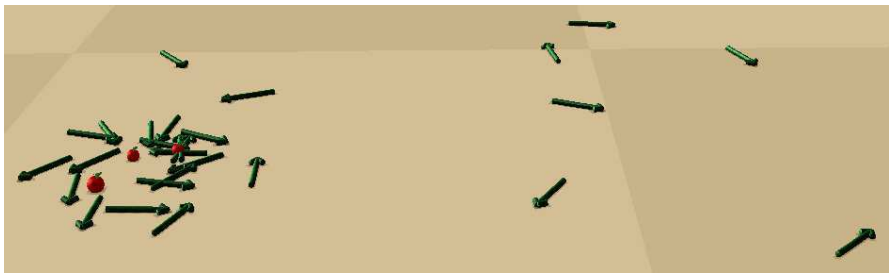

Fig. 5.7 Particle agents simulation mode. Dynamics of arrow-shaped bodies is not regarded in this experiment.

These experiments do not need true simulation of body dynamics – simple "move forward" or "turn left" commands are sufficient. In the "particle agents" mode, simple high-level methods can be used (rotating agents, setting their direction and speed) to facilitate experiment design (Figs. 5.7, 5.10, and 5.12). When bodies are not important, the simulation can be extremely simplified by making each agent a point of mass.

5.2.3 Brain

Brain (the control system) is made of neurons and their connections. A neuron may be a signal processing unit, but it may also interact with body as a receptor (sensor) or effector (actuator). There are some predefined types of neurons, for example:

- "N": the standard Framsticks neuron, which is a generalized version of the popular weighted-sum, sigmoid transfer function neuron used commonly in AI. The three additionally introduced parameters influence speed and tendency of changes of the inner neuron state and the steepness of the sigmoid transfer function. In a special case, when the three parameters are assigned specific values, the characteristics of the "N" neuron become identical to the popular, reactive AI neuron. In this case, neural output reflects instantly input signals. More information and sample neuronal runs can be found in the *simulation details* section at [19].
- "Sin": a sinusoidal generator with frequency controlled by its inputs.
- "Rnd": random noise generator.
- "Thr": thresholding neuron.
- "Delay": delaying neuron.
- "D": differentiating neuron.

It is possible to easