents the 25th to 75th percentiles. The R-max learning rule is standard policy gradient agent without a critic and enters times-out after 50 seconds. Hence it is important that the 3rd factor is TD and not just 'raw' reward.

6. Acrobot task with TD in Actor-Critic with spiking neurons



success



Previous slide.

Application of the same model (spiking three-factor rule) to the Acrobot task.

6. TD in Actor-Critic with spiking neurons

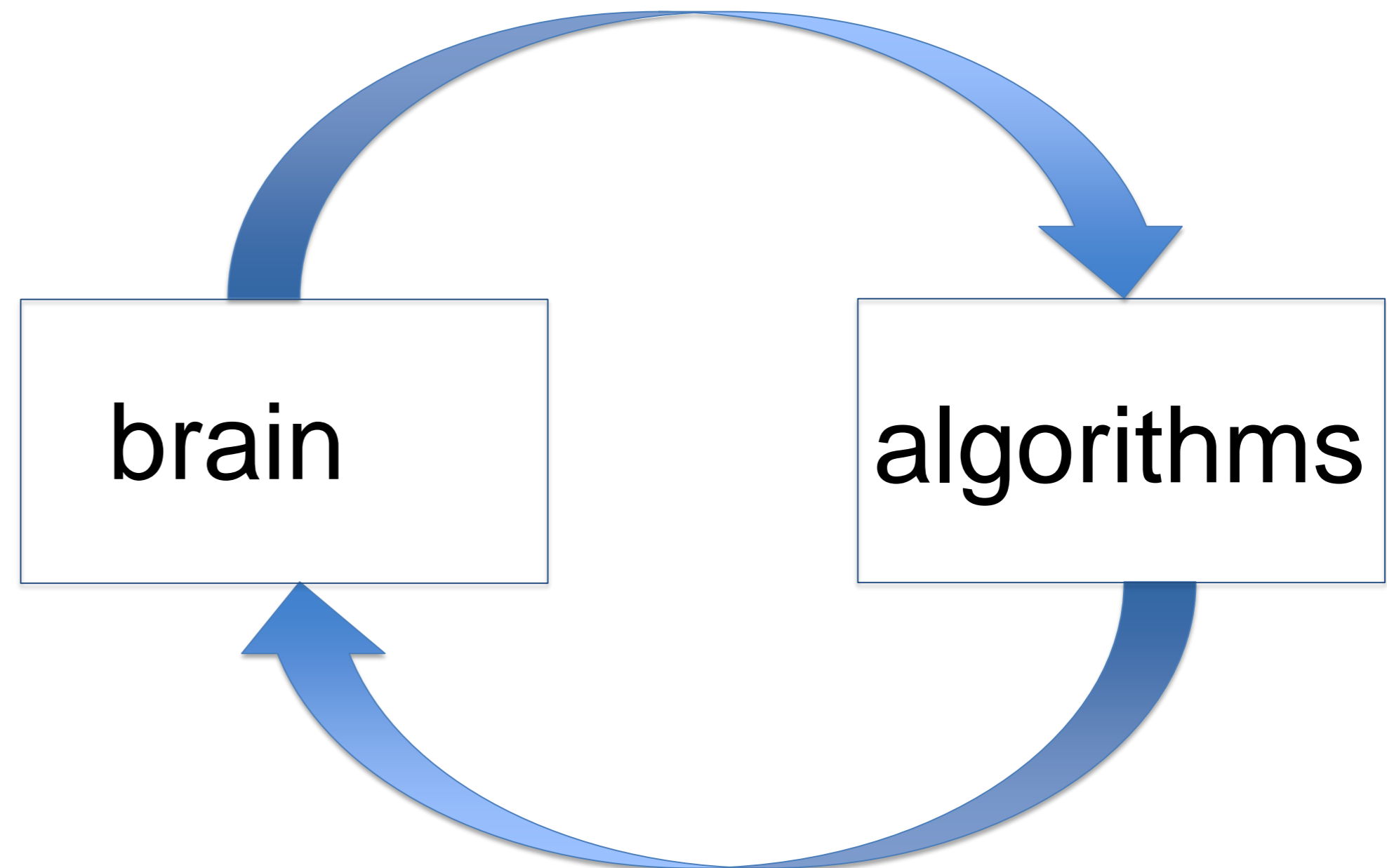
- Learns in a few trials (assuming good representation)
- Works in continuous time. No artificial 'time steps'
- Works with spiking neurons
- Works in continuous space and for continuous actions
- Uses a biologically plausible 3-factor learning rule
- Details of coincidence condition (STDP/spike LTP) irrelevant
- Critic implements value function
- TD signal calculated by critic and broadcasted to network
- Actor neurons interact via synaptic connections
- No need for algorithmic 'softmax'
- 3-factor rules with TD as global signal work much better than standard policy gradient (REINFORCE)

Previous slide.
Summary of findings

6. Summary Learning in the Brain vs RL algo

Advantage Actor-Critic Reinforcement learning needs:

- states / sensory representation
- action selection
- value function/critic
- broadcast of TD error
- TD error calculation



6. Summary

Several aspects of TD learning in an actor-critic framework can be mapped to the brain:

Sensory representation: Hippocampal place cells
(and Cortex)

Actor : Dorsal Striatum

Critic : Ventral Striatum (nucleus accumbens)

TD-signal: Dopamine

Three-factor rule!

With spiking neurons!

Learning in about 10 epochs.

6. Summary

Several aspects of TD learning in an actor-critic framework can be mapped to the brain:

State representation: Hippocampus (and Cortex)

Actor : Dorsal Striatum

Critic : Ventral Striatum (nucleus accumbens)

TD-signal: Dopamine

→ But how can we learn the ‘state representation’ by ‘place cells/radial basis functions’?

REVIEW from Lecture RL3: Continuous space in RL:

Self-localization and Navigation to Goal

- 2-dimensional arena 80cmx60cm
- single goal location
- 120 actions (=directions of movement)

Agent:

Khepera Robot

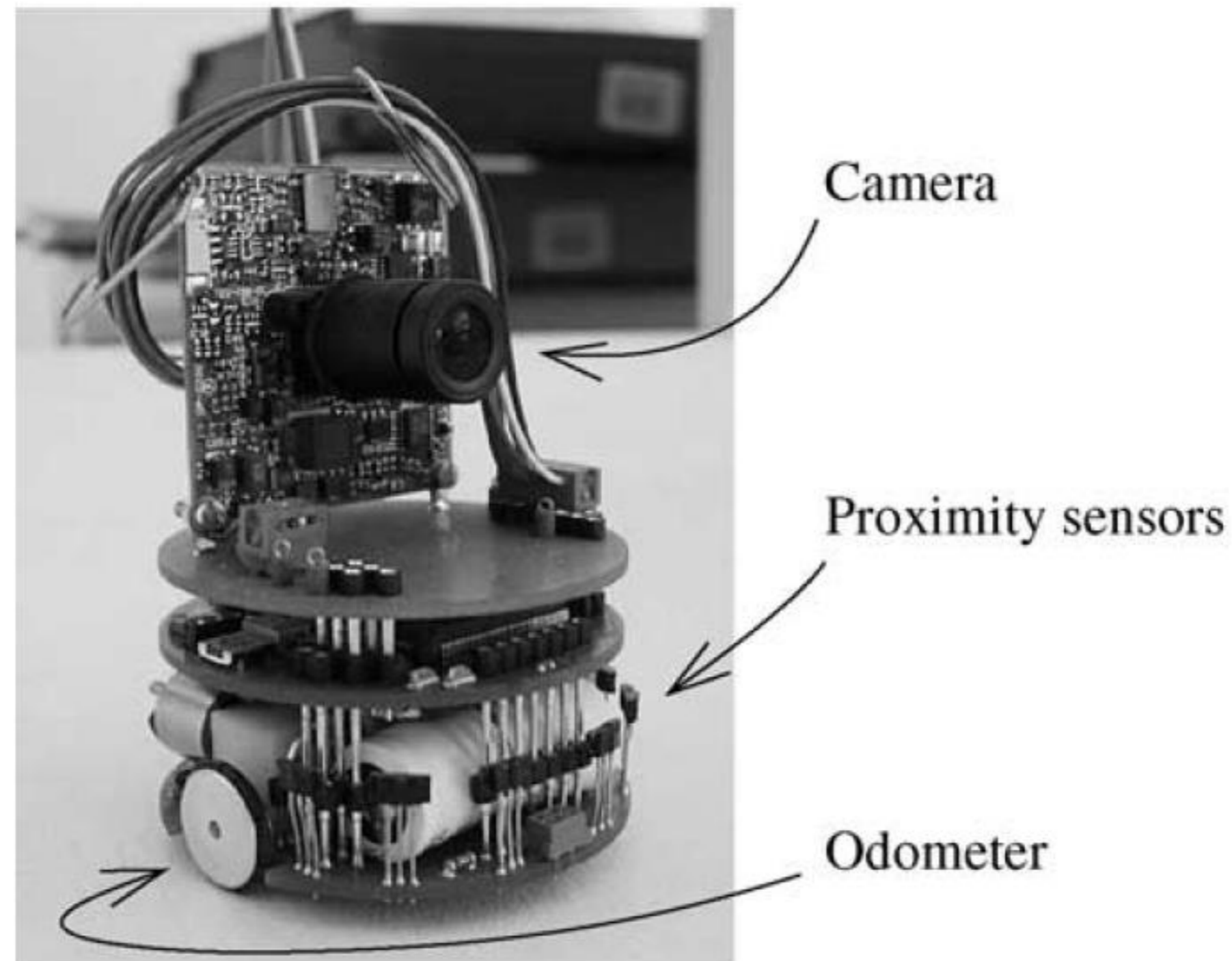
Camera:

240° view

>240 000 pixel

Preprocessing:

Gabor filter bank

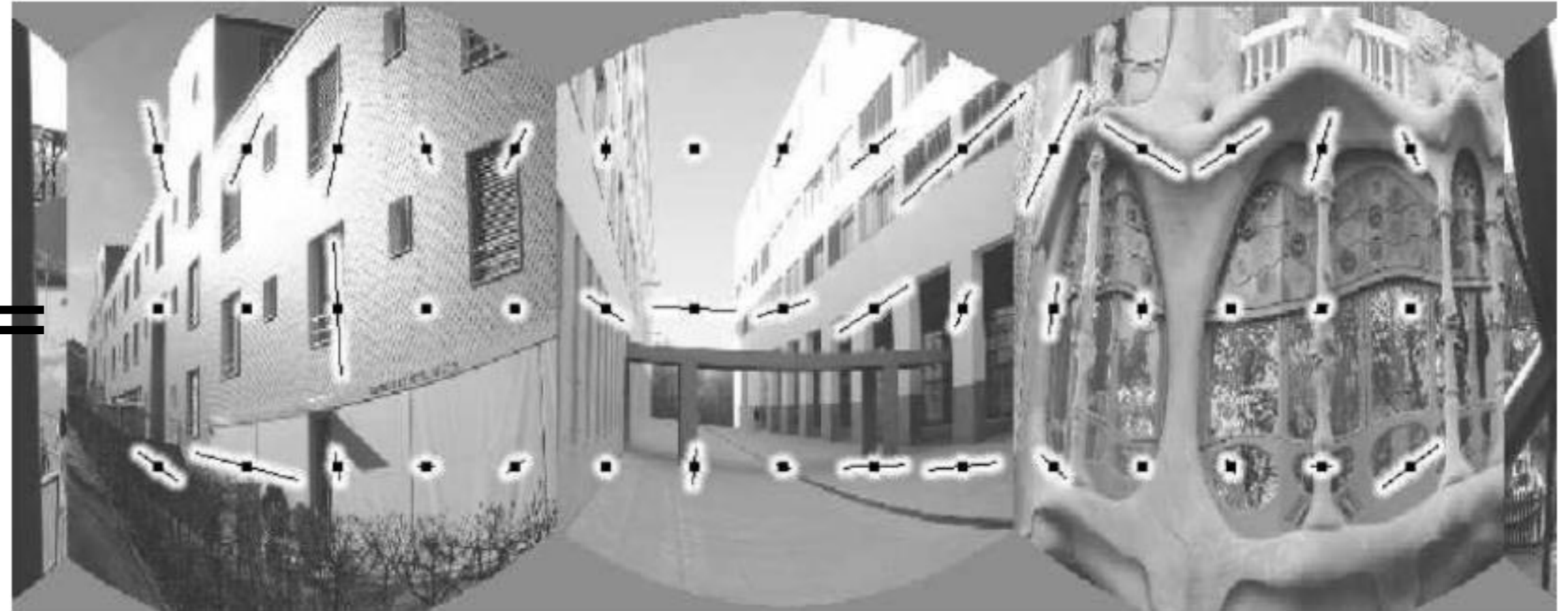


Previous slide.

The camera of the Khepera robot takes snapshots in 4 directions that are combined into a single view covering a viewing field of 240 degree (total would be 360 degree).

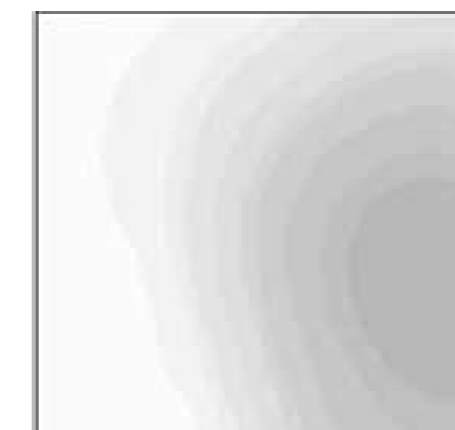
REVIEW from Lecture RL3: Self-localization

- **Preprocessing Gabor filter bank:**
Filters of several spatial frequency and orientation at 45 different locations.



- **Snap-shot of environment =**
if 'novel' store the vector F_j
of filter responses

- **'Basis-function'** $\phi(F(t) - F_j)$
similarity of current view $F(t)$ with stored view vector F_j
after rotation to optimal matching angle



*sample
basis function*

Previous slide.

The sample image shows the orientation of the most strongly responding filter with the lowest spatial frequency at the 45 sampling locations.

The Gabor filters come as pairs of sine and cosine filters (or complex filters) and only the total amplitude, but not the phase of the response of the filter pair is recorded.

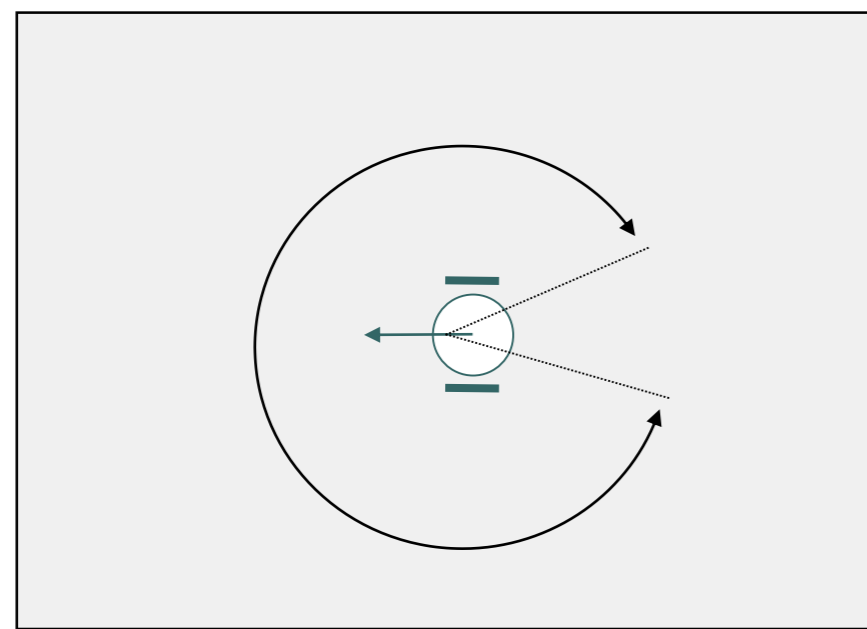
The set of filter responses at time t of all 9000 filters is denoted by $F(t)$

Details of the processing steps are explained in the next few slides

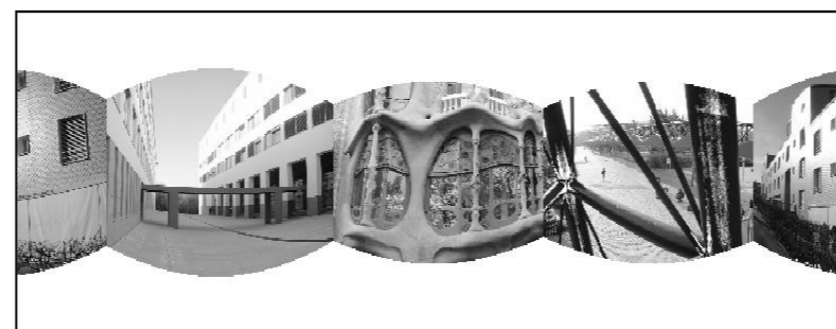
REVIEW from Lecture RL3: Self-localization

If 'novel': store views of visited places

Robot in an environment

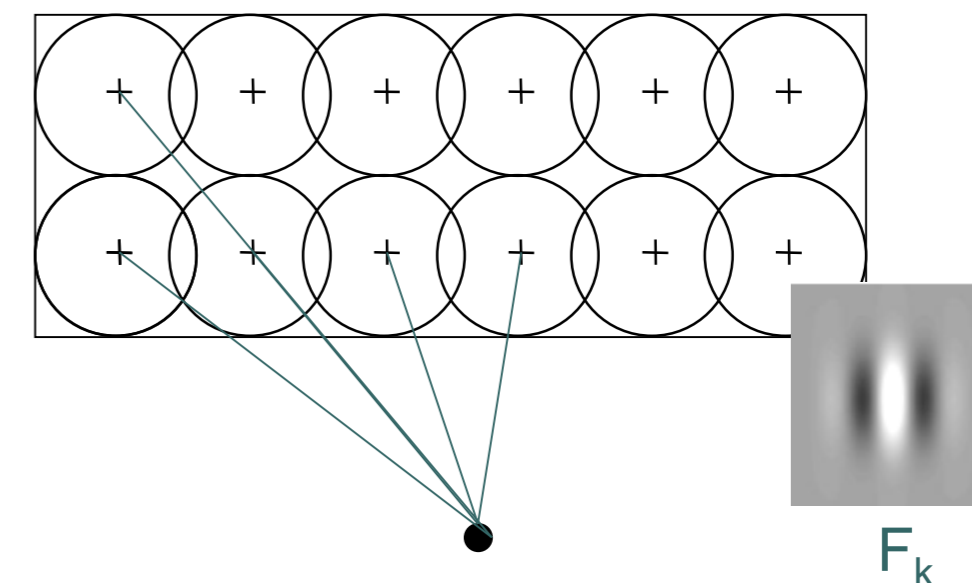


Visual input at each time step



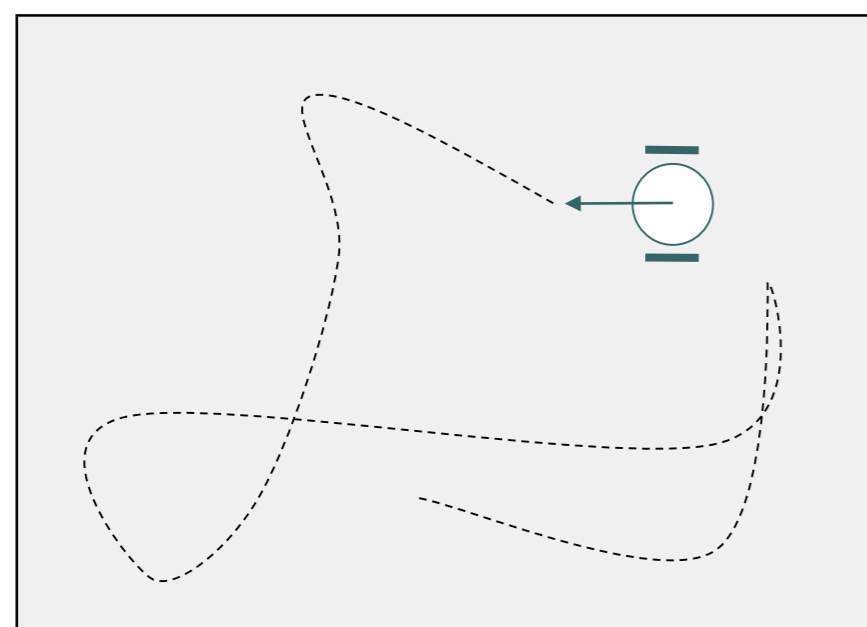
Local view : activation of set of 9000 Gabor wavelets

$$I(\vec{p}, \Phi) \Rightarrow \{F_k\}$$

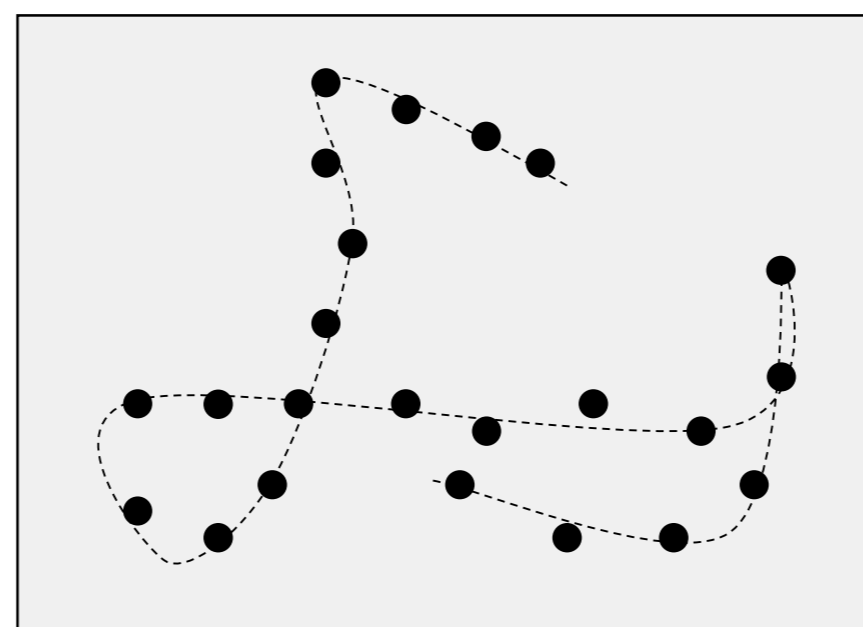


Single View Cell stores a local view

Environment exploration



Population of view cells



All local views are stored in an **incrementally growing** view cell population

Previous slide.

1) During exploration the robot takes a new sample image whenever it does not recognize the view. Recognition is defined that 10 or more cells strongly respond to the new image.

2) The sample image is memorized by storing the set of responses of the 9000 Gabor filters.

Next slide:

If we carefully analyze the algorithms described in steps 1) and 2) above, it can be implemented by Hebbian learning, modulated by a novelty signal.

Hence:

The learning rule is a three-factor rule, where the 3rd factor is 'novelty' as opposed to reward or TD. And indeed, there are neuromodulators that broadcast signals triggered by novelty.

6. Summary

State representation: Hippocampus (and Cortex)

Question: how can we learn the 'state representation' by 'place cells/radial basis functions'?

→ Growing population of view cells
= 2-factor rule, modulated by 'novelty'
= **another 3-factor rule**

Review: Neuromodulators

- 4 or 5 neuromodulators
- near-global action

Dopamine/reward/TD:
Schultz et al., 1997,
Schultz, 2002

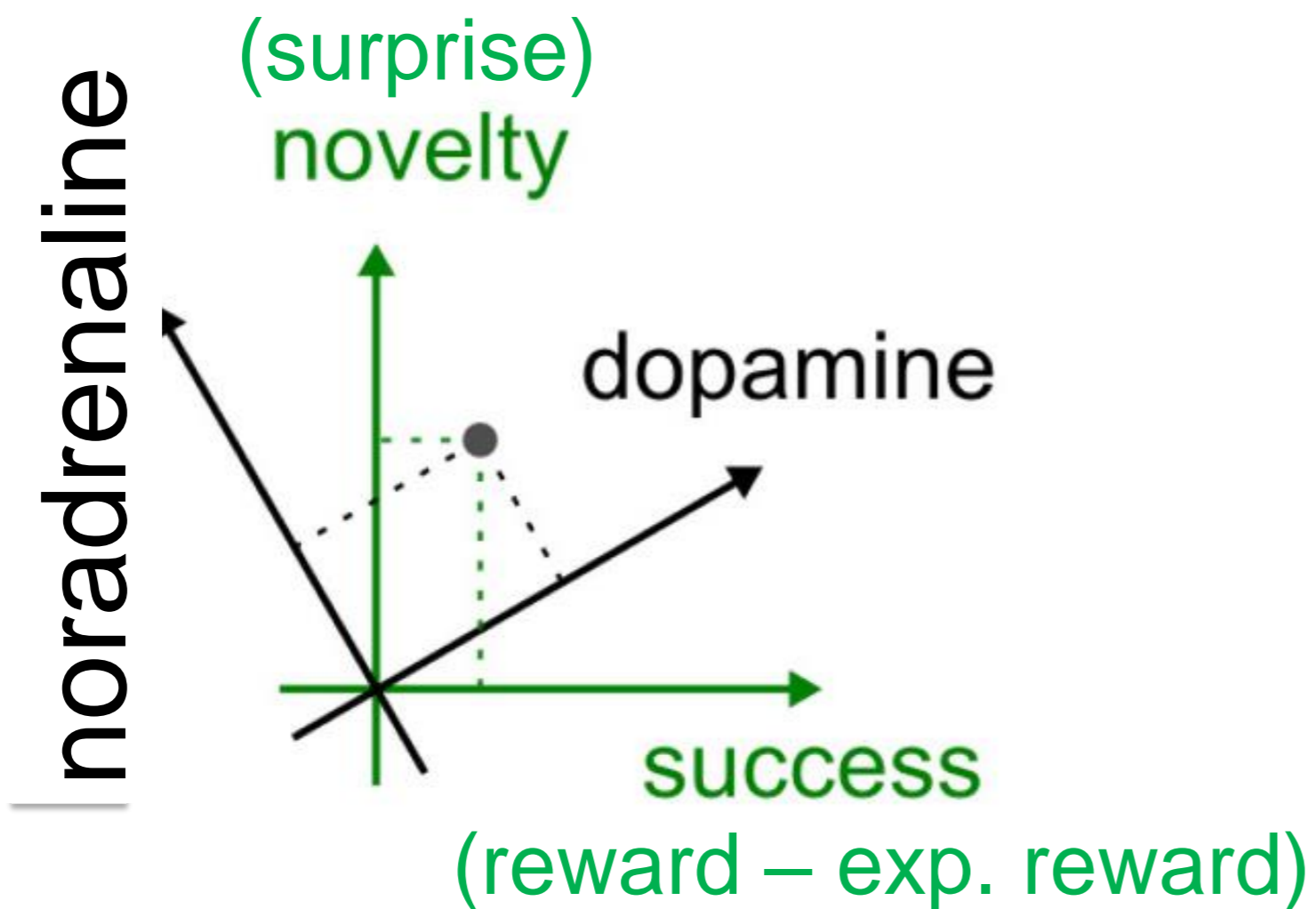
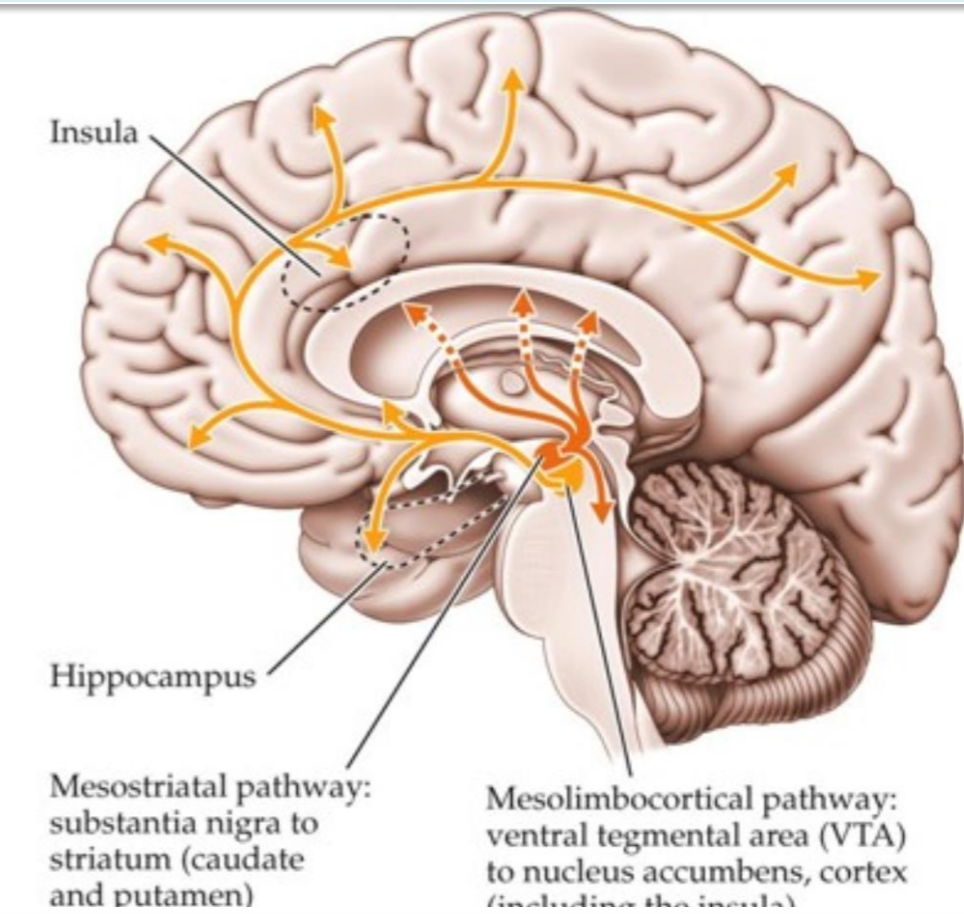


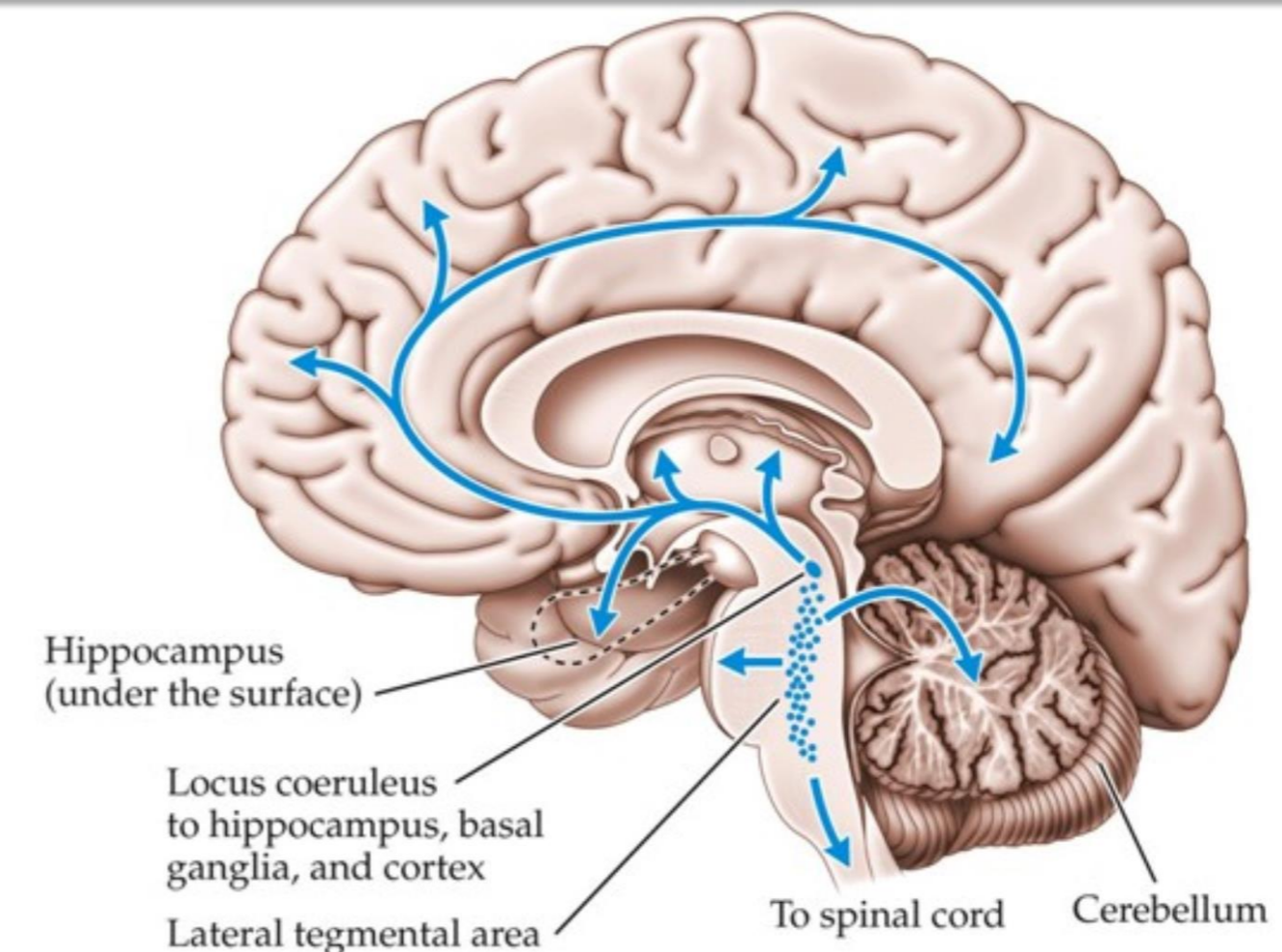
Image:
Fremaux and Gerstner, Frontiers (2016)

Image: *Biological Psychology, Sinauer*

Dopamine (DA)



Noradrenaline (NE)



BIOLOGICAL PSYCHOLOGY 7e, Figure 4.5

6. Summary

Learning outcome: RL learning rules

- **three-factor learning rules can be implemented by the brain**
 - synaptic changes need presynaptic factor, postsynaptic factor and a neuromodulator (3rd factor)
 - actor-critic and other policy gradient methods give rise to very similar three-factor rules
- **eligibility traces as 'candidate parameter updates'**
 - set by joint activation of pre- and postsynaptic factor
 - decay over time
 - transformed in weight update if neuromodulator signal comes
- **the neuromodulator can signal TD error, or novelty**
 - TD responds to reward minus expected reward
 - novelty (not familiar) can also act as a third factor

[] In this last part (navigation) at least 60 percent of the material was new to me

[] In this last part (navigation) I have the feeling that I understood 80 percent or more

[] Even though I study CS/Math/Physics/EE, I found the links to learning in biology interesting (last 2 lectures)

(previous slide)

Conclusion is NOT:

Brain implements exactly SARSA or Advantage-Actor-critic or Q-learning.

Conclusion is more modest: you can find (interesting!) correlations with signatures of RL in the brain.

Artificial Neural Networks and RL : **from brain-style computing to neuromorphic computing**

Objectives for today:

- **Spiking Neural Networks (SNN)**
- **local learning rules for hardware**
- **neuromorphic chips**
- **reducing energy consumption**

Exploit: spikes are 'rare' events.

(most of a time a neuron does not a emit a spike)

(previous slide)

Neuromorphic hardware is a hot topic.

Many (but not all) neuromorphic chips use spiking neurons.

Recent Development at IBM and INTEL:
Chip companies invest in neuromorphic
Potential reduction of energy consumption with SNN
and local learning rules (three-factor rules)

Background reading:

IBM research lab

[Bert Offrein et al., 2020, Prospects for photonic implementations of neuromorphic devices and systems, IEEE Xplore, https://ieeexplore.ieee.org/abstract/document/9371915](https://ieeexplore.ieee.org/abstract/document/9371915)

LOIHI Chip (intel)

<https://en.wikichip.org/wiki/intel/loihi>

<https://download.intel.com/newsroom/2021/new-technologies/neuromorphic-computing-loihi-2-brief.pdf>

Wulfram Gerstner

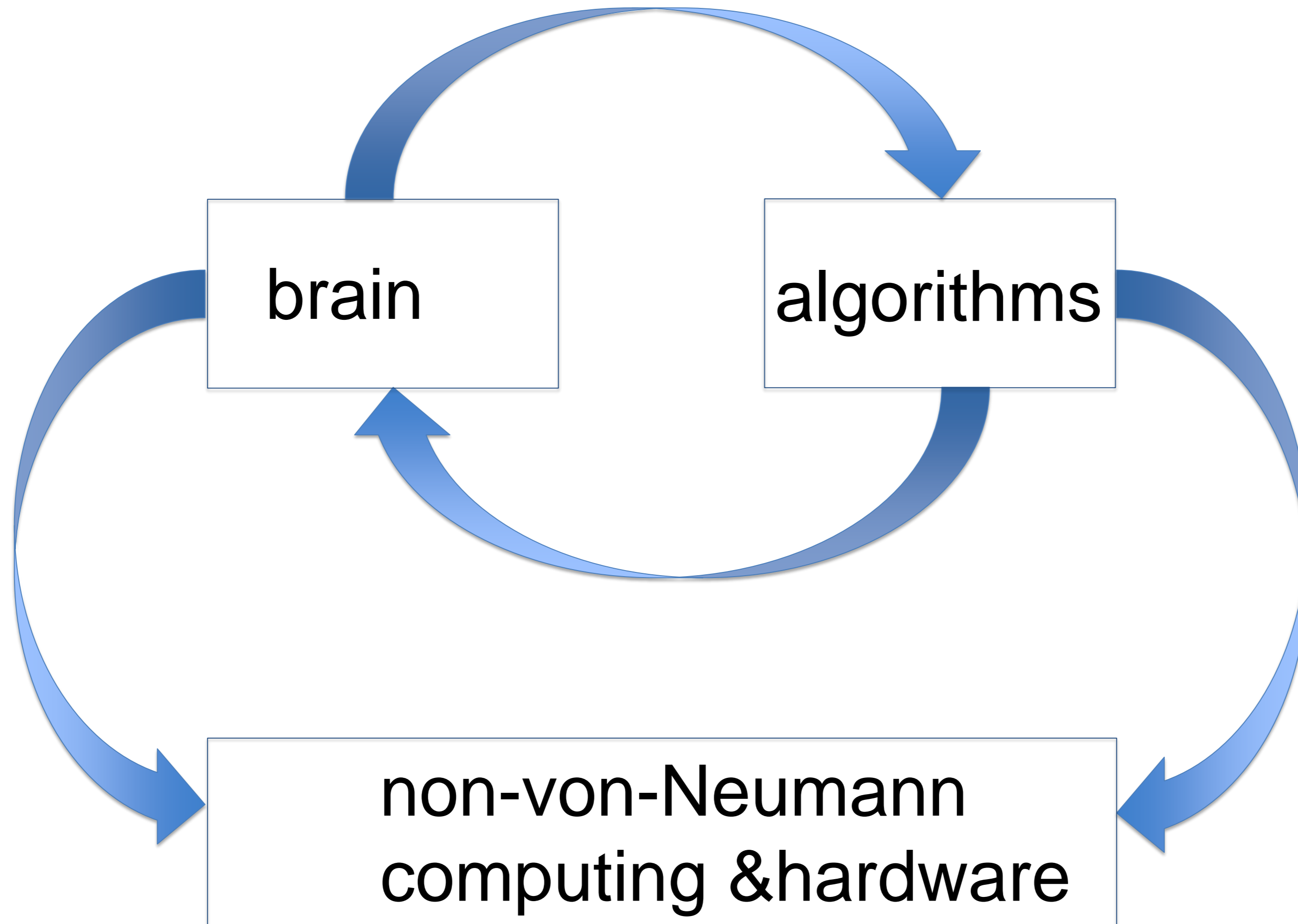
EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL: **from brain-style computing to neuromorphic computing**

1. Detour: Spiking Neural Networks (SNN)
2. Example: Navigation in a Maze (Model Study)
3. **Loihi Chip (INTEL)**

Three-factor Learning Rules

Spiking neurons (event-based signal transmission)



Previous slide:

The Loihi chip of Intel that appeared as a research support chip in 2017/2018 is interesting because it gives a direct implementation of the above three-factor rule.

INTEL:

Loihi (announced 2017, appeared 2018)

Loihi2 (announced fall 2021, access on Intel's cloud)



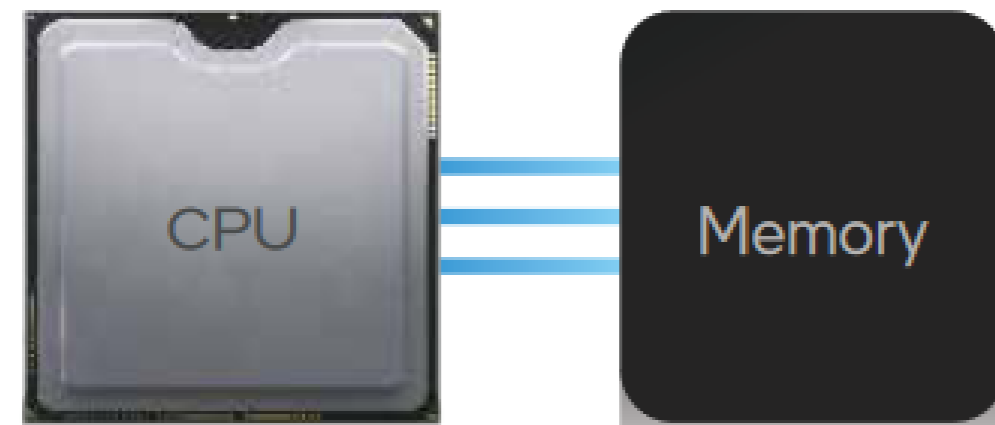
Previous slide:

More recently the first generation of Loihi has been replaced by Loihi2 with more general functionalities.

INTEL, Loihi research chip

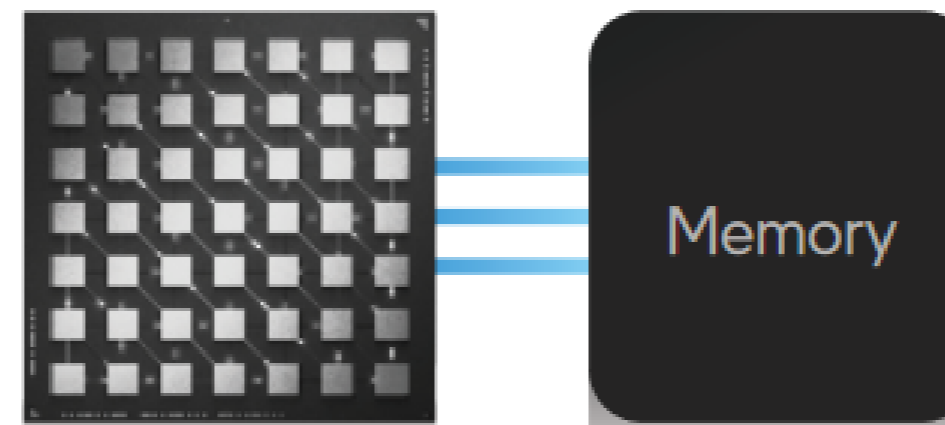
Computing Architectures

Conventional Computing



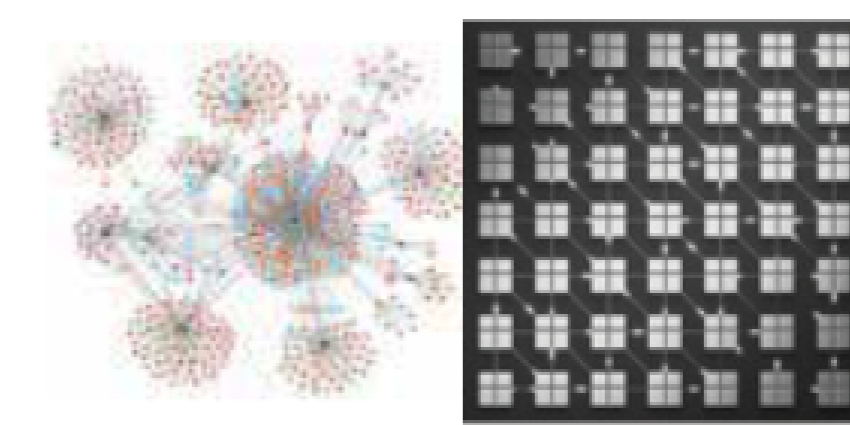
- Programming by Encoding Algorithms
- Synchronous Clocking
- Sequential Threads of Control

Parallel Computing

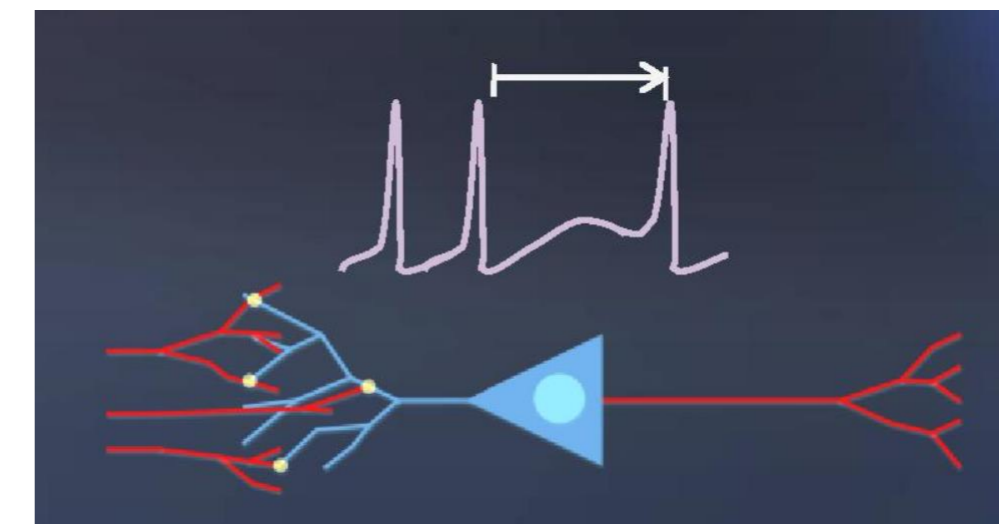
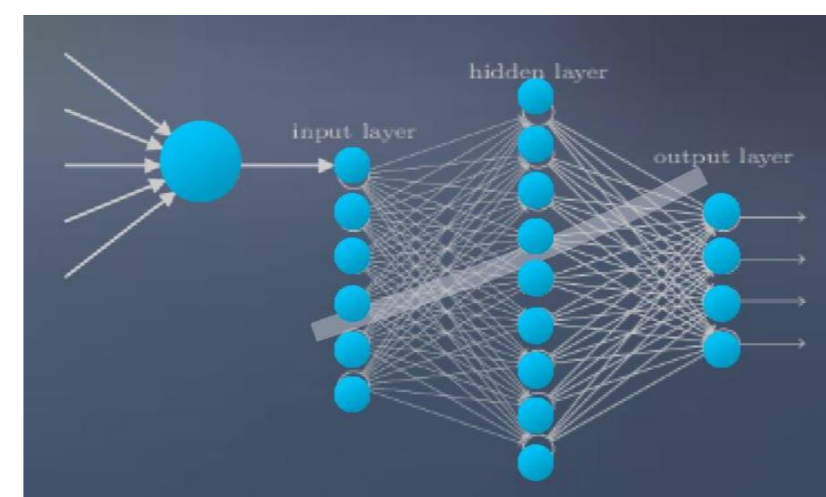


- Offline Training Using Labeled Datasets
- Synchronous Clocking
- Parallel Dense Compute

Neuromorphic Computing



- Learn On-the-Fly Through Neuron Firing Rules
- Asynchronous Event-Based Spikes
- Parallel Sparse Compute



Previous slide:

This slide from INTEL emphasize the differences in the computing architecture.

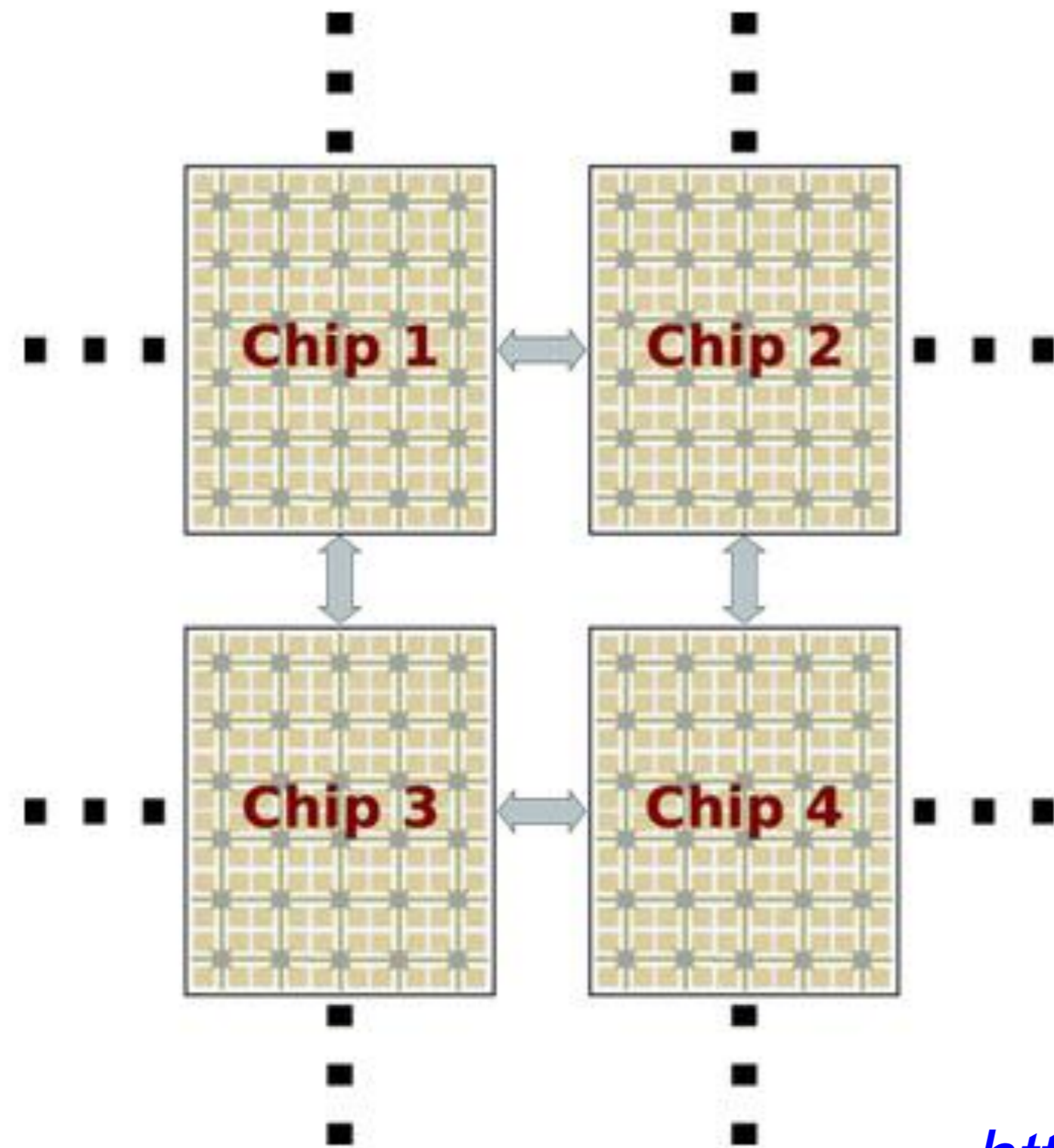
LEFT: classical von Neumann computing with separation of CPU and memory. Compute operations are mapped to logical operations performed in discrete time.

MIDDLE: Parallel computing and GPU architectures. The separation of computing and memory remains, and operations are still performed in discrete time. The only difference is that certain operations (such as convolutions) or updates of layer-wise dynamics in ANNs can be performed in parallel.

RIGHT: Neuromorphic computing architectures. Neurons compute with spikes which leads to nonlinear compute operations and signal transmission at rare moments in time defined by the moments of threshold-crossing. In between neurons are updated in 'subthreshold' mode with simple linear operations (leaky integration). Ideally, computing is asynchronous and in continuous time (even though this specific INTEL hardware implementation is still 'digital').

Two related arguments:

- energy consumption:
Loihi < 1 W (GPU > 300W)
- asynchronous computing/event-based messaging



1 chip = mesh of 128 neuromorphic cores

Spiking neural network (SNN)

1 core = 1024 simple spiking neurons:
leaky integrate-and-fire

On-chip integrated learning rule

<https://en.wikichip.org/wiki/intel/loihi>

Previous slide:

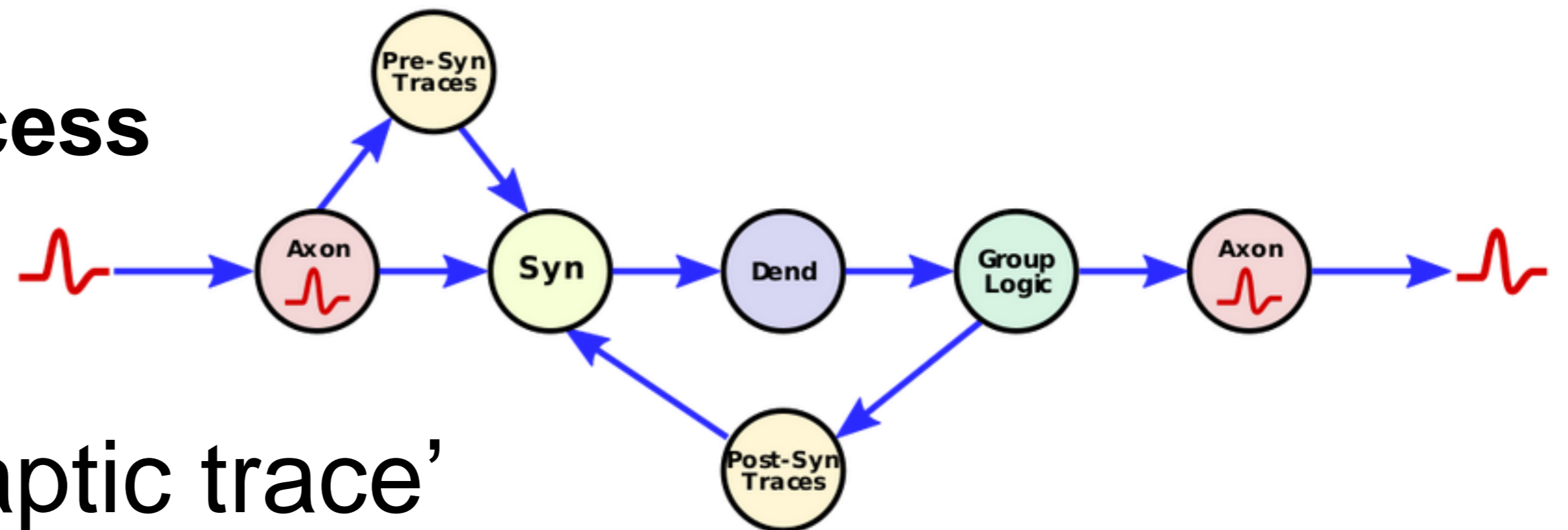
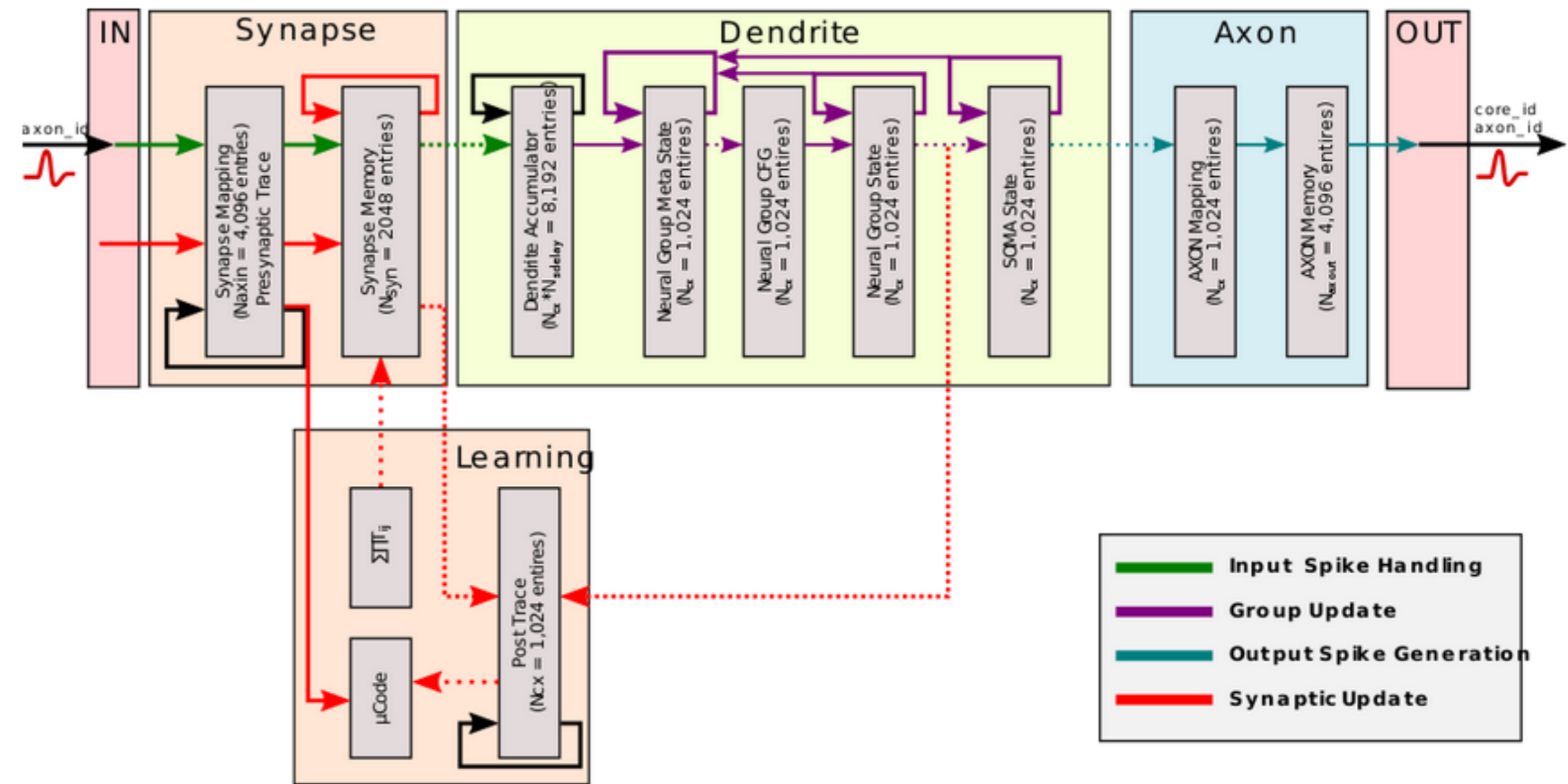
Why would one want to change the computing architecture?

Essentially because asynchronous, event-based computing could lead to enormous reductions in energy consumptions, because expensive nonlinear processing steps and transmission steps are sparse in time: they are rare compared to the elementary time step in a discrete-time implementation.

1 chip contains 128 cores, each one able to simulate about 1000 simple leaky integrate-and-fire neurons.

Loihi: (first chip, 2018)

- 128 neuron cores per chip
- Upto 128'000 neurons per chip
- 2 billion transistors
- Standard integrate-fire neuron model
- **Three-factor learning rule**
trace(pre) trace(post) success



‘each spike leaves a synaptic trace’
→ STDP coincidence

Previous slide:

Importantly, the framework of the learning rule that is possible on the Loihi chip is exactly that of three-factor rules explained above.

Each presynaptic spike leaves a trace (synaptic trace/NOT eligibility trace). The combination with the trace left by a postsynaptic spike gives the coincidence signal. Further combination with a success signal defines the weight update.

Learning rules

- Loihi (2017): Three-factor learning rules
presynaptic factor, postsynaptic factor, **global success**
→ **single-layer RL algorithms**
- Loihi2 (2022): Detailed three-factor learning rules
presynaptic factor, postsynaptic factor, **neuron-specific feedback**
→ **approximate BackProp in Multi-Layer RL**

Previous slide:

In the new version, they generalized the learning rule so that it can now also implement an approximate version of BackProp.

Introducing Loihi 2

Programmable Neurons

Neuron models described by microcode instructions

Generalized Spikes

Spikes carry integer magnitudes for greater workload precision

Enhanced Learning

Support for powerful new "three factor" learning rules from neuroscience

10x Faster

2-10x faster circuits² and design optimizations speed up workloads by up to 10x³

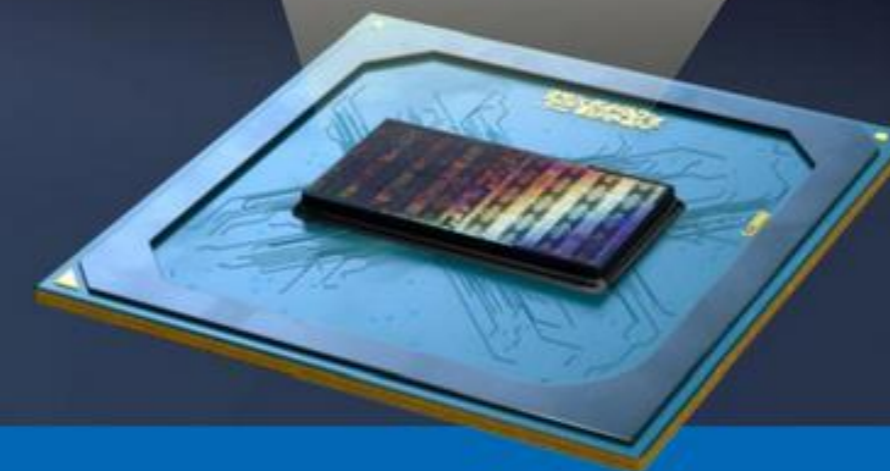
8x More Neurons

Up to 1 million neurons per chip with up to 80x better synaptic utilization, in 1.9x smaller die

Better Scaling and Integration

3D scaling with 4x more bandwidth per link⁴, >10x compression⁵ with standard interfaces

Fabricated with Intel 4 process
(pre-production)



² Based on silicon characterization of Loihi 1 and a combination of silicon and pre-silicon simulation estimates for Loihi 2.

³ Based on simulation modeling of a 9-layer Sigma-Delta Neural Network implementation of the PilotNet DNN inference workload compared to a rate-coded SNN implementation on Loihi 1.

⁴ Based on pre-silicon circuit simulations.

⁵ Based on a 7-chip Locally Competitive Algorithm workload analysis.

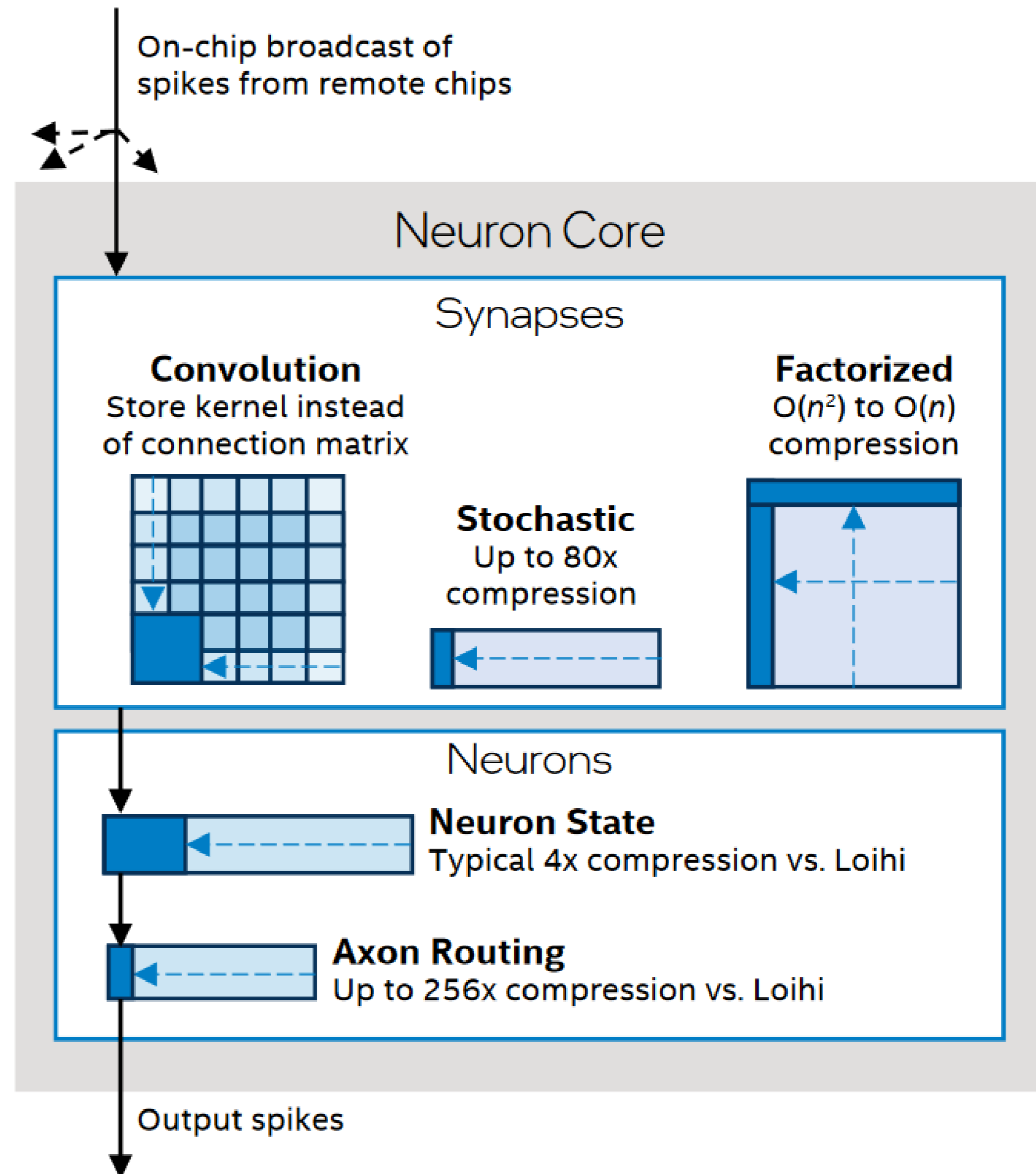
See backup for analysis details.
Results may vary.

Previous slide:

Official INTEL slide.

Loihi2 (2022):

- 128 neuron cores per chip
- Up to 1 Mio neurons per chip
- 2 billion transistors
- programmable neuron model
- programmable learning rule $f(\text{pre}), g(\text{post}), 3^{\text{rd}}(\text{neuron_i})$
- spike broadcast at destination chip
- convolutional networks
- outer-product weight matrix
- Linked to C/Python programming interface



Previous slide:

Apart from spike broadcast (as opposed to targeted delivery lines), the chip also implements features such as weight matrices compatible with convolutional neural networks and outer-product weight matrices (factorial, see conv-net lecture).

Importantly, the learning rule framework now enables the user to switch from a GLOBAL third factor to a user-defined programmable NEURON-specific third factor.

The 80-percent question again:

[] In this hardware part, at least 60 percent of the material was new to me.

[] for this hardware part, I have the feeling that I understood at least 80 percent of the material (at the rough level at which it was presented)

Wulfram Gerstner

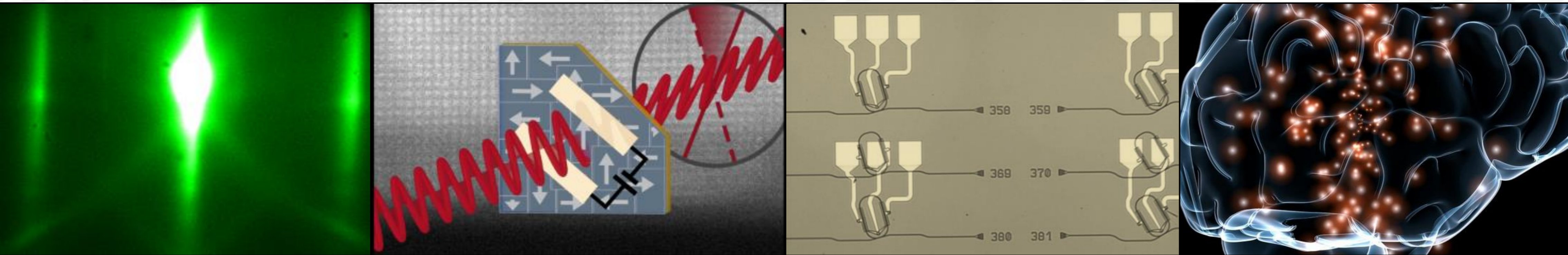
EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL : **from brain-style computing to neuromorphic computing**

1. Detour: Spiking Neural Networks (SNN)
2. Example: Navigation in a Maze (Model Study)
3. Loihi Chip (INTEL)
4. **Memristor technology (IBM)**

Analog synaptic signal processing for neural network inference and training

Bert Jan Offrein

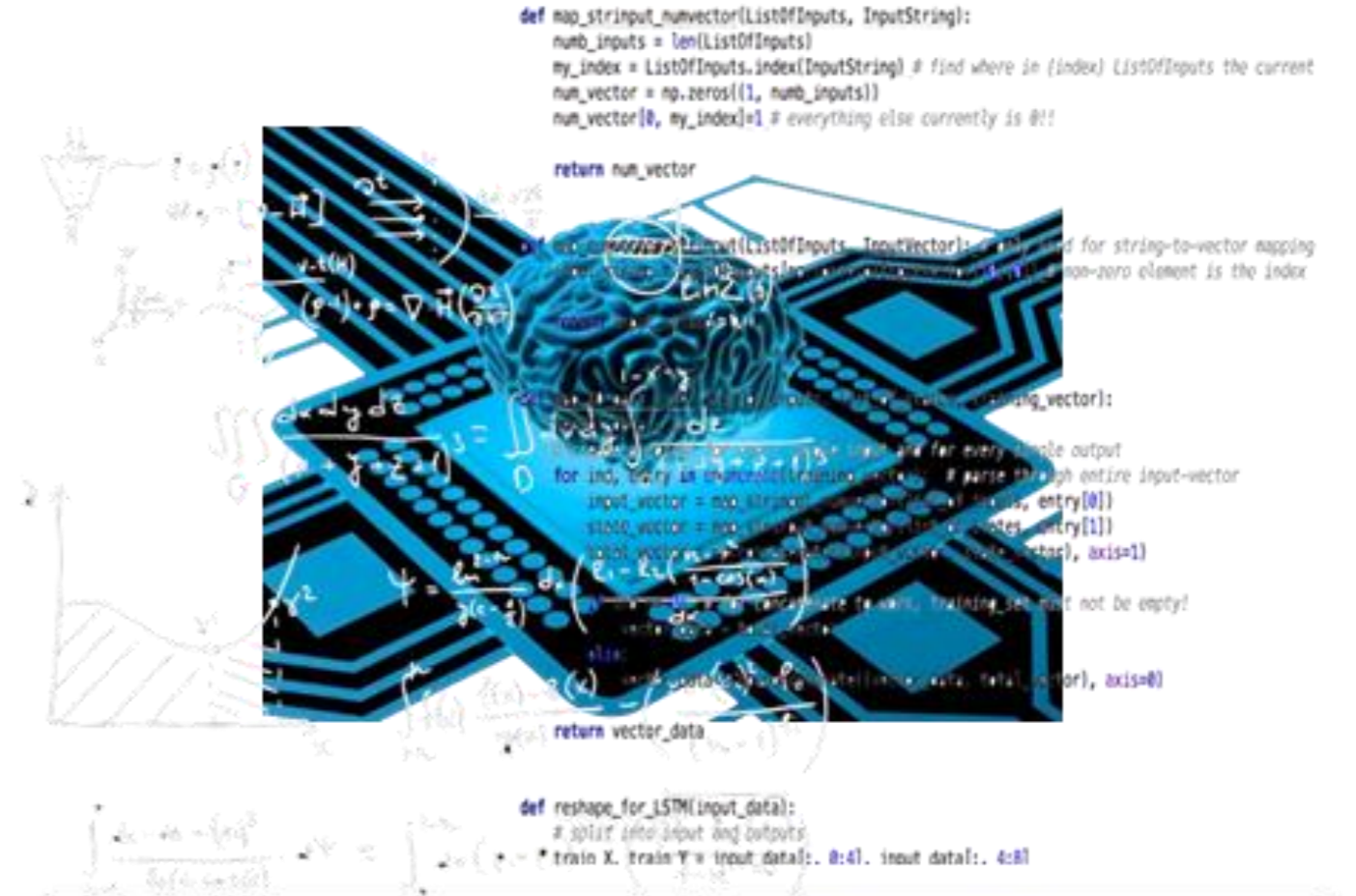
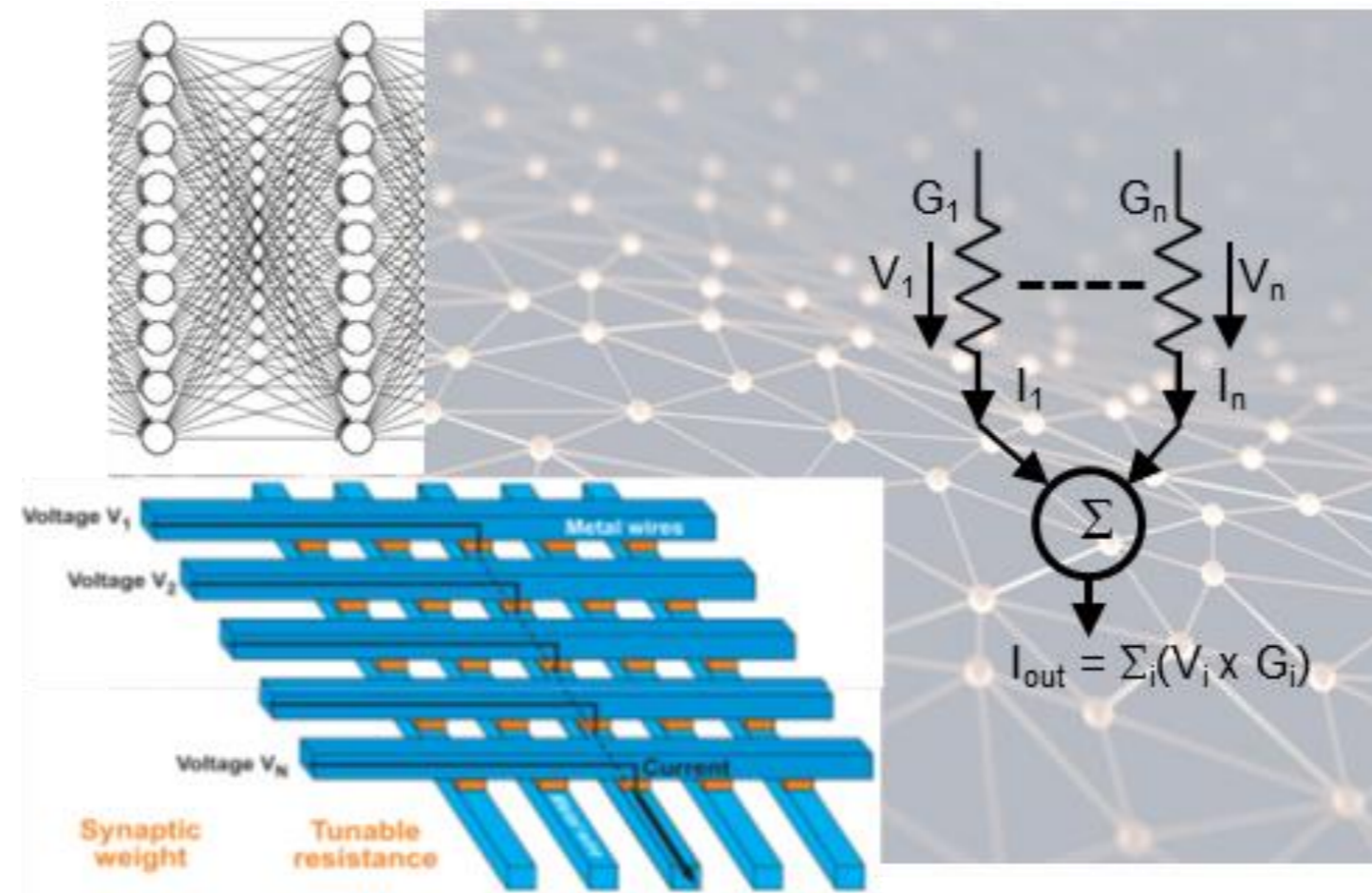
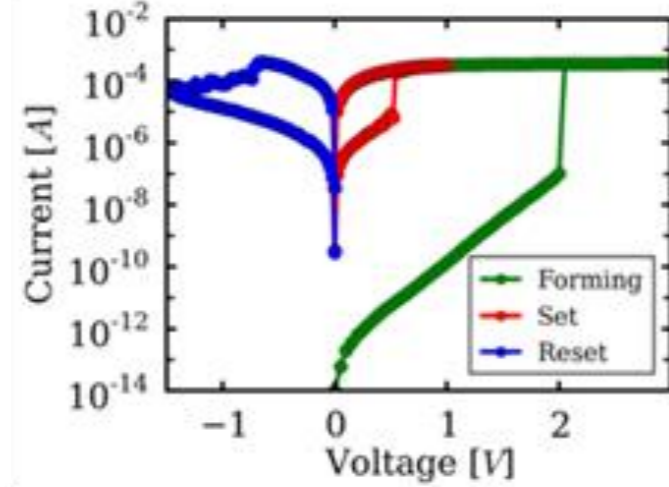
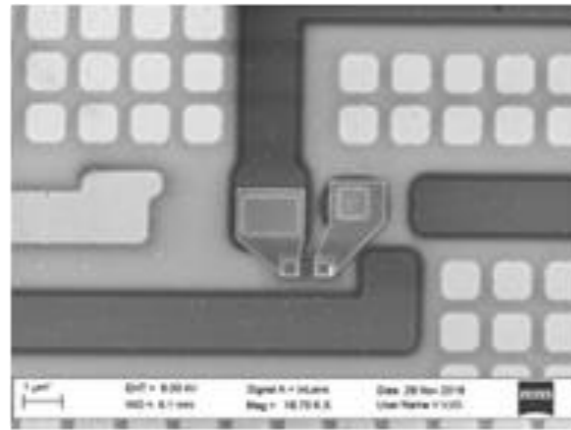
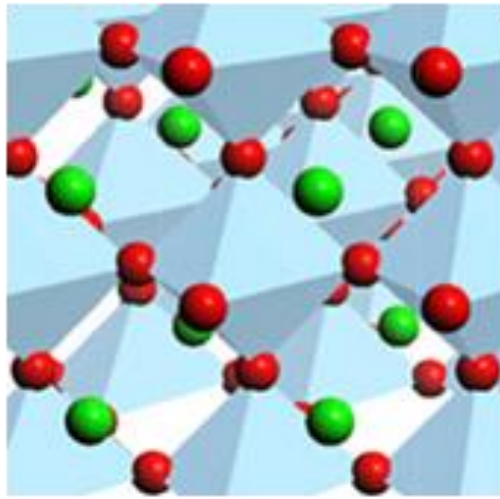


Reading

[Bert Offrein et al., 2020, Prospects for photonic implementations of neuromorphic devices and systems, IEEE Xplore, https://ieeexplore.ieee.org/abstract/document/9371915](https://ieeexplore.ieee.org/abstract/document/9371915)

The slides are adapted from a presentation of Bert Offrein who leads a group of neuromorphic computing at IBM research in Zurich-Ruschlikon.

Accelerating Neuromorphic Workloads – Innovation required at all levels



New Materials
and Devices

Non *von Neumann*
Architecture

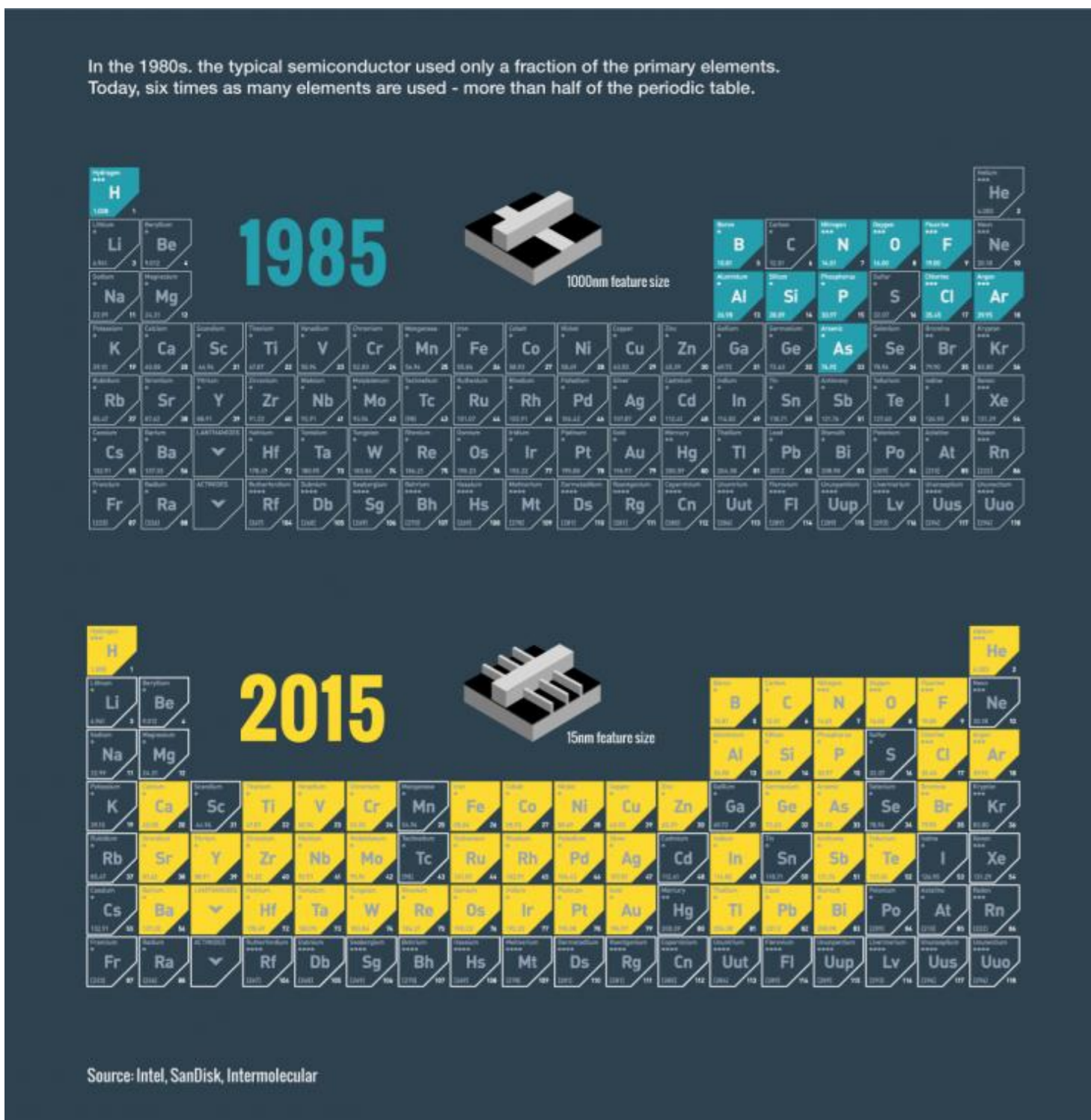
Hardware –
Algorithm Interplay

Previous slide:

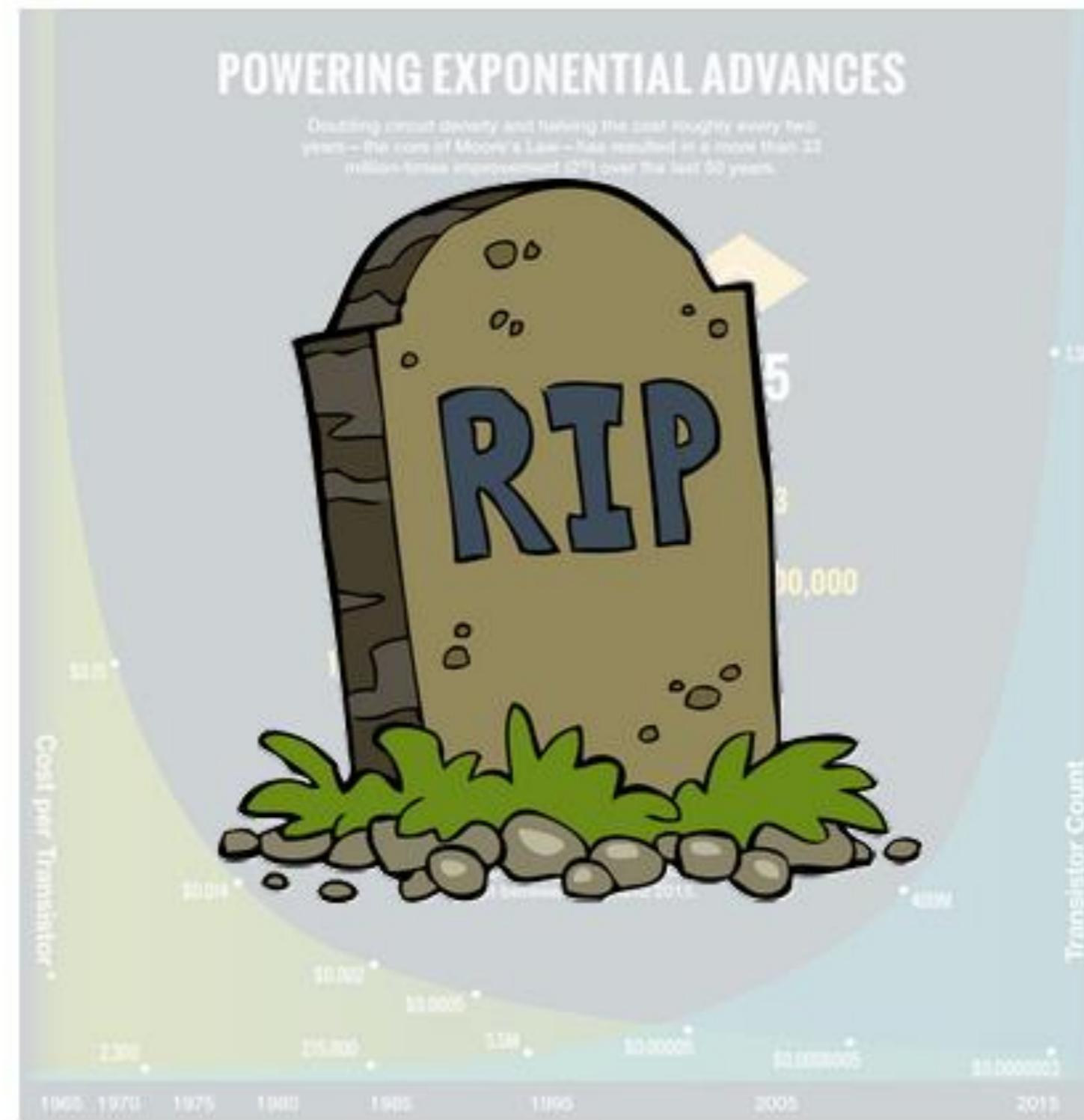
The project of IBM research focuses mostly on Matrix multiplication (middle) and update of the matrix elements as a result of a learning rule ('algorithm', right).

Three pillars for Si technology

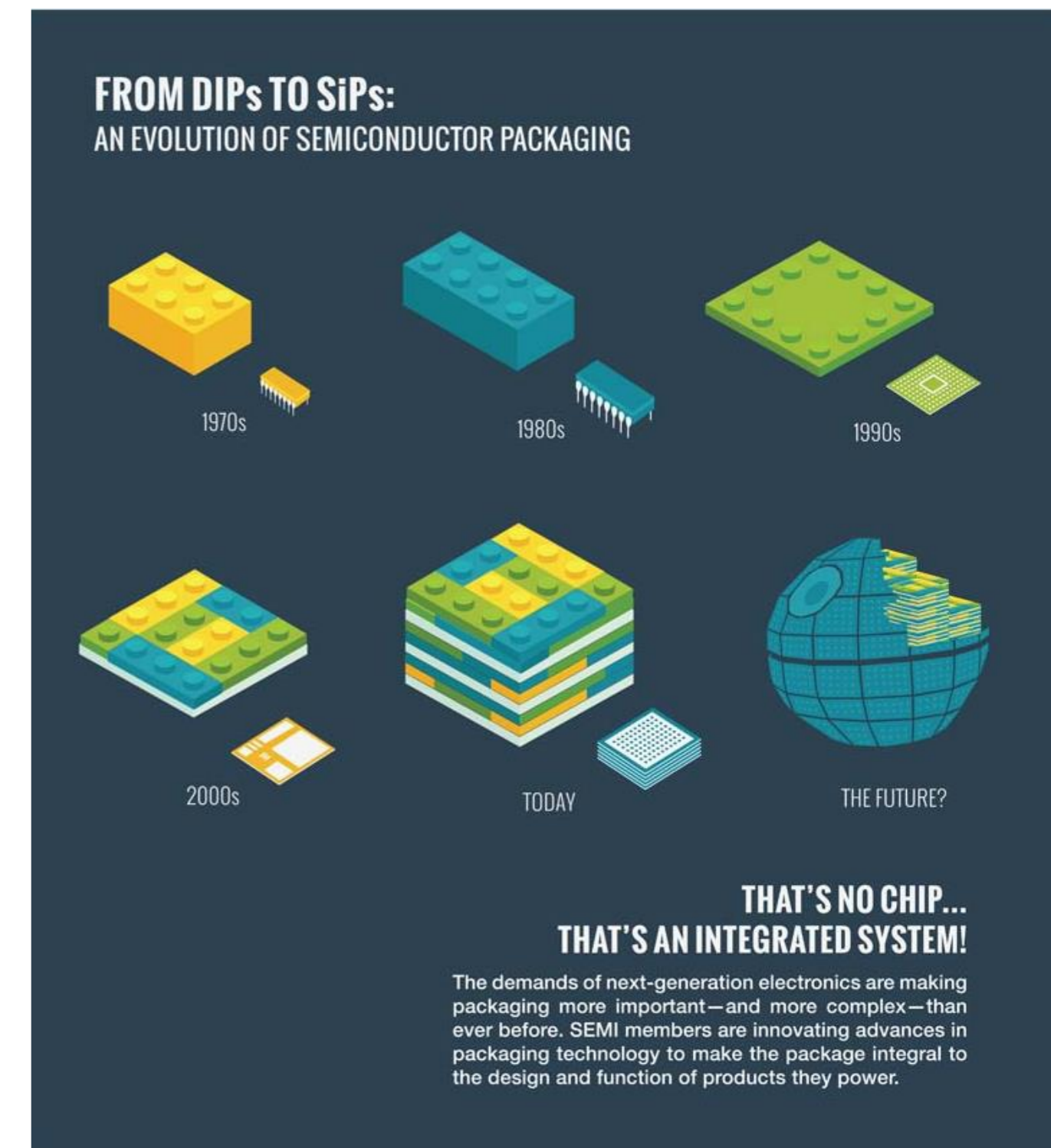
New combinations of Materials



Scaling



Packaging in 3 dimensions



www.semi.org

The traditional scaling law ('Moore's law') is dead! IBM slide

Previous slide:

In the IBM research the focus is more on new materials that enable faster and energy-efficient matrix multiplication as well as weight-update rules.

The three drivers of the changes are:

Left: New materials combine many more elements than older ones.

Middle: Moore's law, the traditional scaling law of hardware performance increase, has come to its end.

Right: Packaging has to go from 2-dim to 3-dim arrangements.

Experiment: "Human Brain vs. Computer"

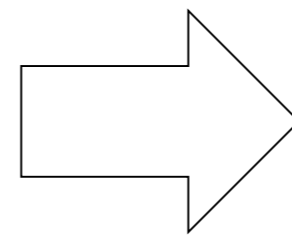
Task 1: Mathematics

$$\sqrt{2} = ?$$

Task 2: Image recognition



**Traditional silicon scaling ended
New types of problems gain interest**



**Explore new functionalities, More than Moore
Explore new computing paradigms**

- approximate computing
- large parallel data streams

Previous slide:

This shows a simple theoretical experiment, where we want to compare the performance of the human brain with a computer based on two different tasks.

In task 1, both candidates have to calculate the square root of 2 as fast as possible.

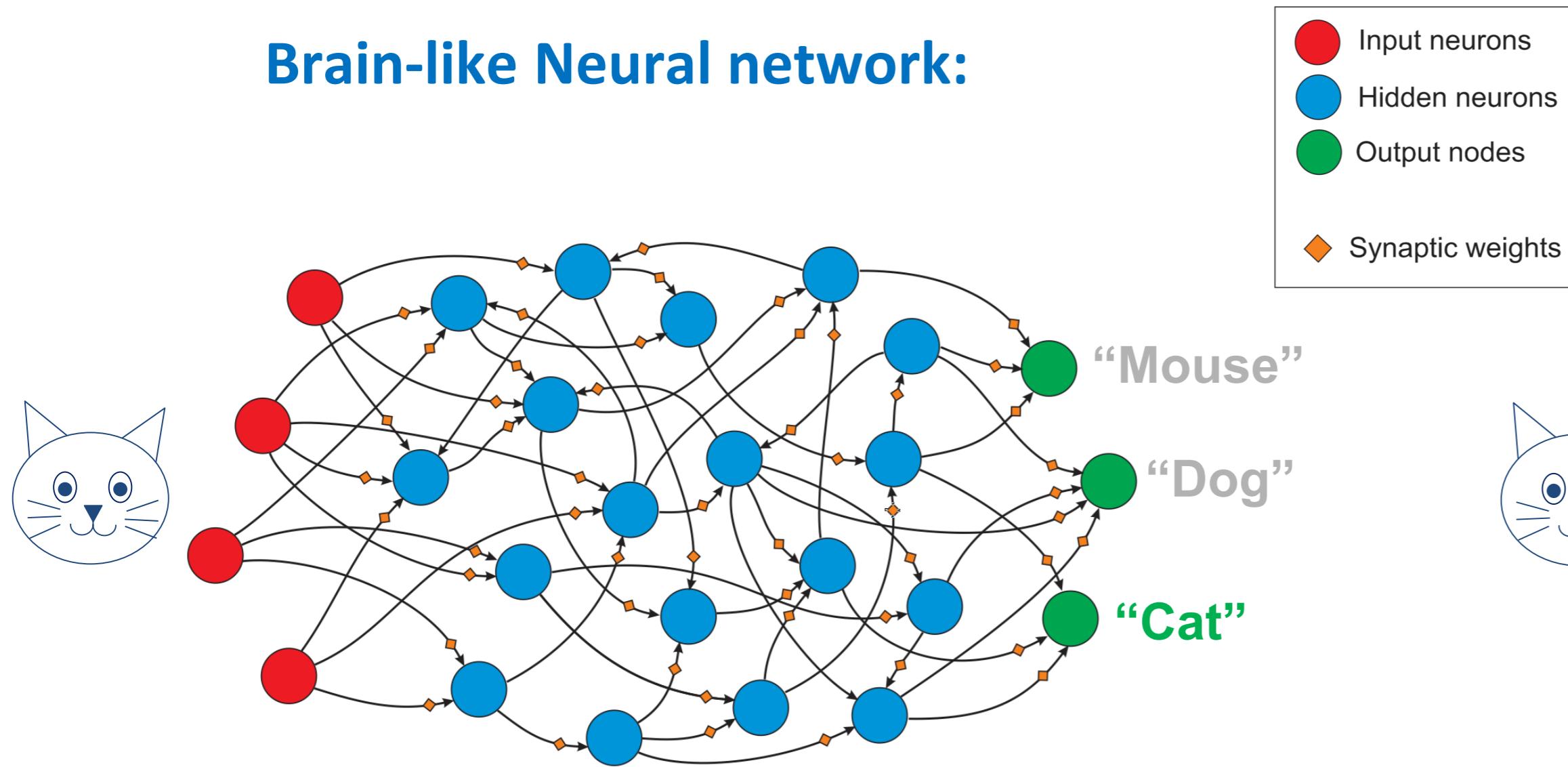
In task 2 both candidates have to interpret a scene.

The point is that that task requirements in task 2 are very different!

For example, a single noisy pixel (or noisy compute process) is less relevant. Handling of large data streams is more important.

Review Brain inspired computing:

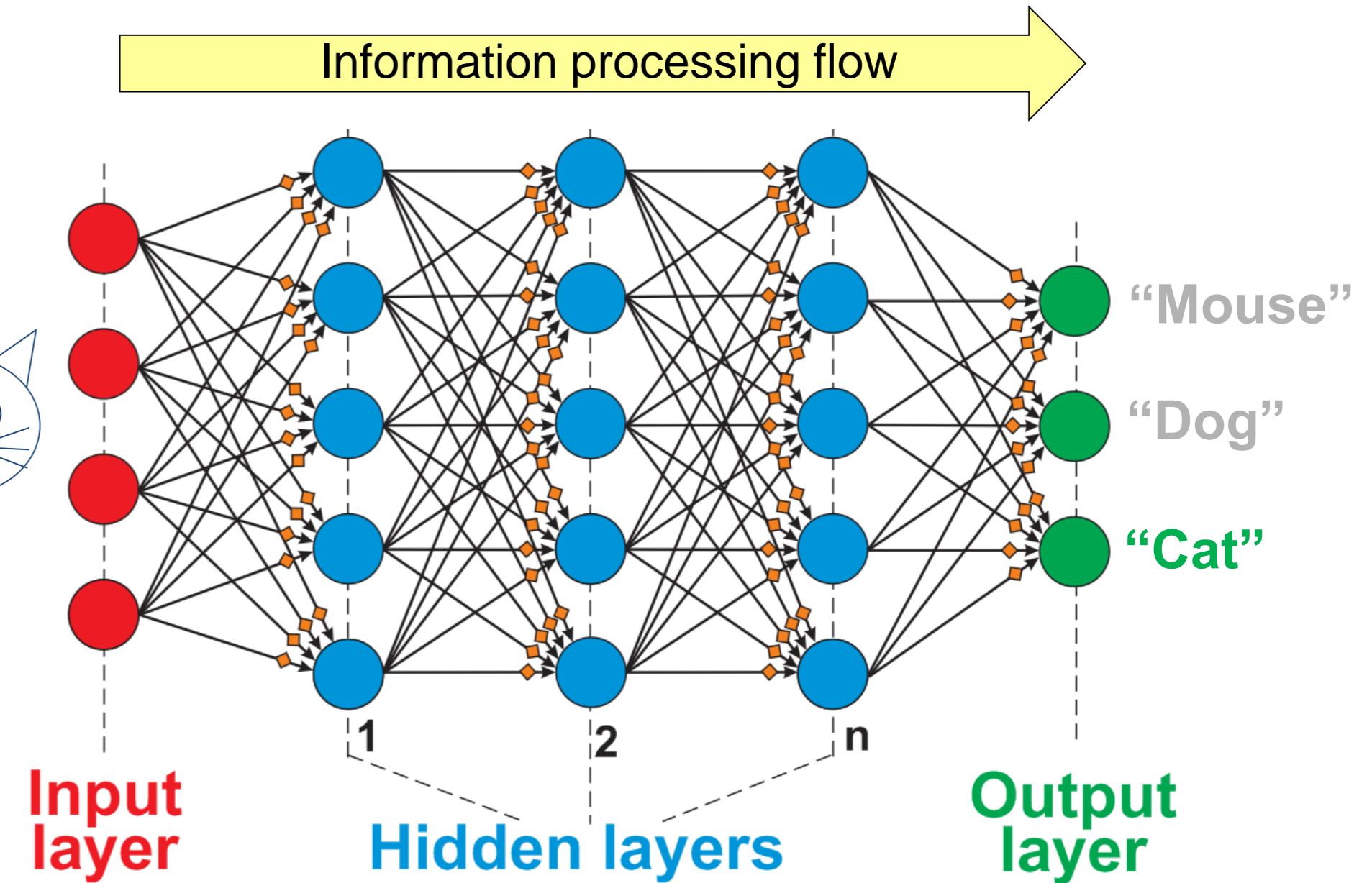
Brain-like Neural network:



Simplify



Deep Artificial Neural Network:



- Omni-directional signal flow
 - Asynchronous pulse signals
 - Information encoded in signal timing/Spiking Neural Networks
- ➔ Difficult to implement efficiently on standard computer hardware

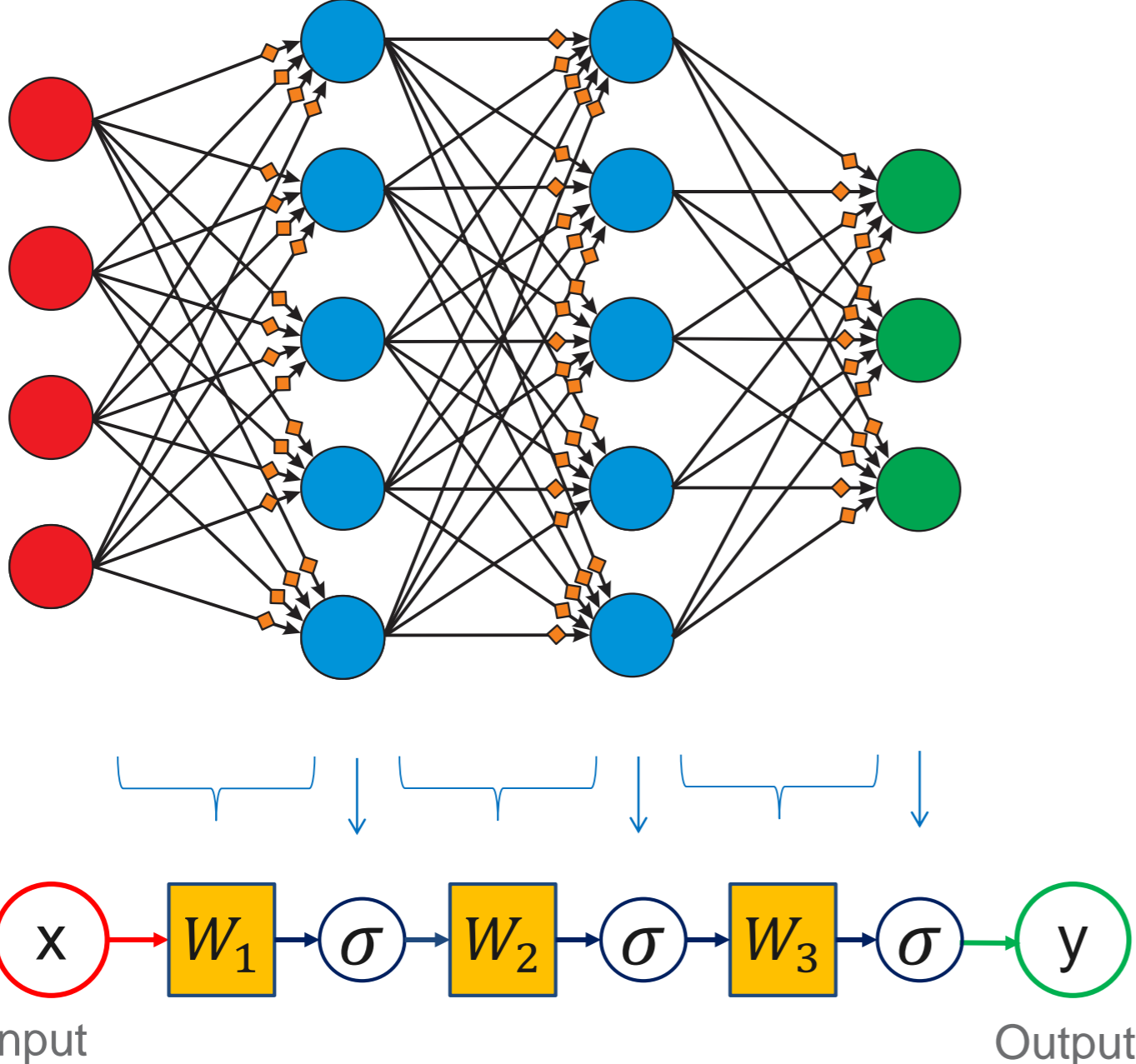
- Feed-forward sequential processing
- Information encoded in signal amplitude
- Neuron activation: Accumulate + Threshold
- **Training: Backpropagation Algorithm**

Previous slide:

Standard comparison of a few differences Brain vs ANN. Not shown in class.

Review: Training with Backpropagation algorithm

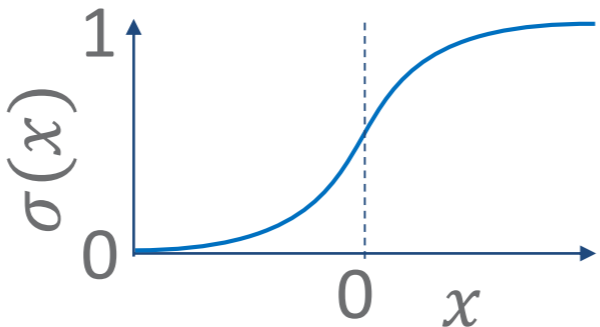
Neural net as chain of vector operations:



→ : Signal vector

W_n : Synaptic weight matrix

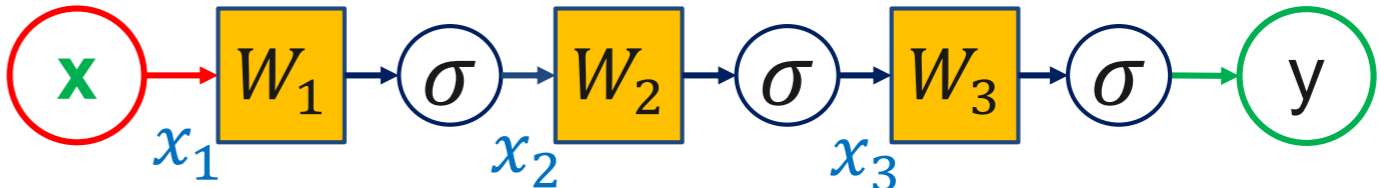
σ : Per-element neural activation function (sigmoid)



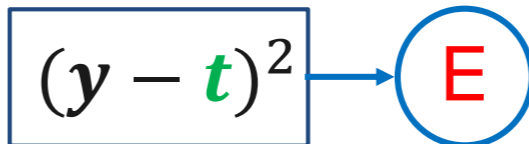
Backpropagation algorithm:

For many training cases \mathbf{x} with target response \mathbf{t} :

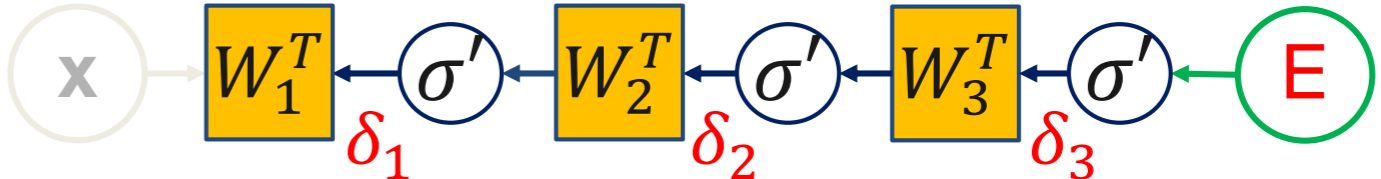
1. Forward Propagate:



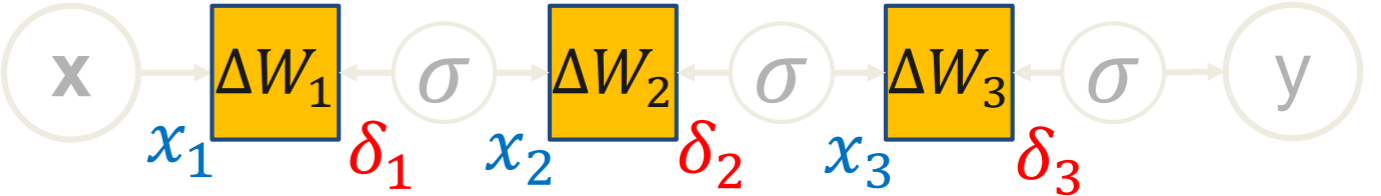
2. Determine output error:



3. Backward Propagate: Determine neuron input influence δ on error E :



4. Adjust the active weights, proportional to their influence on the error: $\Delta W = -\eta \mathbf{x} \otimes \delta$



Previous slide: not shown in class

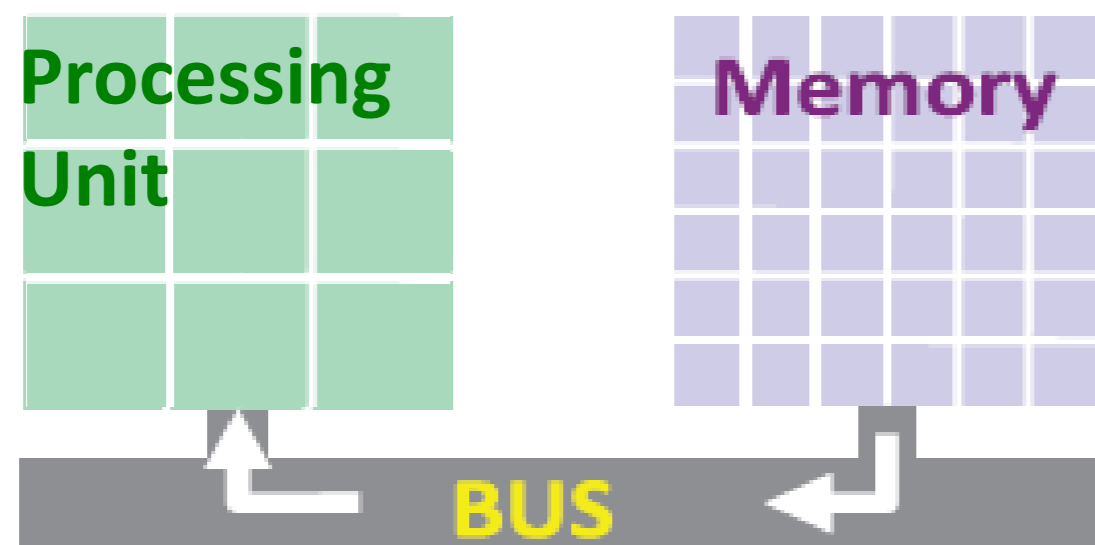
Backpropagation involves

- multiple Matrix multiplications (weight matrix per layer)
- Update of the matrix elements (learning rule)

Analog signal processing for scalability

- **Limiting factors of von Neumann architecture**

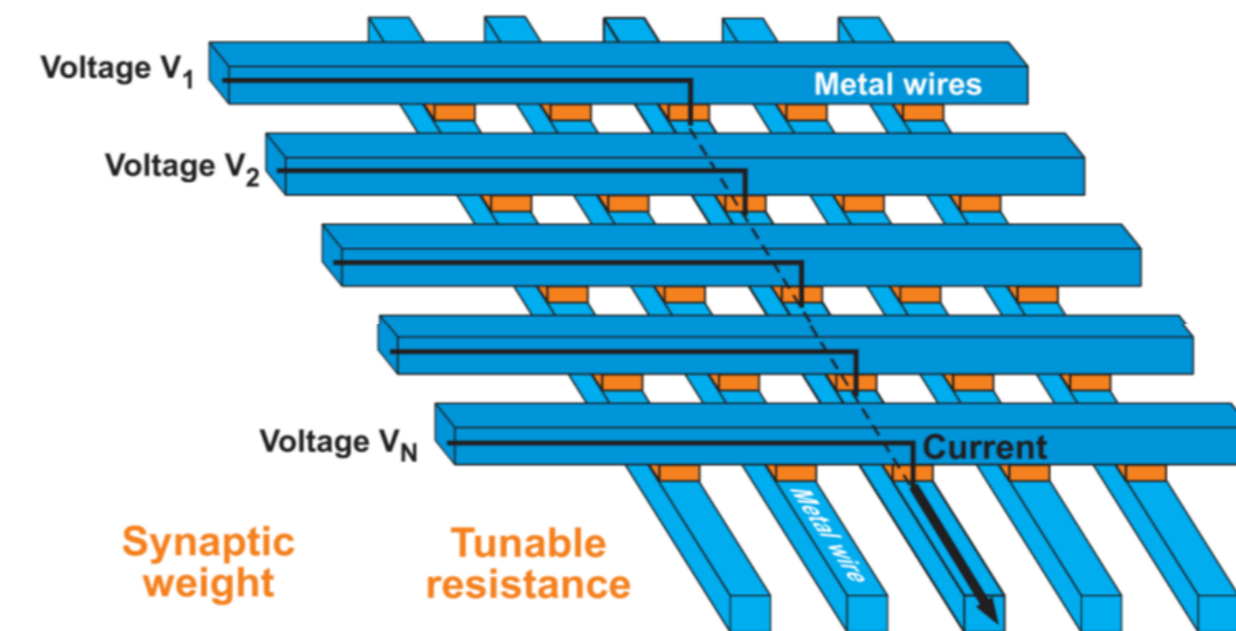
- Memory access
- Sequential operations
- Digital signal processing



Compute effort $\sim O(\#\text{Neurons}^2)$

- **Overcome by**

- In-memory computing
- Parallel operations
- Analog signal processing



Compute effort $\sim O(N)$

Electrical (and optical solutions) are viable candidates

Previous slide:

For these kind of matrix operations we should exploit new computing concepts.

The traditional von-Neumann paradigm is limited by signal flow and bad scaling as a the number of neurons per layer increases.

Training of Artificial Neural Networks: many matrix multiplications

Training by Backpropagation Method:

- **Processing dominated by many large matrix operations**

- Forward Propagation: $W_{1,2..}$

- Backward Propagation: $W_{1,2..}^T$

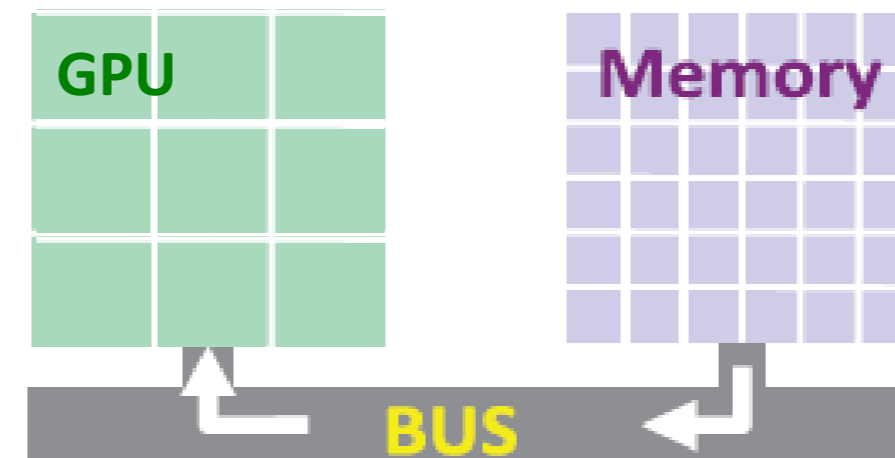
- Weight Update: $\Delta W_{1,2..}$

Scale $\propto N^2$

Neurons/layer

- **Inefficient on standard Von Neumann systems:**

- (Mostly) Serial processing
- Low computation to IO ratio \rightarrow Memory bottleneck

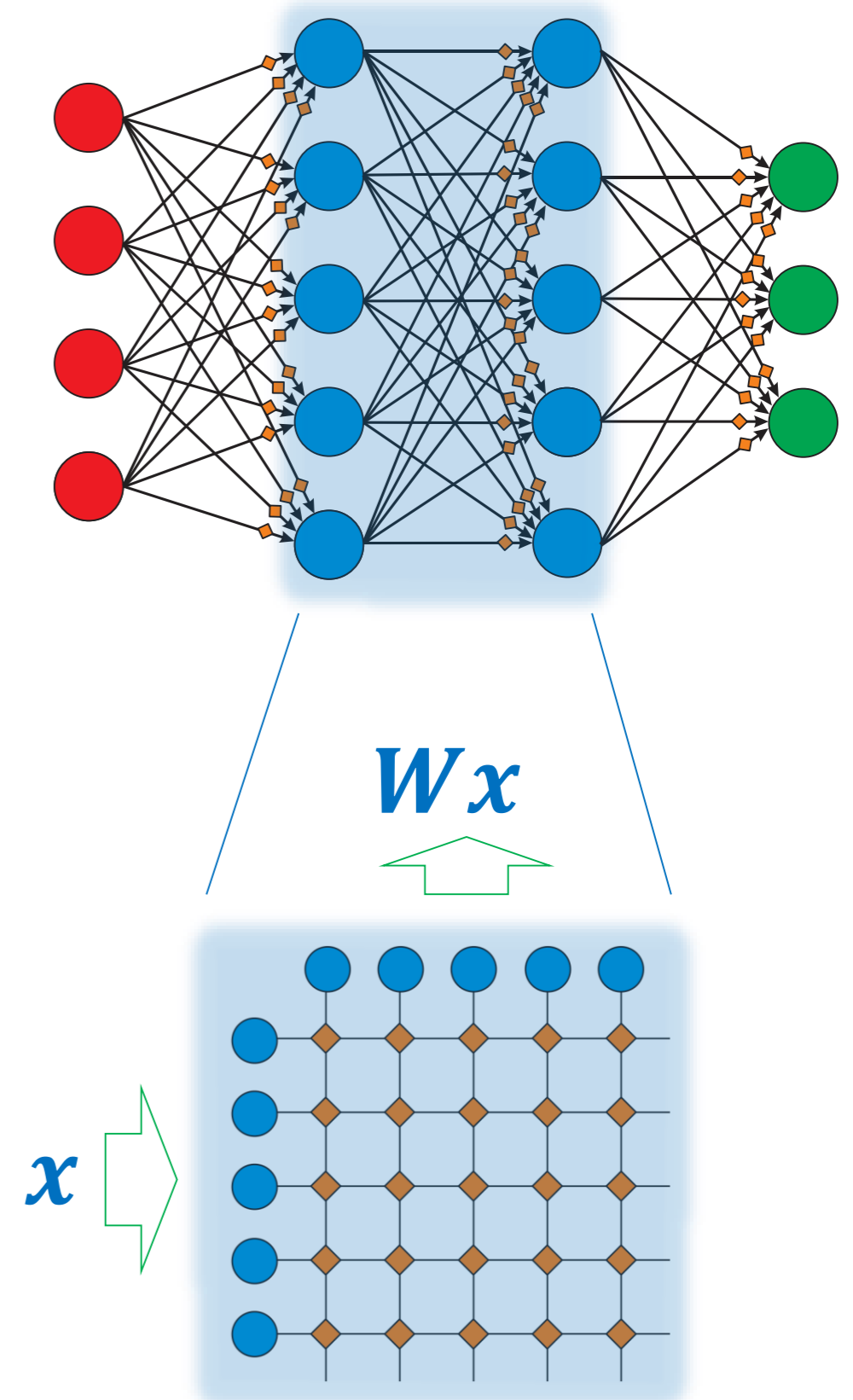


For fast and efficient neural network data processing:

- Fully parallel processing
- Tight integration of processing and memory
- Analog signal processing

Crossbar arrays

- Electrical
- Optical



G. W. Burr et al., "Tech. Dig. - Int. Electron Devices Meet. IEDM, vol. 2016–Febru, no. 408, p. 4.4.1-4.4.4, 2016.

T. Gokmen and Y. Vlasov, Front. Neurosci., vol. 10, no. JUL, pp. 1–13, 2016.

Previous slide:

Top:

In the week on BackPropagation we already discussed the scaling:

The algorithm scale proportional to the number of weights.

Assume that we have many layers and N neurons per layer. Then the scaling is $O(N^2)$.

This is true for each of the three steps: forward pass, backward pass, weight update.

Bottom:

With analog implementation of the matrix multiplication we should be able to achieve a better scaling:

Forward pass: $O(1)$

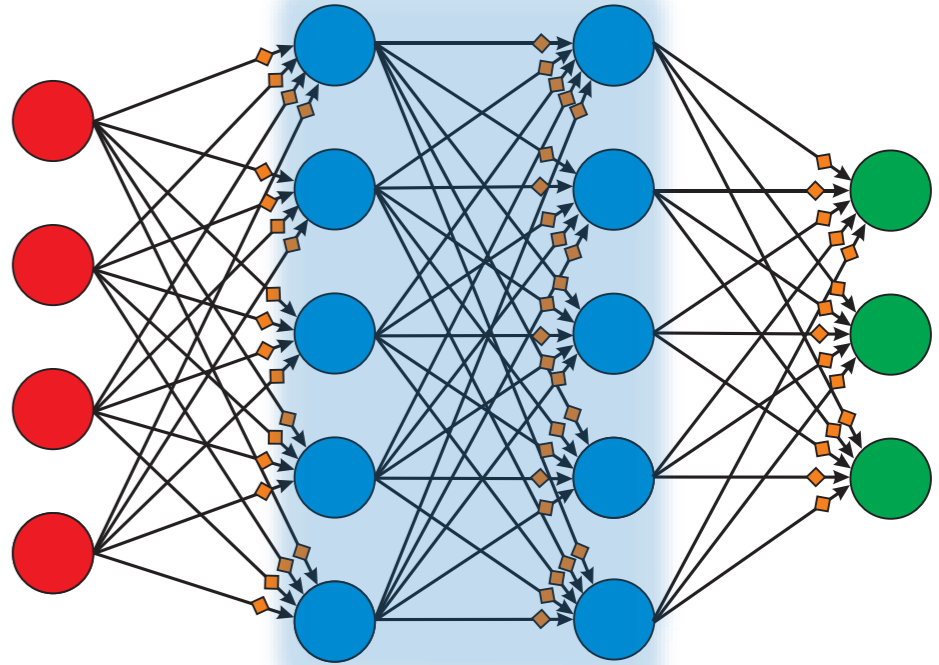
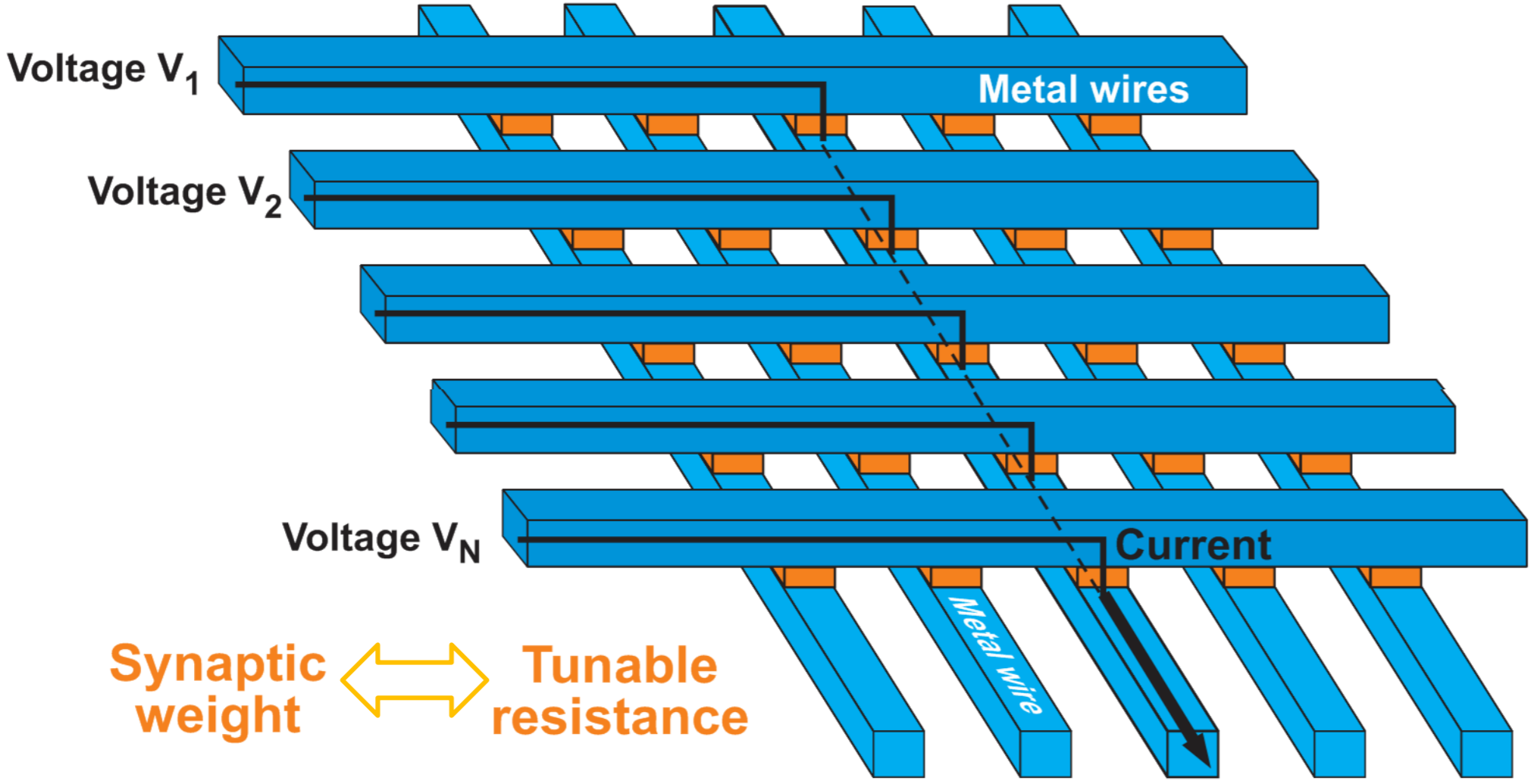
Backward pass: $O(1)$

Weight update: $O(N^2)$????

Efficient training of Deep Artificial Neural Networks:

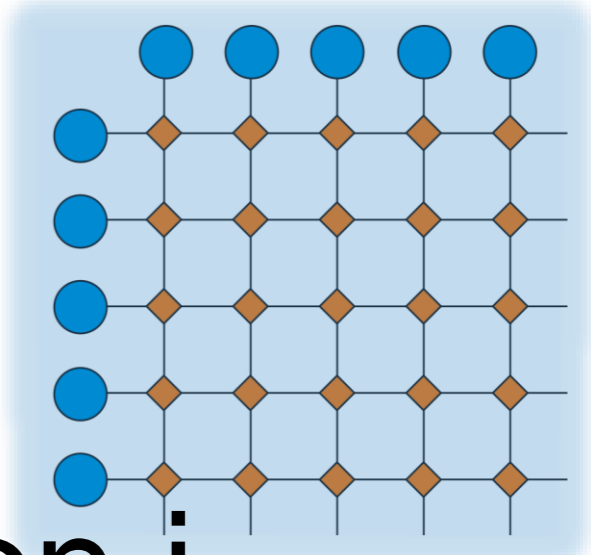
Matrix multiplication = Ohms law: $V=RI$

Electrical crossbar array:



Wx

x



Input signal $x_j = V_j$ voltage of neuron j

Weight $w_{ij} = 1/R_{ij}$ resistor at crossing

Output $I_i = \sum_j \frac{V_j}{R_{ij}} = \sum_j w_{ij} x_j$ current into neuron i

Previous slide:

Each blue bar is a perfect conductor. The red crossing points are tunable resistors that play the role of synaptic weights.

From Ohm's law follows that the current from neuron j to neuron i is $I_{ij} = V_j/R_{ij}$.

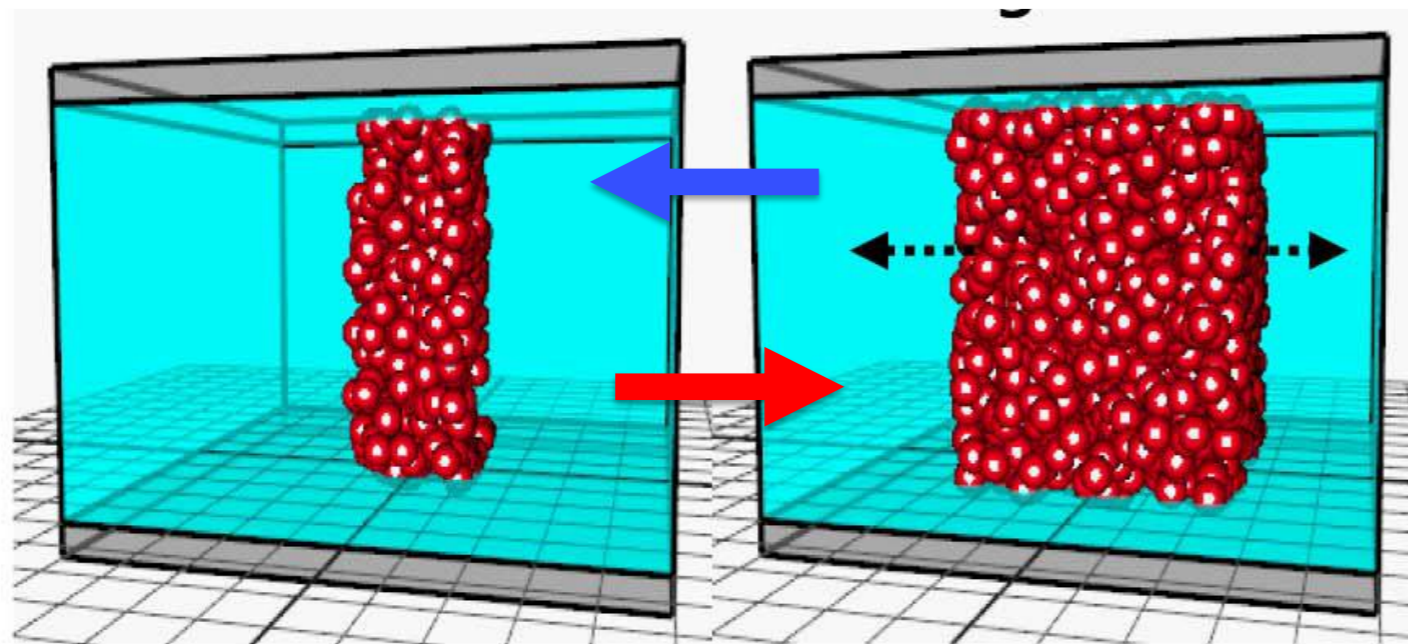
Kirchhoff's law (conservation of current) gives the final summation equation.

Tunable weights via Memristive Devices

‘memory of resistance’ = ‘memristor’

Understanding the mechanism

IBM MO_3+HfO_2
Continuous &
symmetric
change of R



Woo et al. IEEE Electr. Dev. Lett. 38, 9 (2017)

- Resistance depends on molecular configuration
- Resistance increase or decreases with voltage pulses above threshold value
- Resistance keeps memory

Previous slide:

Memristive material studied by IBM.

The basic function arises from the following principle.

The material in light blue is an electrical insulator (dielectric material). However, with a first strong voltage pulse one can create an initial breakdown in the material. This leads to a short-cut illustrated by a thin red column of molecules in a conducting state (lower left). Now the material is now longer insulating, but has a finite resistance.

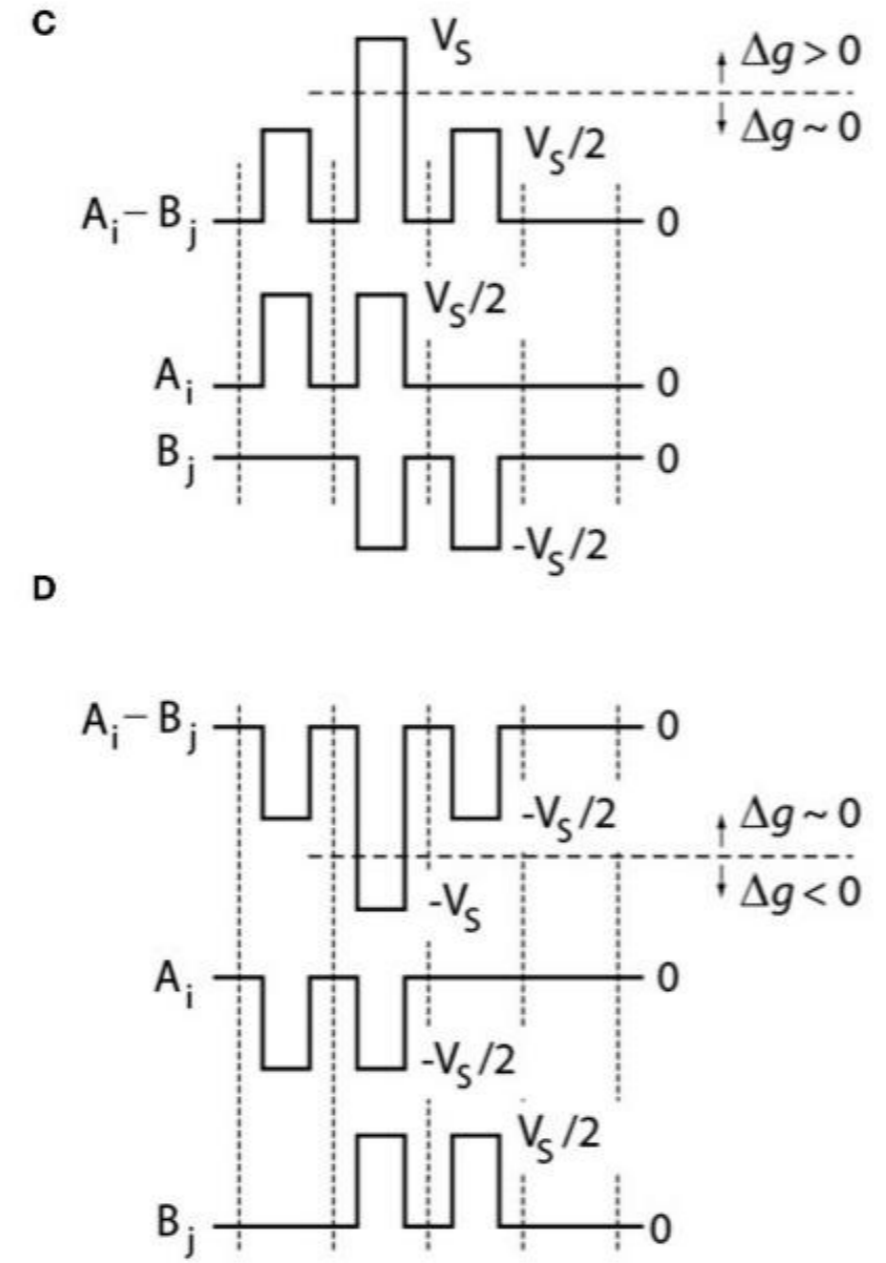
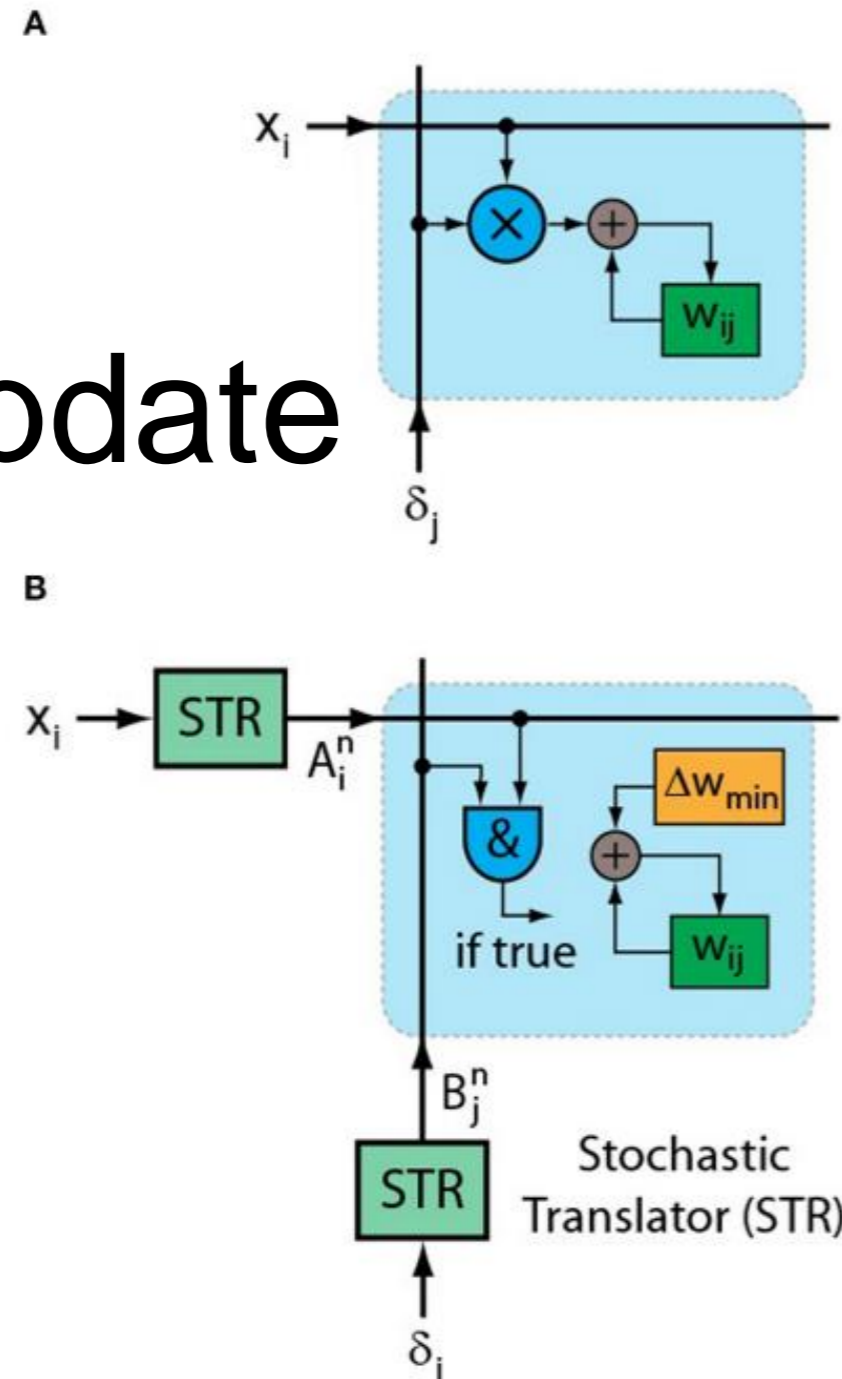
With an additional medium-sized **positive voltage pulse** (red), the column of conducting molecules can be made thicker so that the resistance decreases (lower right).

With a later medium-sized **negative voltage pulse** (blue), one can return to the initial configuration (lower left).

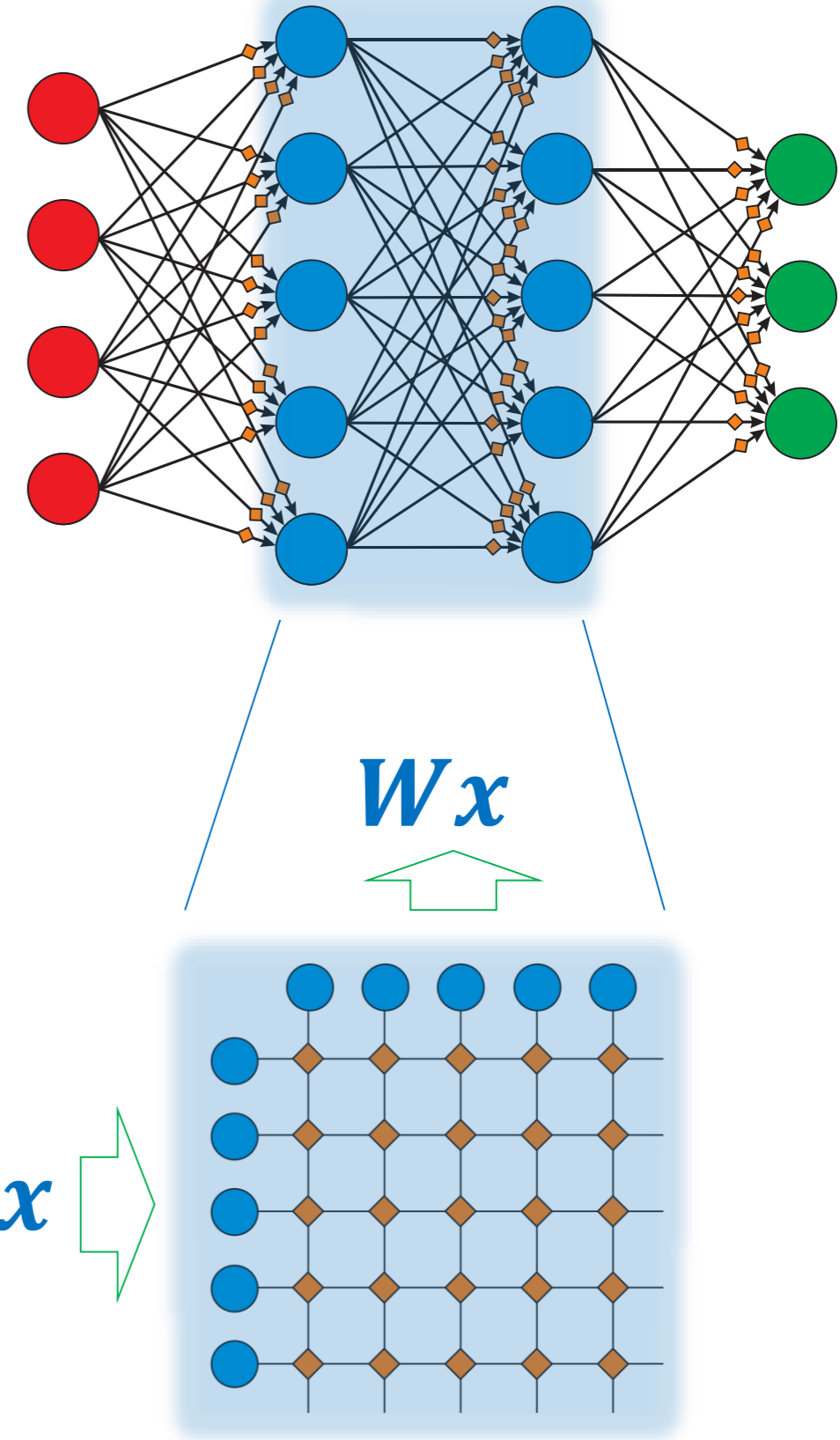
Weak currents and weak voltage pulses have no effect. Hence the material keeps its configuration and resistance for a long time. It has a '**Memory of Resistance**' → **Memristor**.

Efficient training of Deep Artificial Neural Networks: spiking network

Weight update order 1 !



- Local Hebbian Learning rule
- Spike coding (SNN)



For fast and efficient neural network data processing:

- Fully parallel processing
- Pulse coding
- Stochastic Poisson Process

Crossbar arrays
• Electrical

Gokman and Vlasov, Acceleration of Deep Neural Networks with Resistive Cross-Point Devices Frontiers, 2016

Previous slide:

We now image the following coding principle (not yet implemented in hardware, but proposed some years ago).

Each presynaptic neurons sends voltage pulses ('spikes' of finite width) at random moments in time (Poisson process).

Each postsynaptic neuron sends voltage pulses ('spikes' of finite width) at random moments in time (an independent Poisson process).

The amplitude of the single pulse is such that it does not reach the switching amplitude of the memristive material. But if two pulses coincide, then it reaches the threshold and increases the weight (decreases the resistance).

Thus we have a proposition to implement a local (two-factor) Hebbian learning rule in hardware. And, unexpectedly, we need spike coding for this implementation scheme!

FIGURE 1 | (A) Schematics of original weight update rule of Equation (1) performed at each cross-point. **(B)** Schematics of stochastic update rule of Equation (2) that uses simple AND operation at each cross-point. Pulsing scheme that enables the implementation of stochastic updates rule by RPU devices for **(C)** up and **(D)** down conductance changes.

$$w_{ij} \leftarrow w_{ij} + \eta x_i \delta_j \quad (1)$$

where w_{ij} represents the weight value for the i^{th} row and the j^{th} column (for simplicity layer index is omitted) and x_i is the activity at the input neuron, δ_j is the error computed by the output neuron and η is the global learning rate.

In order to implement a **local and parallel** update on an array of two-terminal devices that can perform both weight storage and processing (RPU) we first propose to significantly simplify the multiplication operation itself by using stochastic computing techniques (Gaines, 1967; Poppelbaum et al., 1967; Alaghi and Hayes, 2013; Merkel and Kudithipudi, 2014). It has been shown that by using two stochastic streams the multiplication operation can be reduced to a simple AND operation (Gaines, 1967; Poppelbaum et al., 1967; Alaghi and Hayes, 2013). **Figure 1B** illustrates the stochastic update rule where numbers that are encoded from neurons (x_i and δ_j) are translated to stochastic bit streams using stochastic translators (STR). Then they are sent to the crossbar array where each RPU device changes its conductance (g_{ij}) slightly when bits from x_i and δ_j coincide. In this scheme we can write the update rule as follows.

$$w_{ij} \leftarrow w_{ij} \pm \Delta w_{min} \sum_{n=1}^{BL} A_i^n \wedge B_j^n \quad (2)$$

Gokman and Vlasov, Acceleration of Deep Neural Networks with Resistive Cross-Point Devices
Frontiers in Neuroscience, 2016

Previous slide:

This is a copy of the relevant section of the original publication

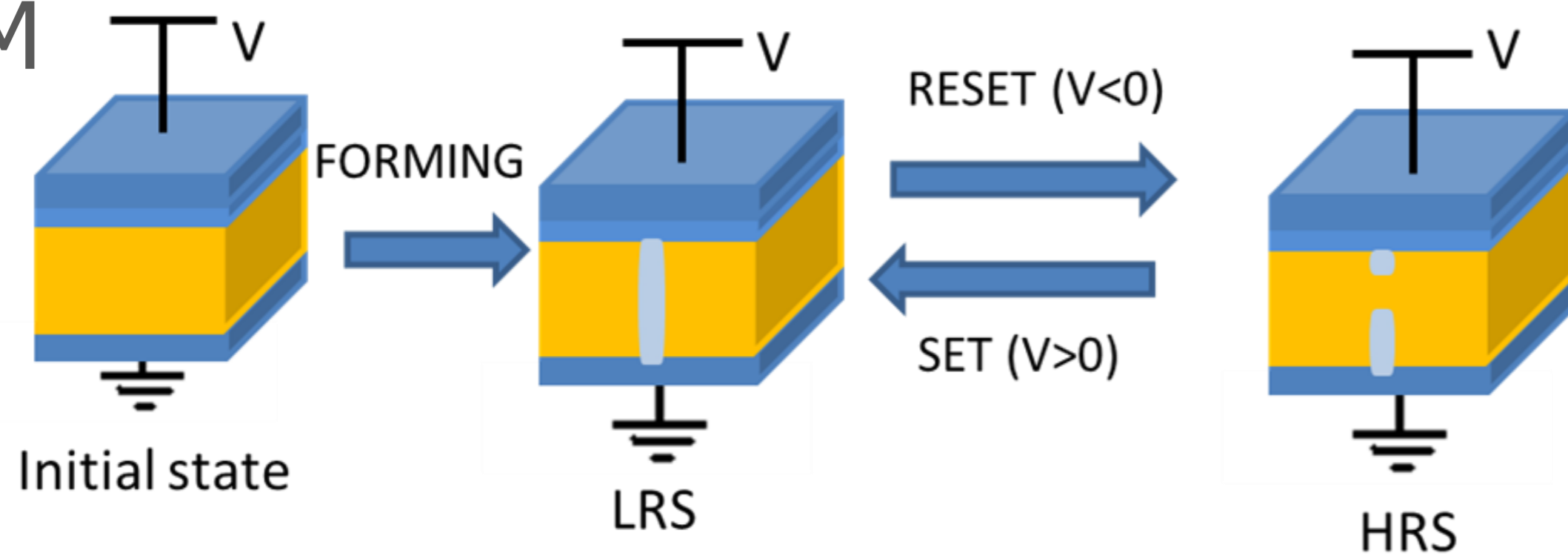
The device challenge

- Create breakdown ('mild shortcut')
- Make size of breakdown tunable

$$w = 1/R$$



RR
AM



Courtesy E. Vianello

Images: IBM

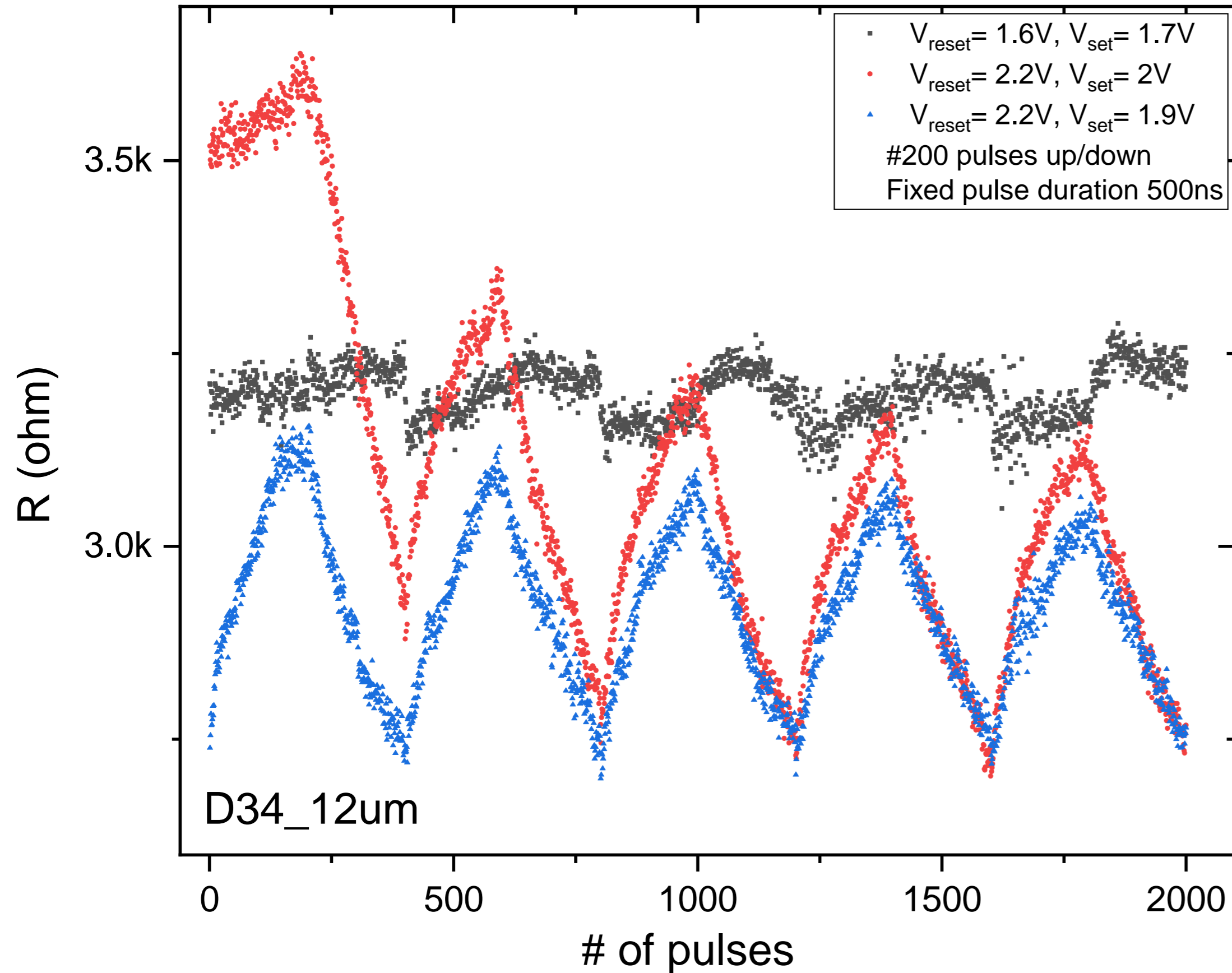
Previous slide:

Here The material in yellow is an electrical insulator (dielectric material). However, with a first strong voltage pulse one can create an initial breakdown (blue channel) in the material.

The question now is the following: Can we SMOOTHLY TUNE
sith several additional medium-sized **positive voltage pulse** (red), or **negative voltage pulse** (blue), one can return to the initial configuration (lower left).

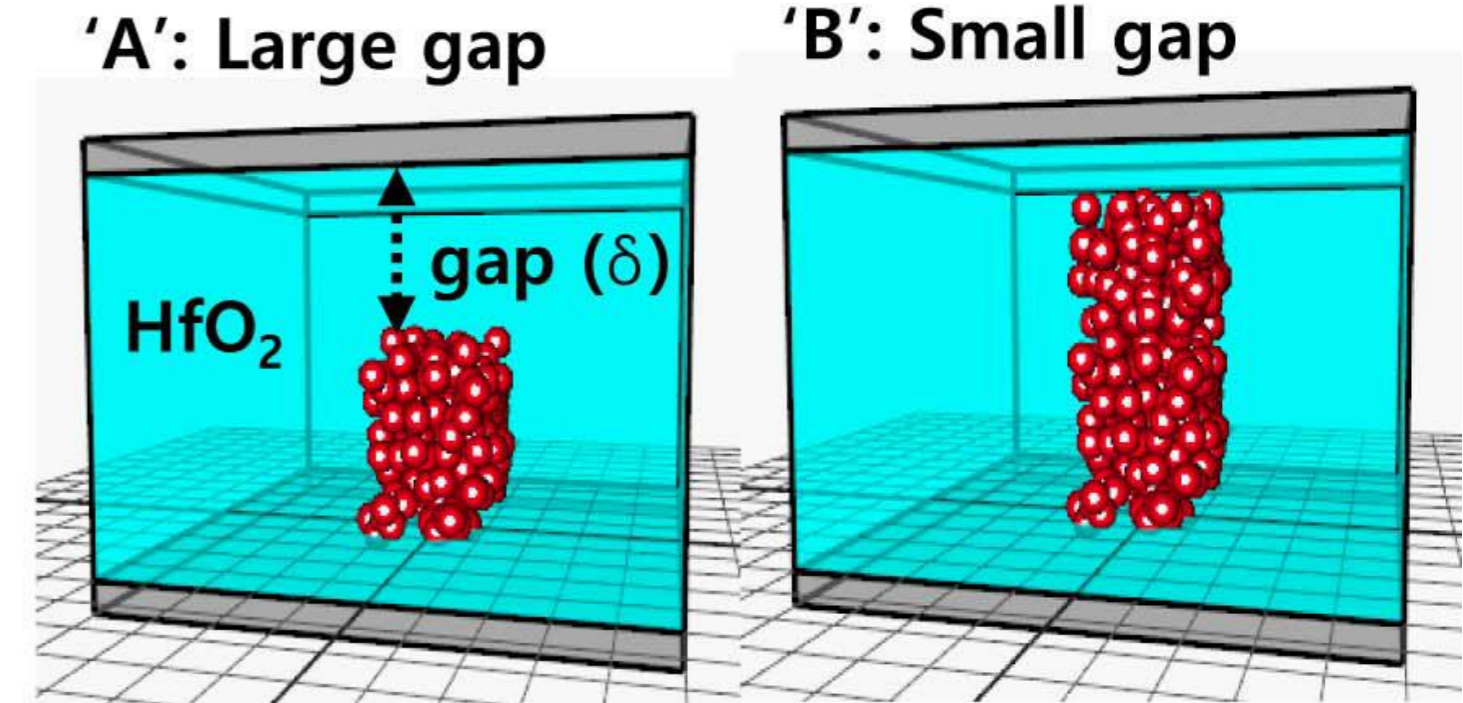
Changes induced by 200 pulses up (and down)

→ change the weights of ANN by appropriate pulses

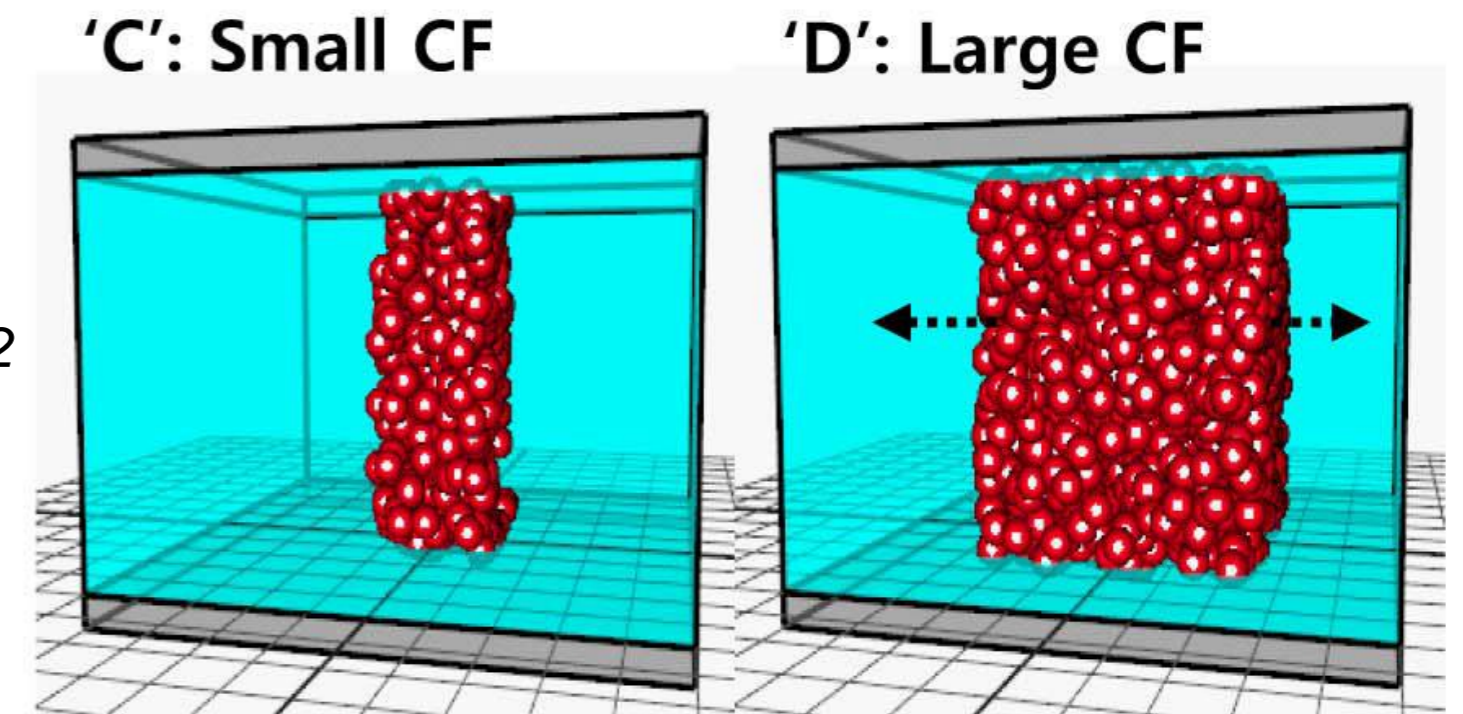


Understanding the mechanism

HfO_2 baseline
Abrupt switching



IBM $\text{MO}_3 + \text{HfO}_2$
Continuous & symmetric change of G



Woo et al. IEEE Electr. Dev. Lett. 38, 9 (2017)

Experimental demonstration of symmetric and continuous change of G

Images: IBM

Previous slide:

Experimental test with the material at the bottom shows that smooth tuning is possible (blue dots). Horizontal axis shows the number of pulses applied. After about 200 pulses the sign is switched so that the resistance goes down again.

Retention of the Resistance Values over time (intermediate values)

1. Negative sweep to put it in Low-Resistance State
2. Read at 0.2V constantly for 1000s
3. Negative + positive cycle to put it in increasing High-Resistance State (HRS)
4. Read at 0.2V constantly for 1000s for each HRS

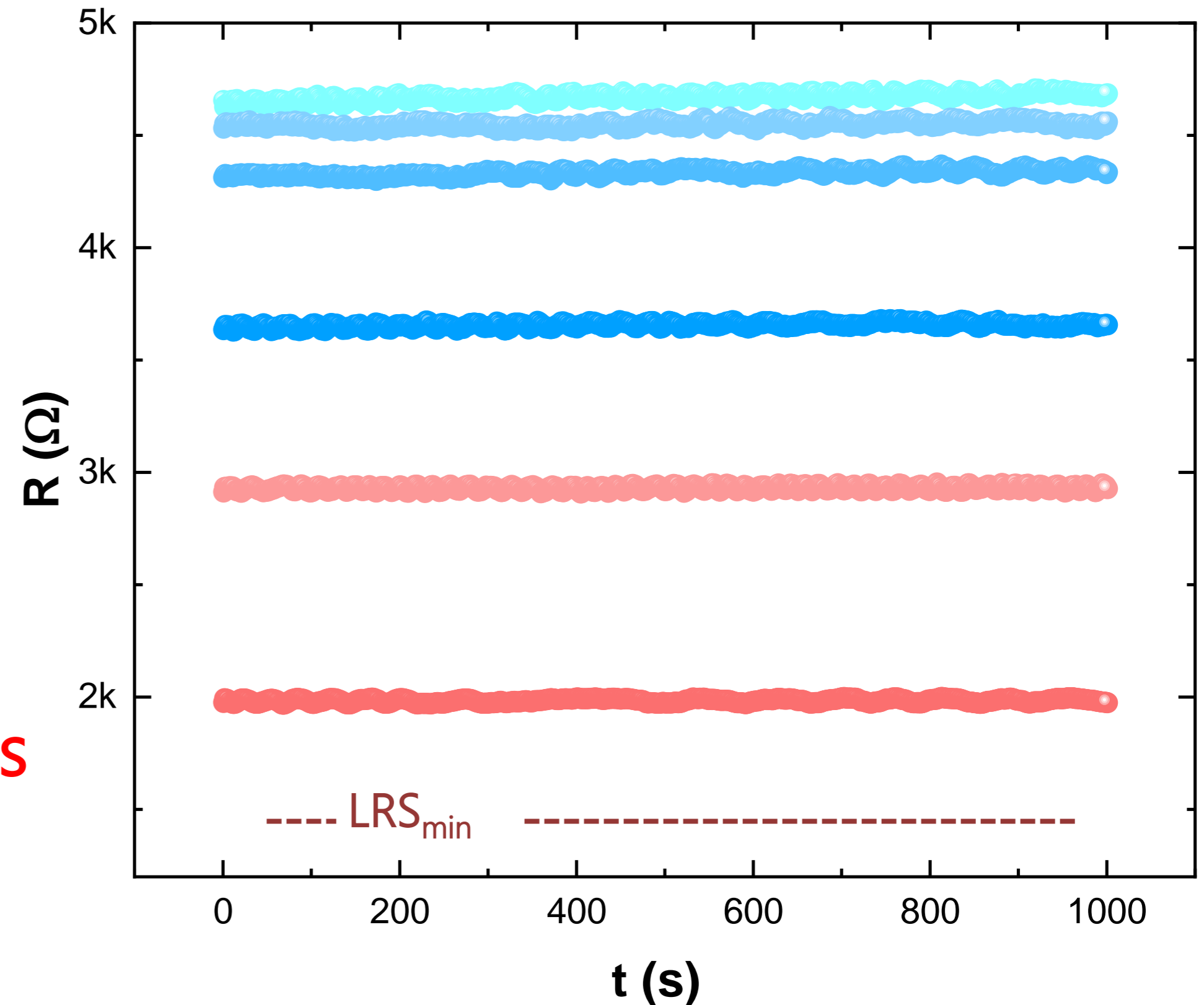
Intermediate states show no drift up to 1000 s

→ We can change values of resistance

→ New resistance is reliable over time

→ We can change again

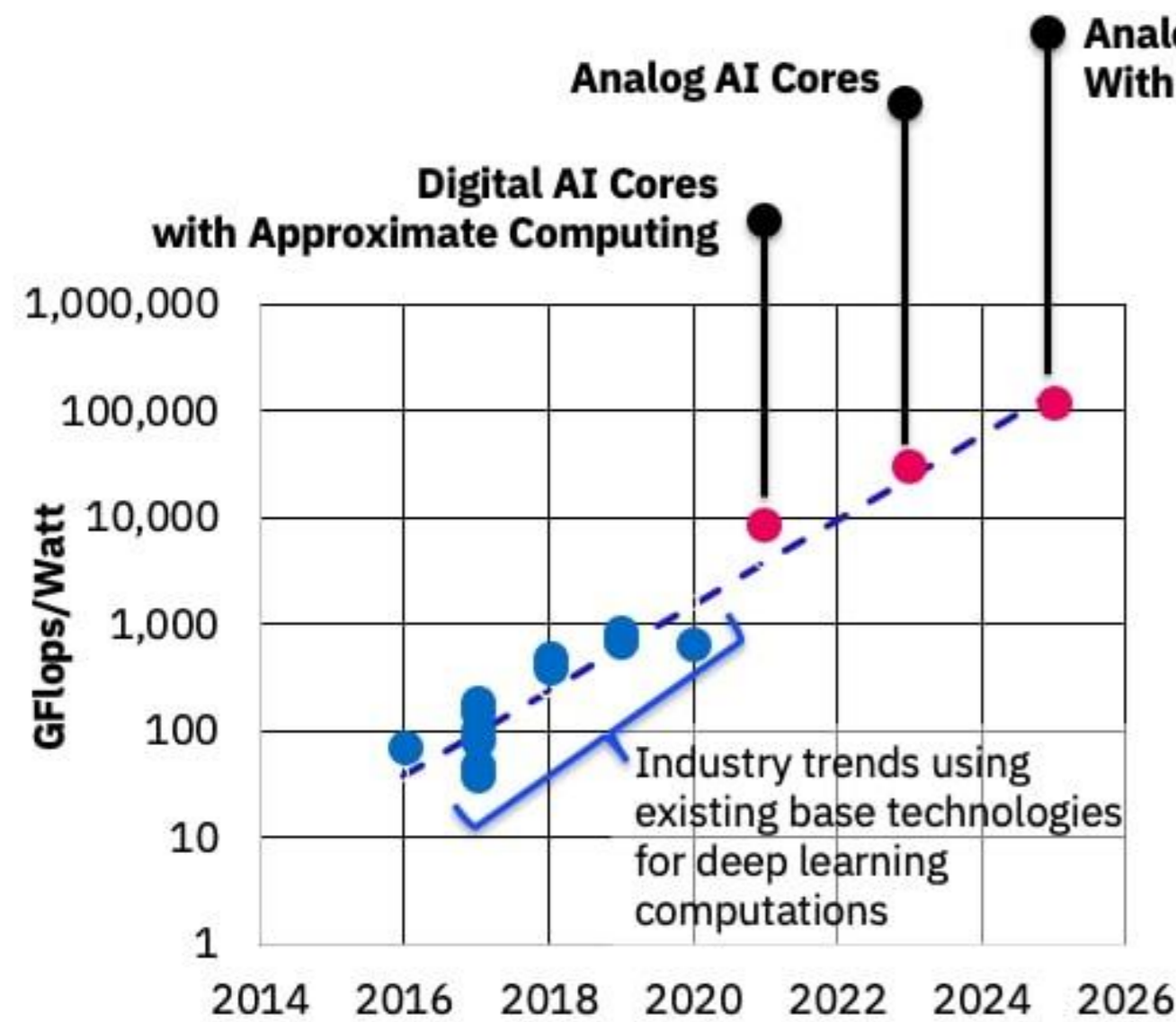
→ 'Online Learning'



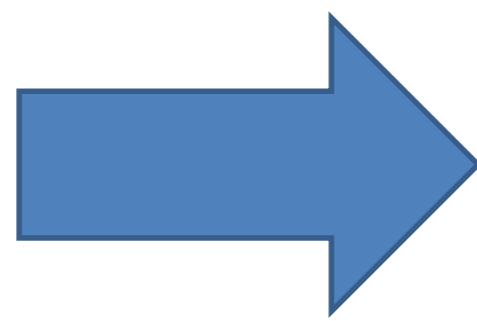
Previous slide:

Moreover, after tuning the resistance remains constant

AI Technology roadmap



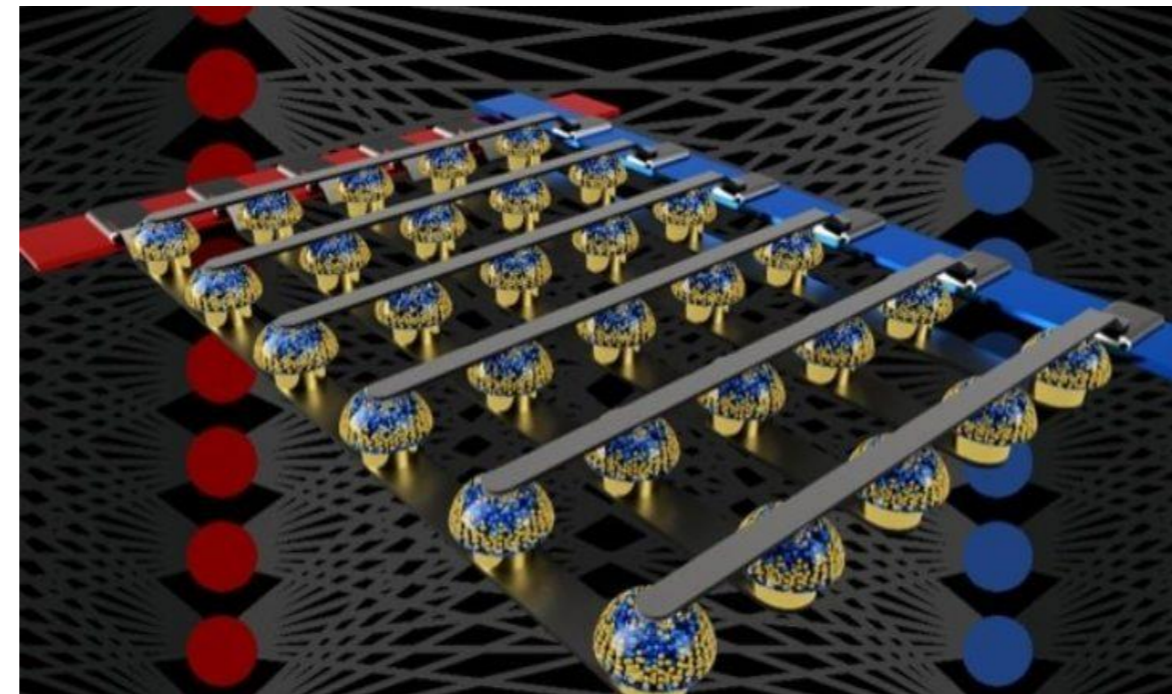
compute performance efficiency



New memristive devices required

Technology

IBM Invests \$2 Billion in New York Research Hub for AI



- Analog AI Cores
 - For the synaptic processing function
 - Apply memristive devices: Ohms law & Kirchhoff's law
 - Parallel forward inference & backward and weight update

Analog synaptic processing

	Inference	Training
Resistance	1-100 MΩ	1-100 MΩ
# Levels	100	1000
Weight set / update	To desired level	Symmetric

Previous slide: not shown in class

The road map shows several aspects:

- Traditional scaling has increased not only the compute power, but also reduced the Watt per Flop.
- Traditional scaling expected to come to an end (or may continue, red dots)
- Staying digital, but allowing for approximate computing might give an extra jump in performance
- Going analog would yield a further jump.

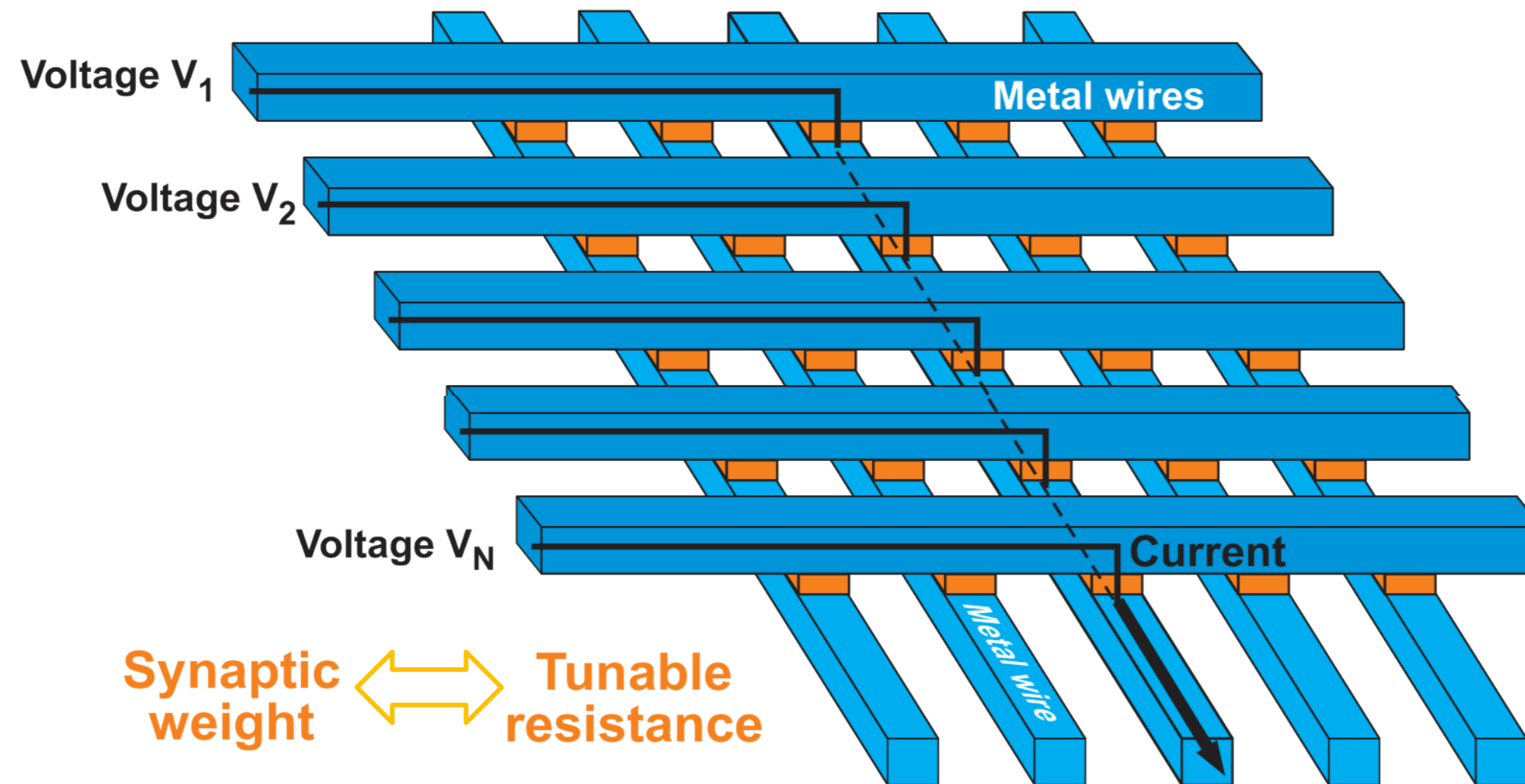
The 80-percent question again:

[] In this hardware part on memristors, at least 60 percent of the material was new to me.

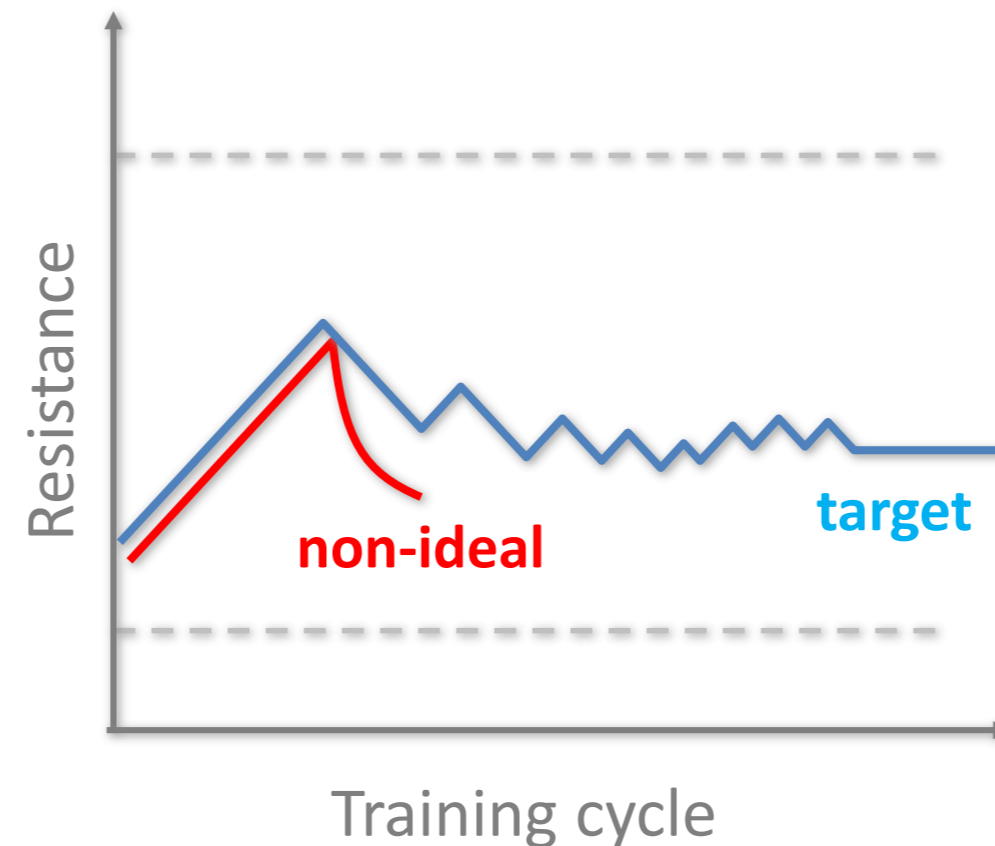
[] for this hardware part, up to here, I have the feeling that I understood at least 80 percent of the material

Analog crossbar arrays: Update for BackProp

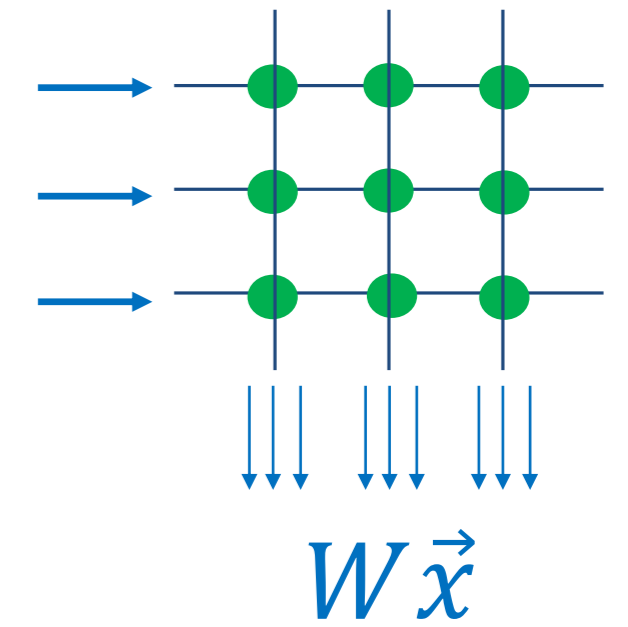
Electrical crossbar array:



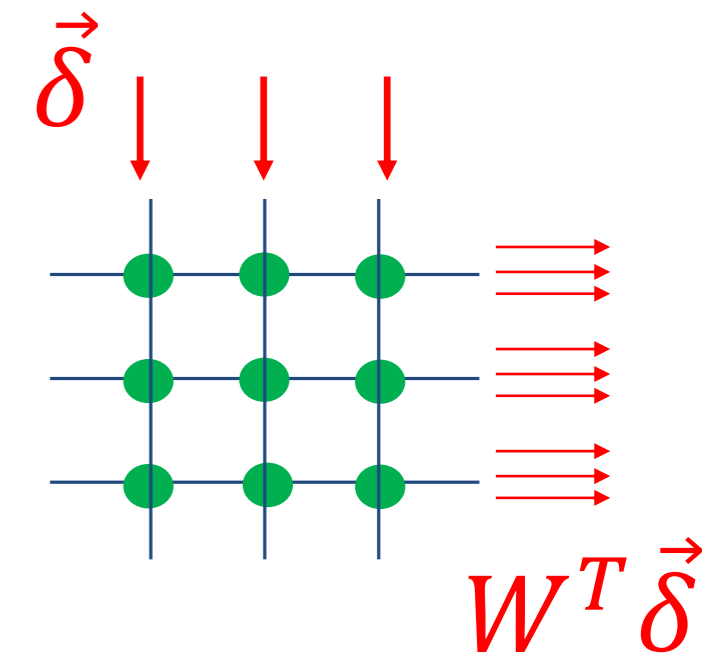
- **Weight update:** proportional to signals on row and column
 - **Symmetric** increase and decrease of weight
 - **>1bit analog levels** required
- **Physical challenge:** Identify material systems that meet these requirements



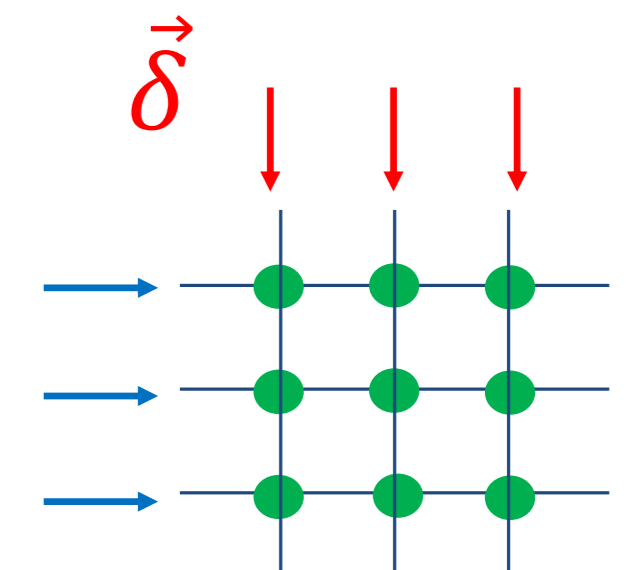
Forward propagation:



Backward propagation:



Weight update:



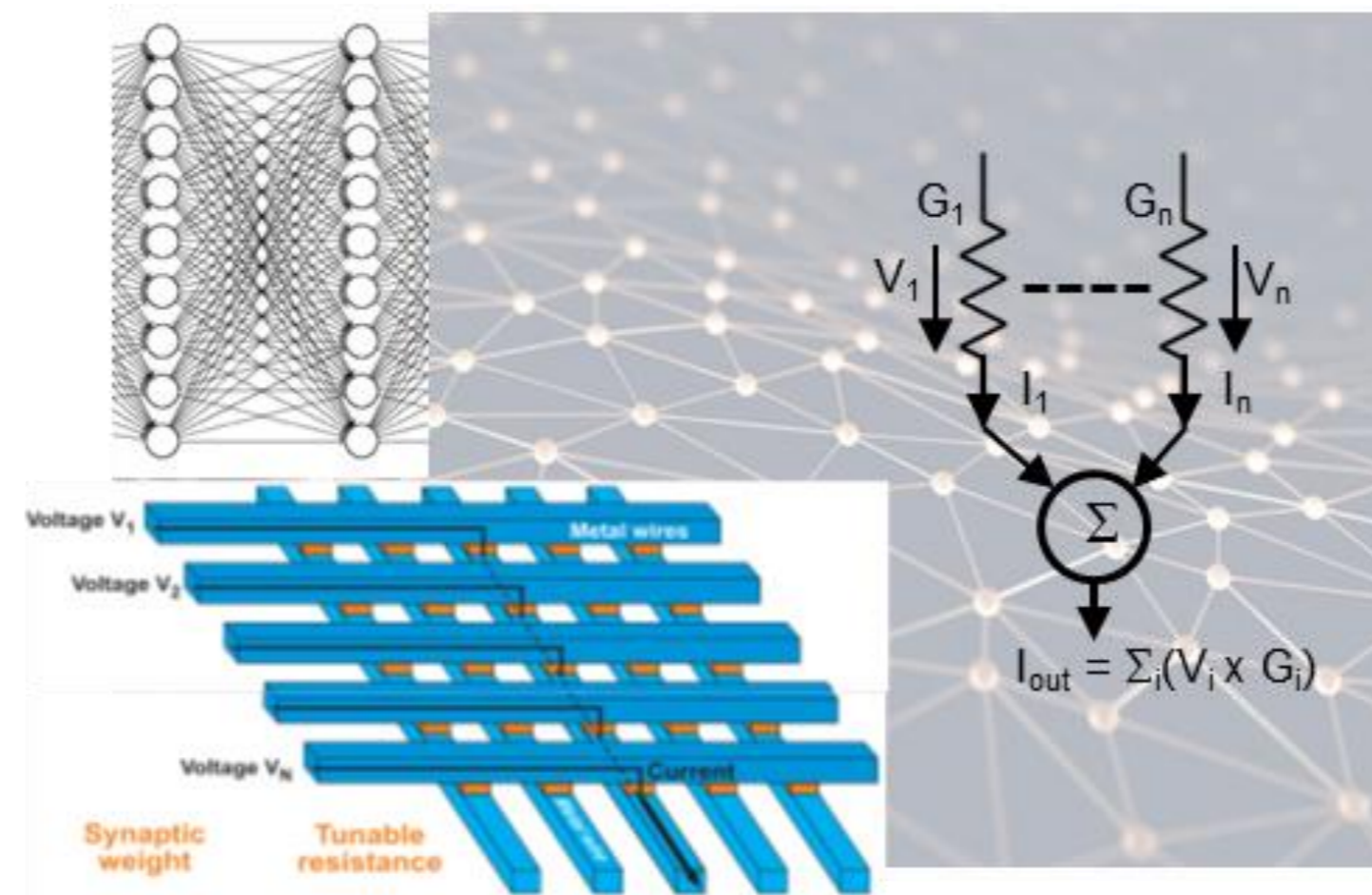
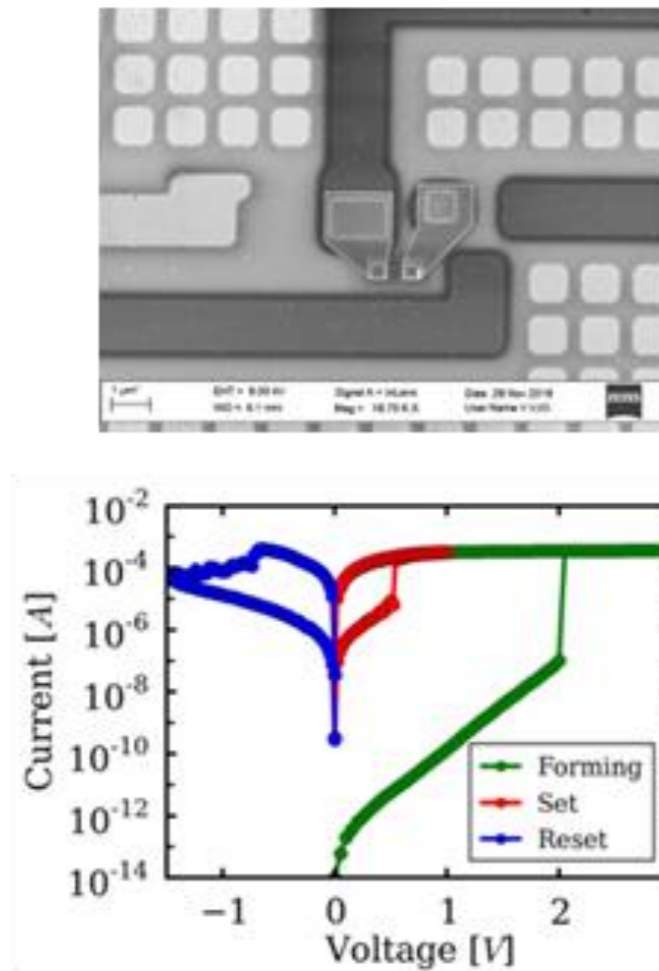
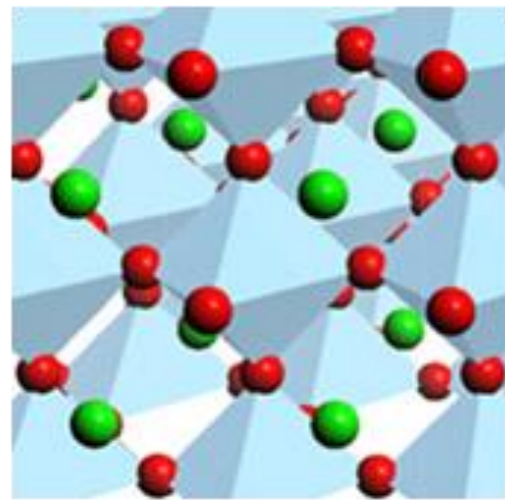
$$\Delta w_{ij} = -\eta x_i \delta_j$$

Previous slide: not shown in class.

To have an impact, local Hebbian rules are not enough.

But one can also extend these ideas to local implementations of BackPROP.

Neuromorphic – local learning rules in hardware



```
def map_string_to_vector(ListOfInputs, InputString):  
    num_inputs = len(ListOfInputs)  
    my_index = ListOfInputs.index(InputString) # find where in (index) ListOfInputs the current  
    num_vector = np.zeros([1, num_inputs])  
    num_vector[0, my_index]=1 # everything else currently is 0!  
  
    return num_vector  
  
def reshape_for_lstm(input_data):  
    # split into input and outputs  
    train X, train Y = input_data[:0:4], input_data[:0:4]
```

New Materials
and Devices

Non *von Neumann*
Architecture

Hardware –
Algorithm Interplay

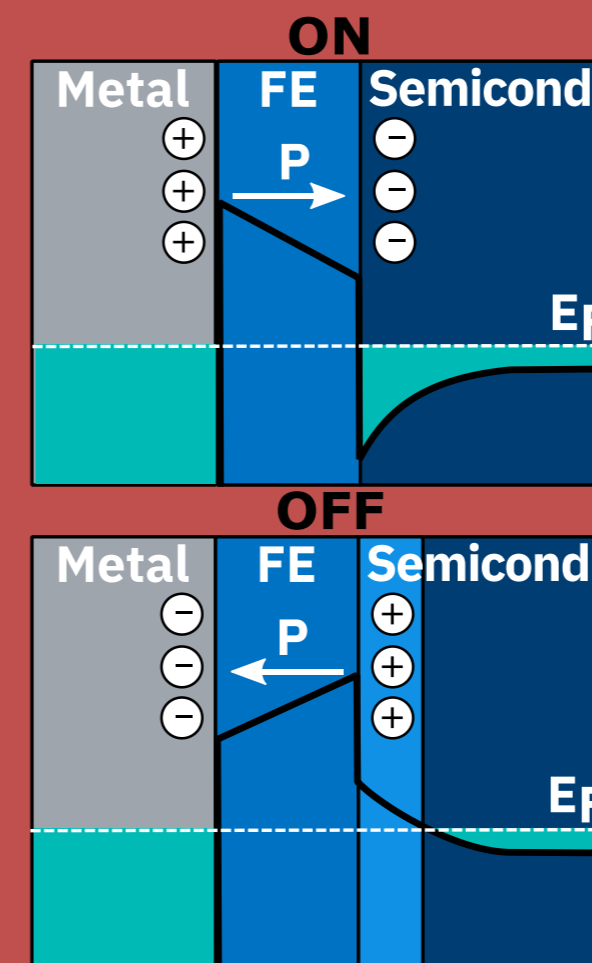
- Extension from two-factor to three-factor rules possible $O(1)$!
- Extension to (approximative) Backprop possible $O(N)$!

Literature of ferroelectrics in AI hardware

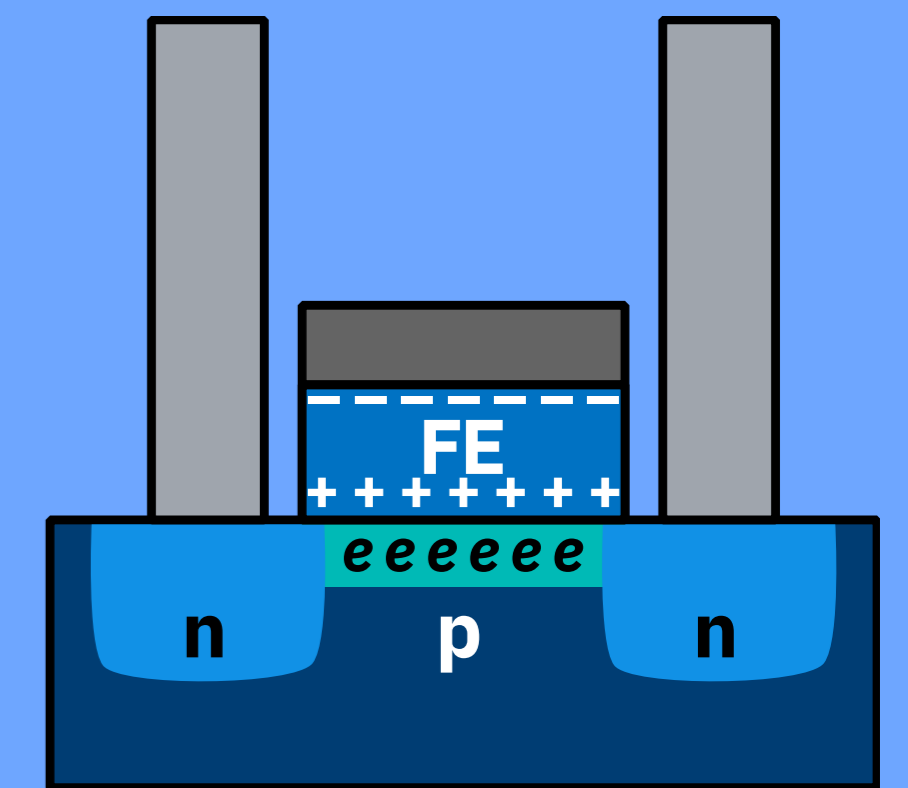
- 2011: discovery of a ferroelectric phase in HfO_2 .
- 2017: FeFET integrated in a 28nm HKMG technology (Mulaosmanovic *et al.*, VLSI 2017)
- 2018: IBM: crystallization of HfZrO_4 in the FE phase **below 400°C** (O'Connor *et al.*, APL Mater. 6, 121103 (2018))
- 2020: this work: first demonstration of a BEOL, CMOS FeFET

Ferroelectric

• FTJ



• FeFET



151

Summary

- Silicon technology remains the basis for computing devices
 - Leverage existing processes, infrastructure and know-how
 - Continuous extending of materials and function
- New computing paradigms – Neuromorphic computing - provides a path to handle unstructured data
 - Analog signal processing in crossbar arrays
 - Parallel processing of key algorithms in neural networks
 - Electrical and optical implementations

→ Extension to three-factor rule possible!

Previous slide:

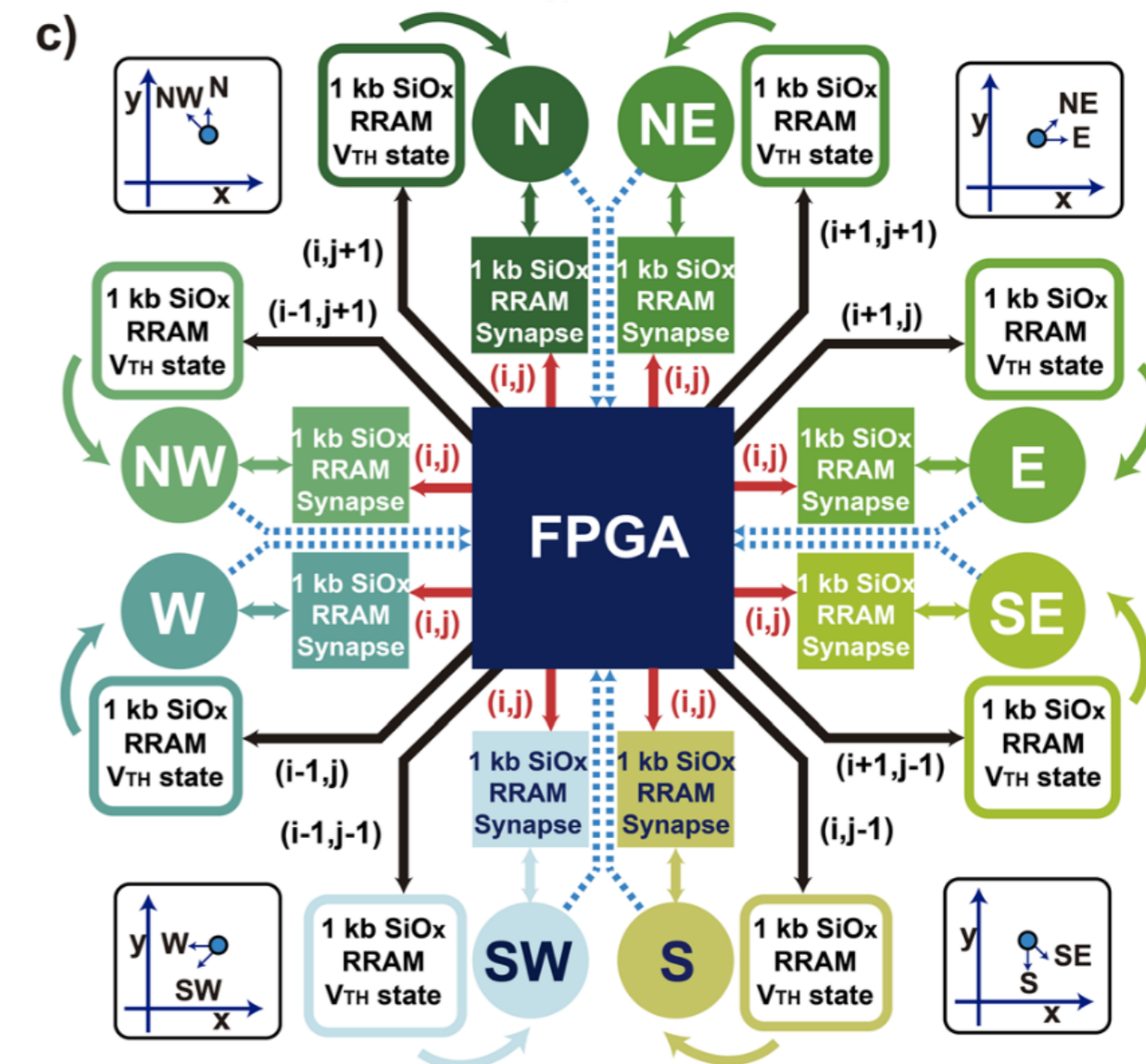
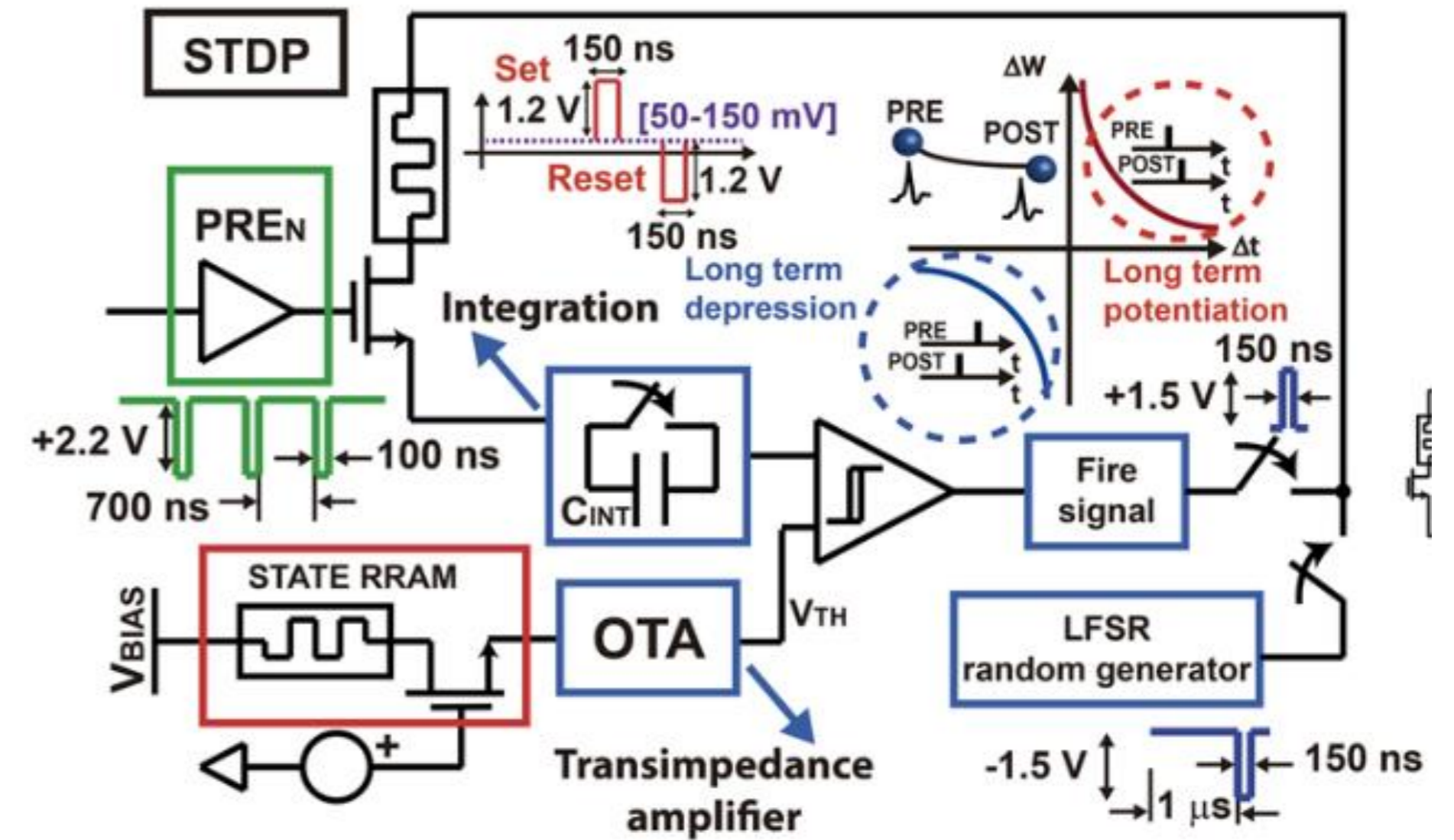
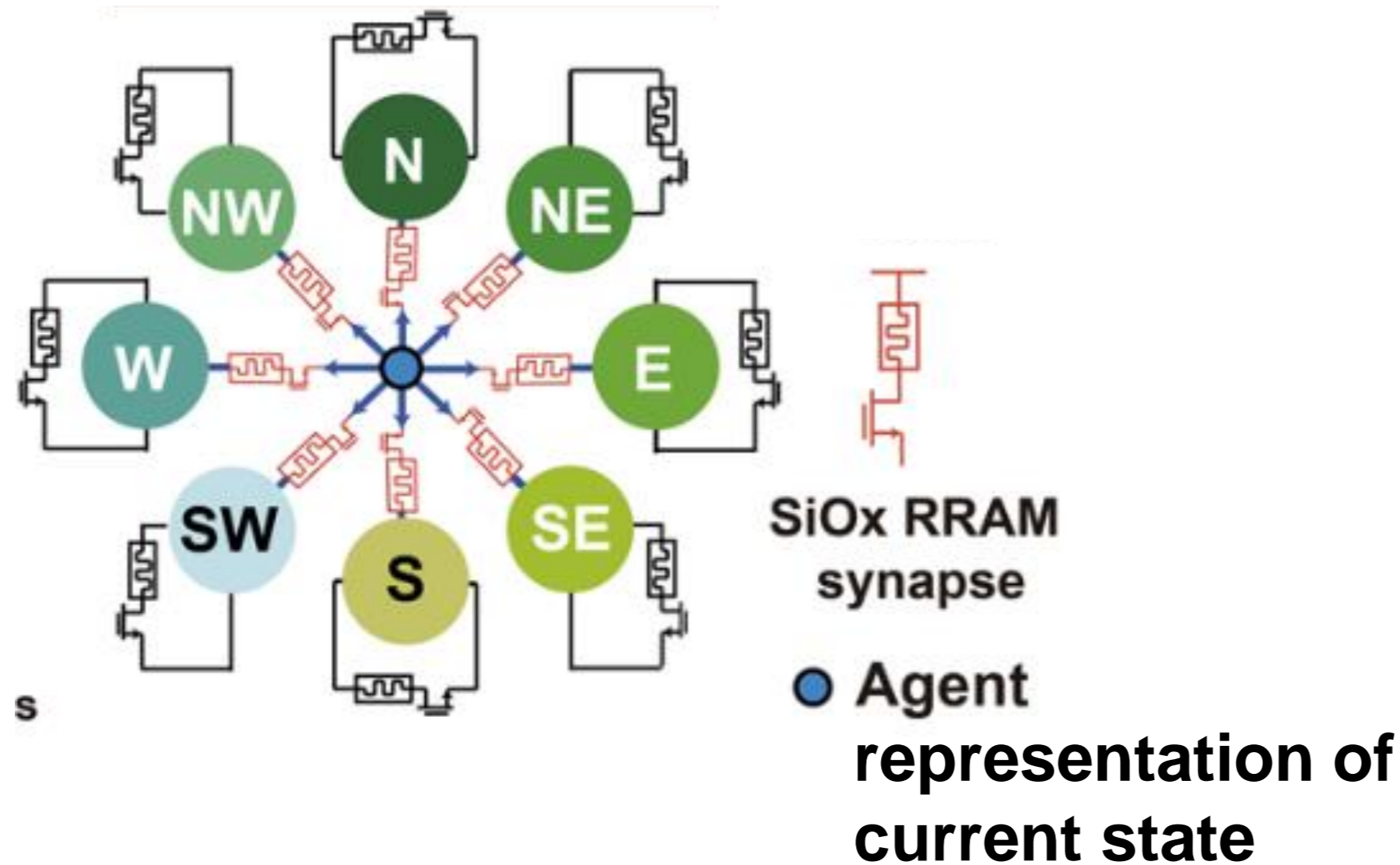
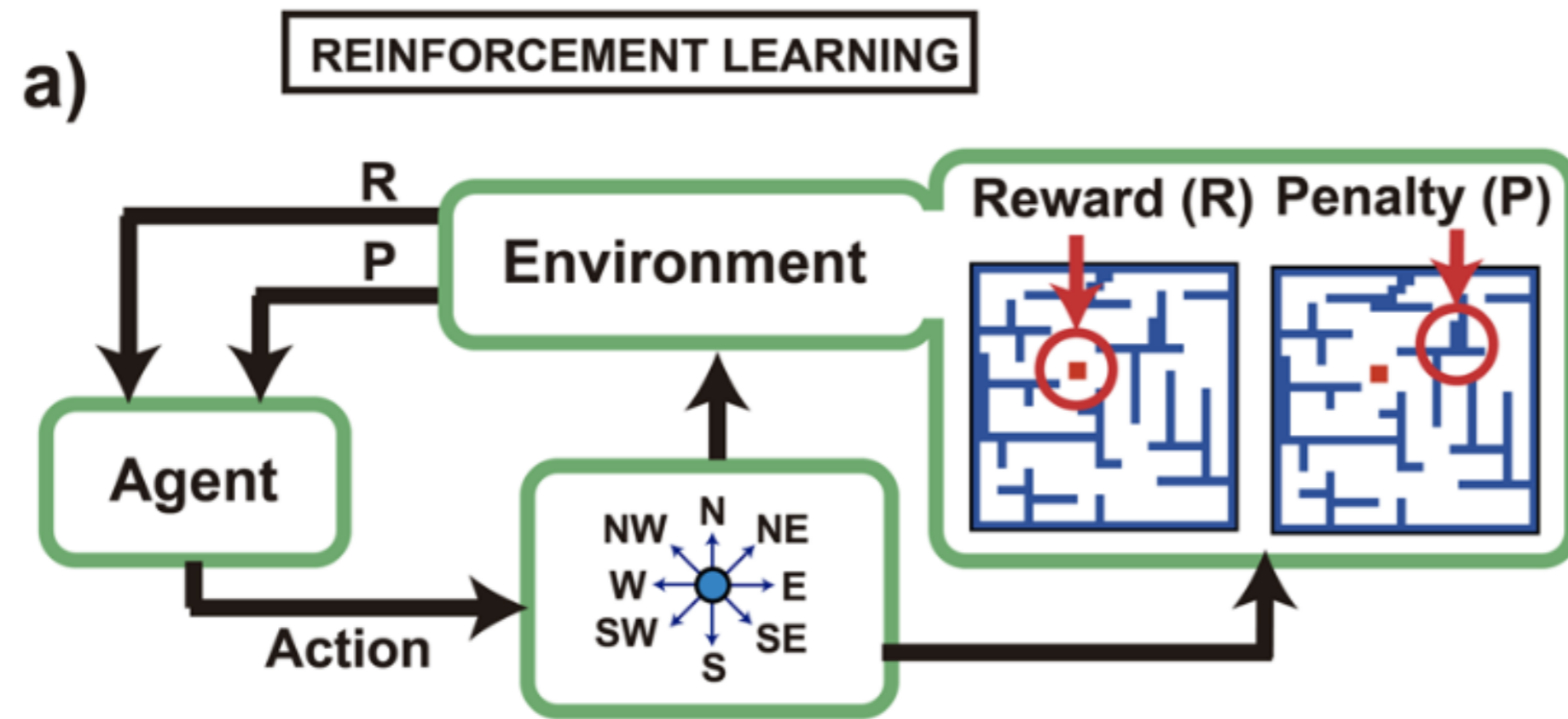
Wulfram Gerstner

EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL : **from brain-style computing to neuromorphic computing**

1. Detour: Spiking Neural Networks (SNN)
2. Example: Navigation in a Maze (Model Study)
3. Loihi Chip (INTEL)
4. Memristor chips (IBM)
- 5. Memristor chips for toy RL application**

A recent toy application: Neuromorphic/memristors



Bianchi et al, Nat. Comm. (2023)

<https://doi.org/10.1038/s41467-023-37097-5>

A recent toy application: Neuromorphic/memristors

- Very small toy application/exploratory lab research
- Digital control
- Memristors that switch from high to low resistance (binary)
- FPGA

→ Very preliminary results, very small size,
but potential for large alternative compute

→ Will this save energy?

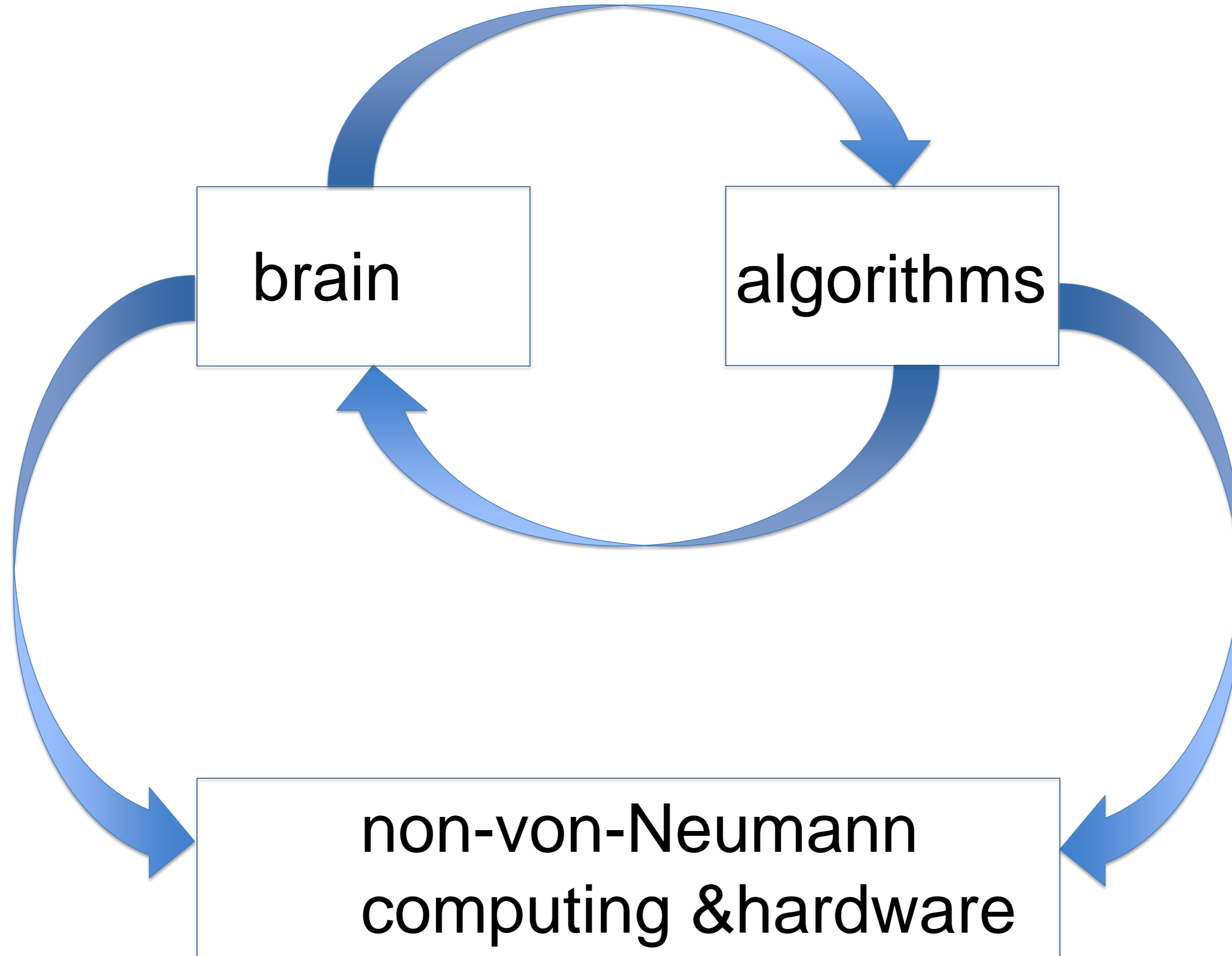
Wulfram Gerstner

EPFL, Lausanne, Switzerland

Artificial Neural Networks and RL: **from brain-style computing to neuromorphic computing**

1. Detour: Spiking Neural Networks (SNN)
2. Example: Navigation in a Maze (Model Study)
3. Loihi Chip (INTEL)
4. Memristor chips (IBM)
5. Memristor chips for toy RL application
6. **The problem of energy consumption**

Learning Rules



Energy consumption of the brain

- Sedentary humans eat and use 2500 kCal per day
- Translate to Joule → 10 000 kJ
- Brain facts: 20 percent of energy consumption of human at rest goes into the brain
- Most of it goes into synaptic signaling (spike transmission)
- Brain uses 24 – 30 Watt (5 modern light bulbs)

The power consumption of the brain is relatively low!
→ 10h of hard thinking = 0.3kWh

Previous slide.

Claim the power consumption of the brain (30W) is relatively low.

Low compared to what?

- Compare with GPU

= Compare with household power consumption.

Compare: Energy consumption of one GPU

- 300 W (RX 6800/6900 XT)
- 350 W (RTX 3080/3090)

Previous slide:

Energy consumption of one GPU

- 300 W (RX 6800/6900 XT)
- 350 W (RTX 3080/3090)

→ 10h of training an ANN on 1 GPU = 3.5 kWh

1 day of training an ANN on 1 GPU = 8000Wh = 8 kWh

4 months GPU usage → 1000 kWh

12 months GPU usage → 3000 kWh

Previous slide:

A day has 24 hours. So we multiply the power (350W) with the number of hours.

4 months have 120 days. Again a simple multiplication

The question then is: are 3000kWh per year a lot?

We need to compare with 'normal' energy consumption.

Electrical household energy consumption

Typical Swiss **electricity use in household** (fridge, TV, light)

→ **about 1000 kWh per year and person.**

- 2 persons sharing apartment = 2200kWh per year
- 4 persons sharing house = 4000kWh per year

Heating/warm water with heat pump

- 4 persons sharing house: 6000kWh per year

→ **1500 kWh per year and person**

Previous slide:

Comparison

- Brain 30W → 260 kWh per year and person
- Living in Switzerland → 2500 kWh per year and person
- GPU → 3000 kWh per year and GPU

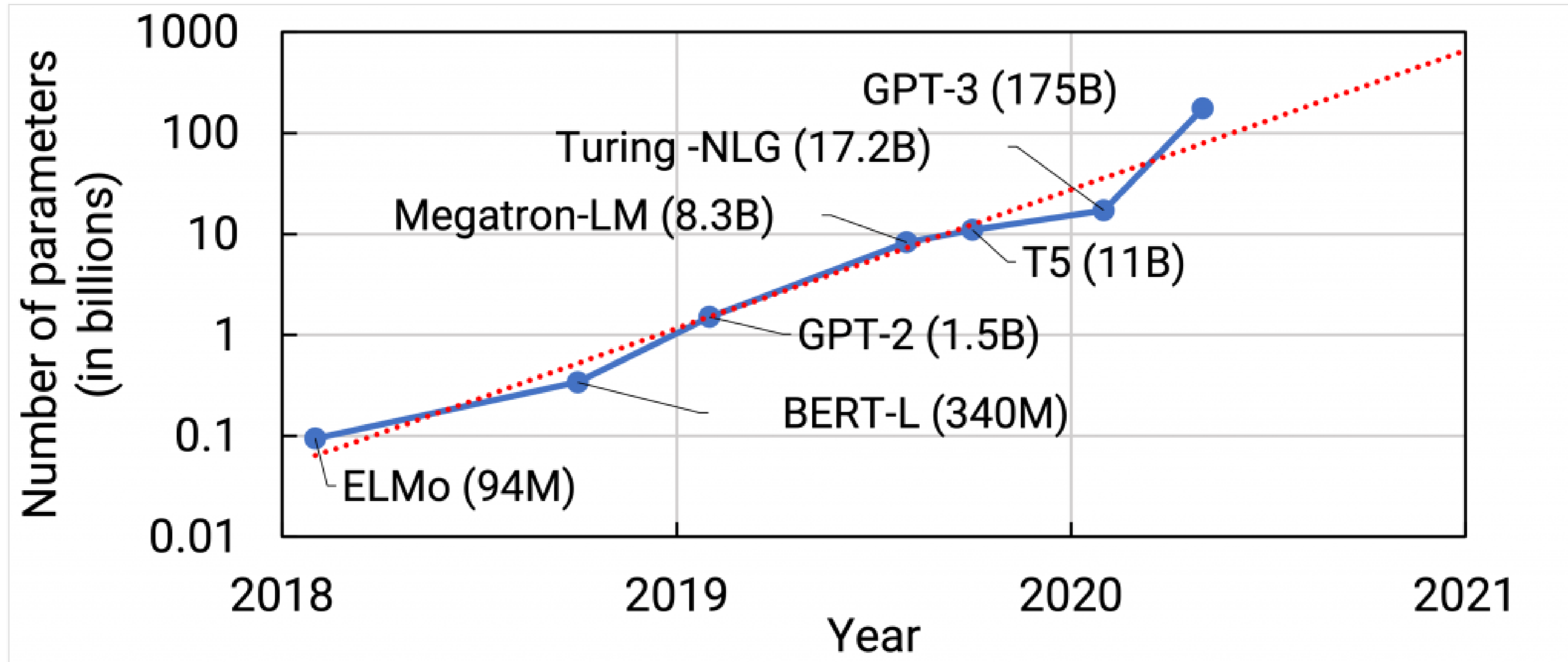
Problem!!!!

Solution? – use your machine carefully!
- think about better computer architecture!

Previous slide:

The problem is that if you your model optimization uses on average a single GPU over the year, you use more energy on your GPU than you use by living in a normal rental apartment.

The neural network *size* explosion

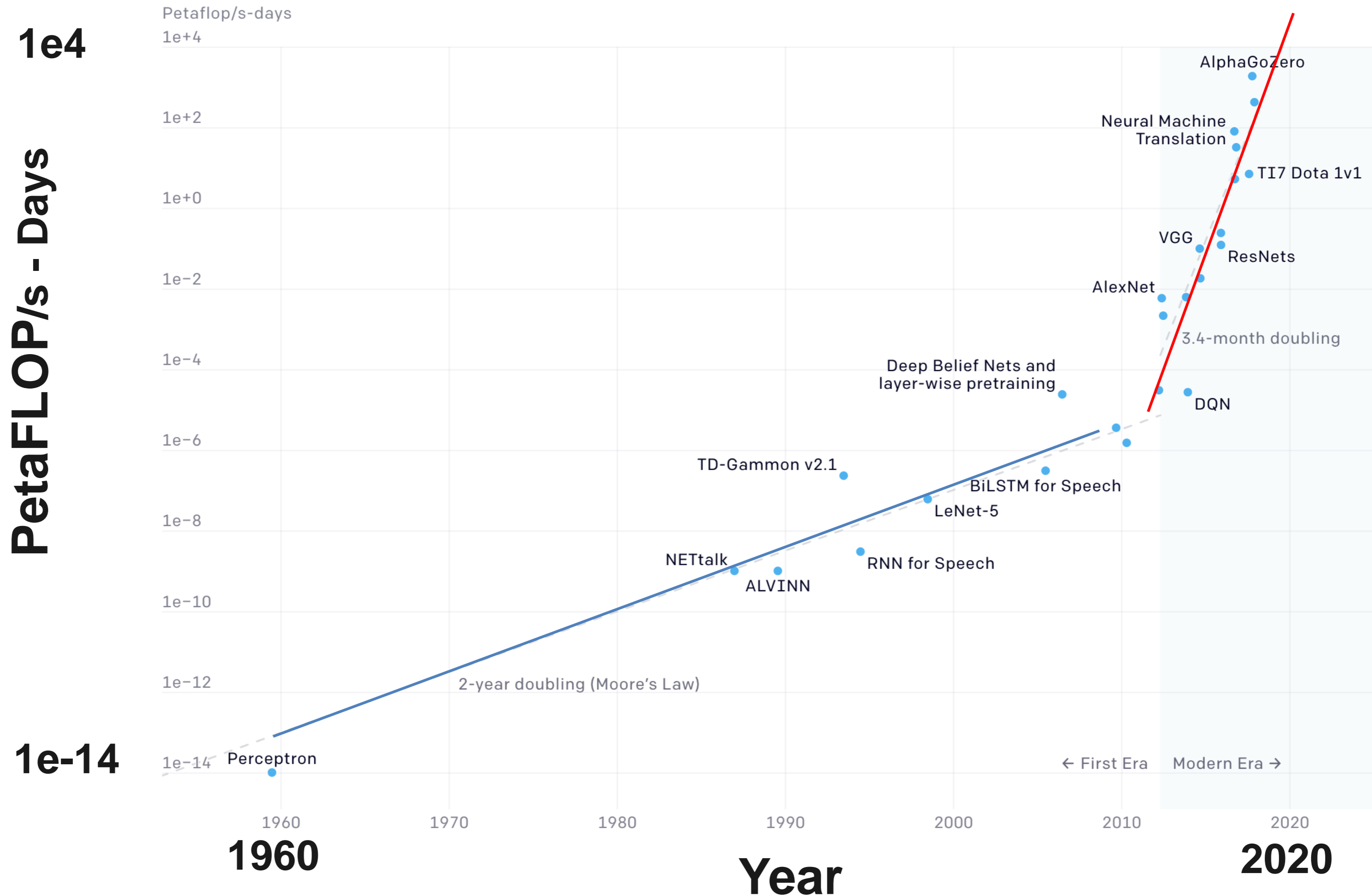


• Source:
NVIDIA

Previous slide:

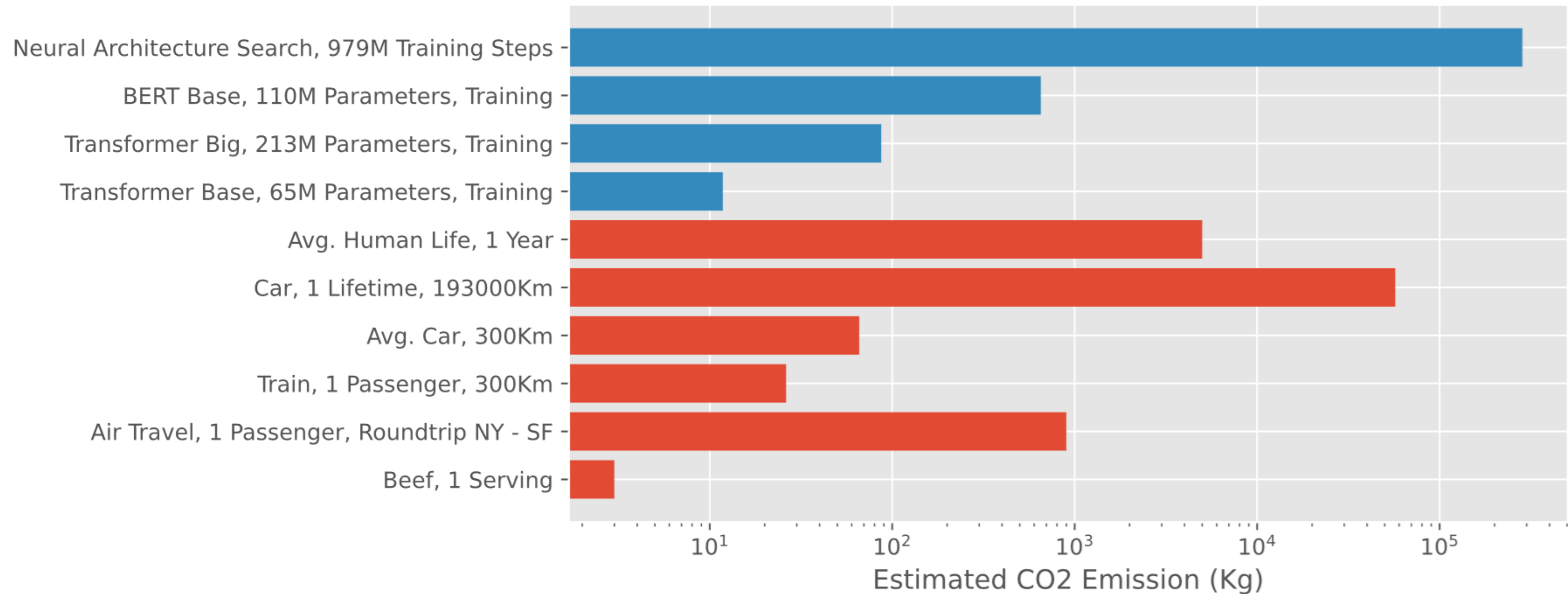
The *compute explosion*

Source:
Openai.com



Previous slide:

The *power and carbon emission* explosion



• Source: E. Strubell et al.,
arXiv:1906.02243

oil/gasoline: 1 liter = 10kWh = 2.5 kg CO₂

1000 km by plane emits as much CO₂ as 1000 km by car.

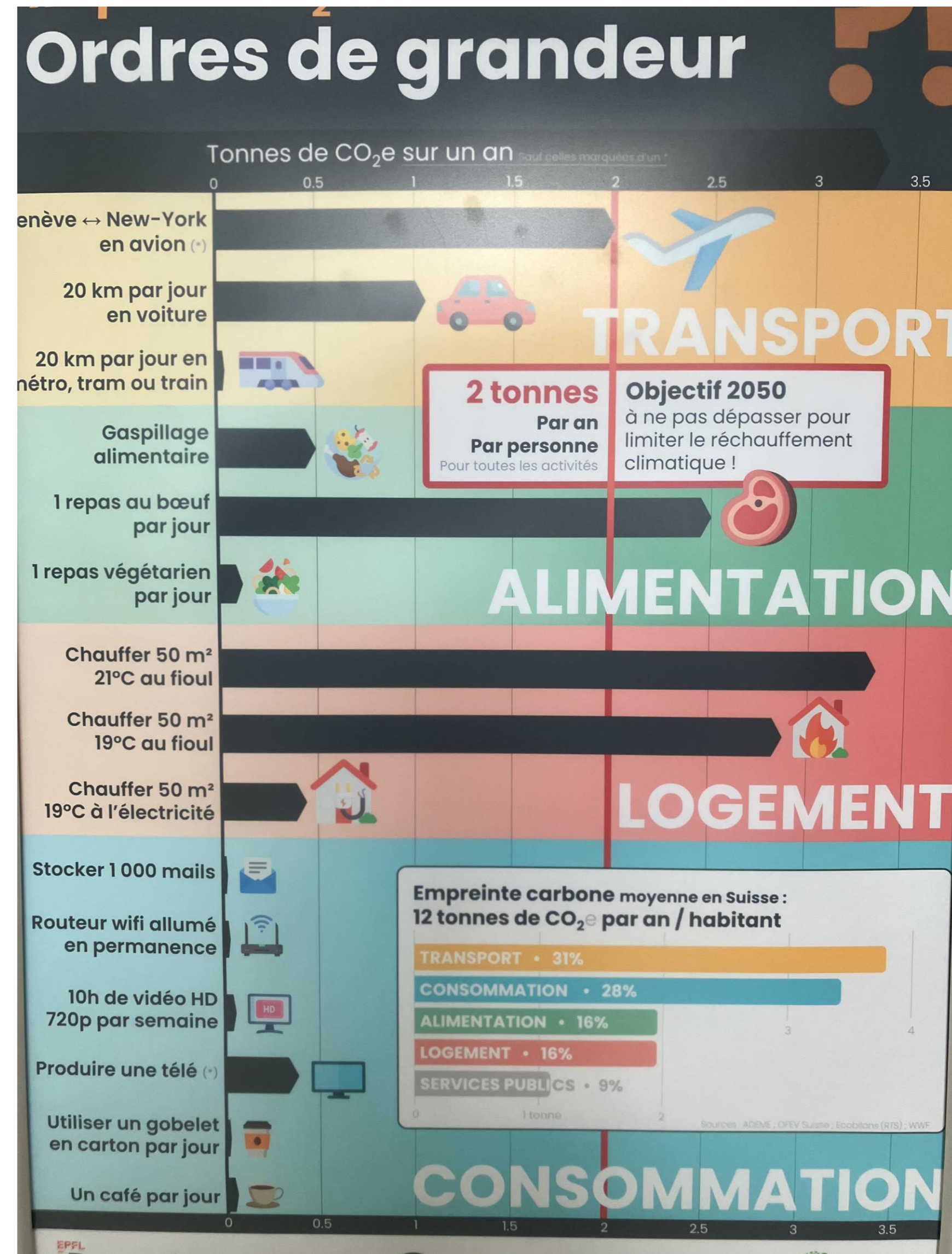
New York → Geneva = 6200km

Flight Geneva \leftrightarrow New York

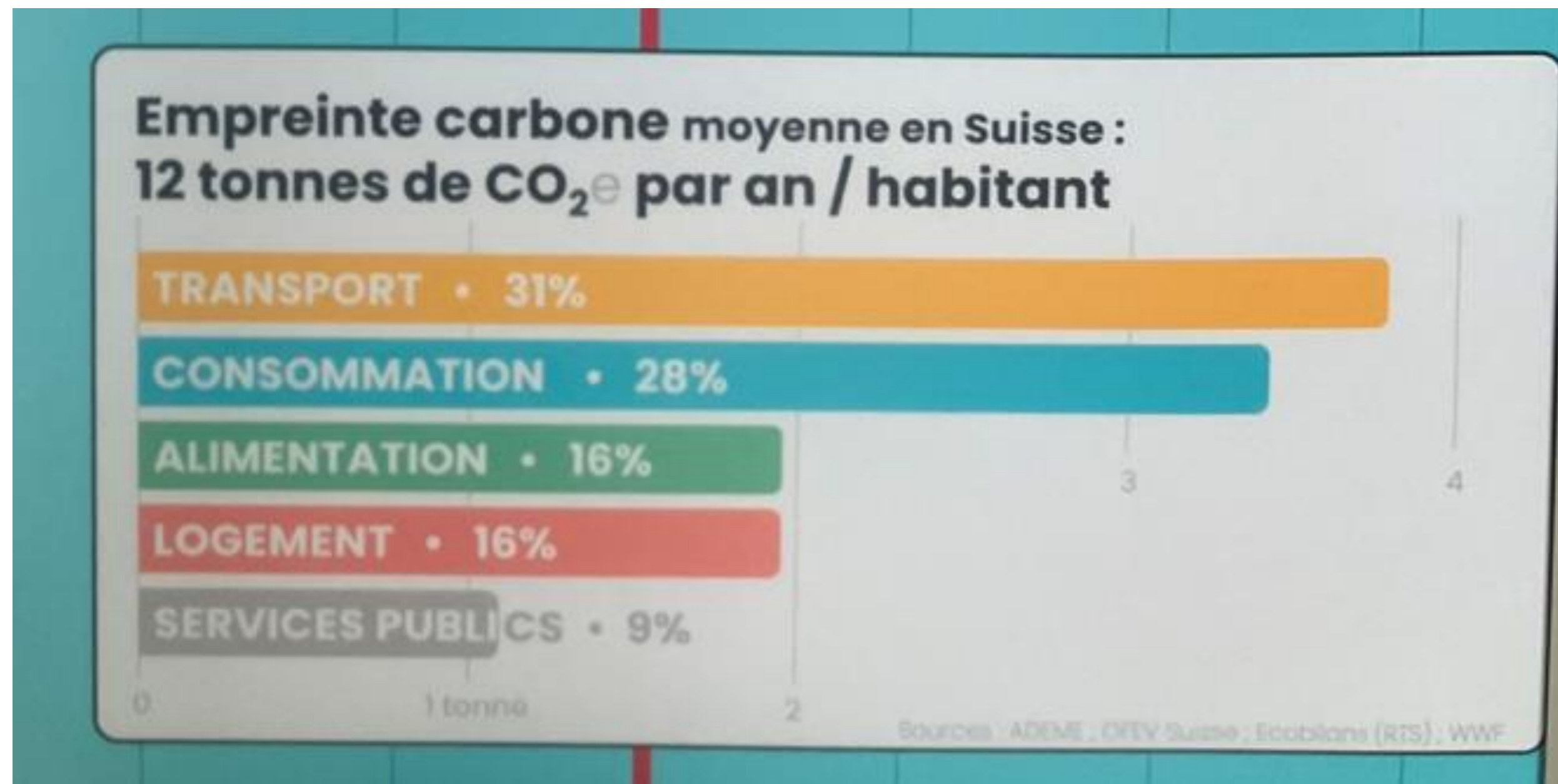
1 beef meal per day

Heating 50m²: 21°vs
(fossil fuels) 19°
heat pump

Fabrication TV screen
1 paper cup per day



Where can YOU contribute?



Le Temps, April 2024, Bien au Contraire: Decarboniser l'aviation

Aviation «verte» et publicité trompeuse

CONTRE



PIERRE KOHLER

DOCTEUR EN ÉCONOMIE, FONCTIONNAIRE INTERNATIONAL SPÉCIALISTE EN COMMERCE ET DURABILITÉ

S'il existe une industrie condamnée à devoir décroître rapidement pour respecter l'Accord de Paris, hormis l'industrie fossile et sa finance, c'est sans aucun doute l'aviation. Les experts sont unanimes, il n'y a aucune probabilité pour que d'éventuelles technologies pour «verdir» l'aviation puissent être développées et déployées à l'échelle requise dans les délais impartis. Alors, quel sort réserver à cette industrie climaticide, accessoirement symbole du «mode de vie impérial» décortiqué par les auteurs Brand et Wissen?

En 2022, les Prs Nick et Thalmann de l'EPFL avaient estimé que, pour atteindre l'objectif net zéro d'ici à 2050, tout en exploitant les avancées technologiques existantes et réalisables, l'aviation mondiale devrait réduire sa voilure de 85% et revenir à un niveau d'activité équiva-

l'aéroport de Zurich prévoit de construire une piste supplémentaire.

Malgré ces développements, les médias suisses continuent de jouer le jeu de Bertrand Piccard, chante inégalé de l'aviation «verte» et ancien marchand d'espoir, sans apporter la contradiction journalistique requise. Ce manquement est difficilement excusable, car l'homme aux 1000 techno-solutions pour la «transition verte» n'en est pas à son coup d'essai. Vingt ans après avoir créé Solar Impulse en 2004, il ne fait que récidiver en «innovant» à nouveau avec Climate Impulse.

Présenté comme un projet à la pointe de l'innovation technologique lors de son lancement,

«B
CON
NAU
AIRE»

Déc
l'a
boner
ation

Développe
rien plus ve

Interdire l'aviation? Soyons sérieux!

POUR



BERTRAND PICCARD

PRÉSIDENT DE LA FONDATION SOLAR IMPULSE

Il est plus que nécessaire de lutter contre les émissions de CO₂ produites par l'aviation, et l'assurance de certains à considérer l'aviation verte comme impossible me paraît être une simple répétition du passé.

Le préposé aux brevets de Londres avait déclaré dans les années 1860 que tout ce qui pouvait être inventé l'avait déjà été. Des spécialistes avaient calculé qu'un aéronef plus lourd que l'air ne pourrait jamais voler, et ensuite qu'il n'arriverait jamais à traverser un océan. On pensait la généralisation des téléphones portables utopique parce qu'il

faire de même pour des avions. Climate Impulse essaiera de stabiliser ce carburant vert à -253 degrés pendant neuf jours et cela aura des répercussions cruciales pour l'industrie.

La décarbonation prendra du temps, et il faut commencer par inclure la charge CO₂ dans le prix des billets, diminuer bien sûr cette frénésie de voler simplement parce que c'est bon marché, améliorer urgemment les procédures et les opérations. Mais vouloir interdire l'aviation, en plus du chaos généralisé que cela engendrerait, est complètement utopique, en Europe et a fortiori dans le reste du monde, qui ne demande qu'à se développer davantage. Soyons sérieux! Plutôt que dans l'illusion, engageons-nous dans ce que les solutions d'aujourd'hui, énergies renouvelables et hydrogène en tête, peuvent nous permettre d'ac-

Energy consumption problem (for computing) will further increase over time!

Solution? – use your machine carefully!

- think more, simulate less!
- invent better computer architecture!

Global warming is a reality!

→ Some regions no longer inhabitable/agriculture

→ Big migration waves/relocation

Solution? – tax on CO2

- reliable and predictable increase from 10cent to 100 dollars over 25 years.
- few countries start, others will follow

Thanks!

The END

... for today. There will be 2 more sessions.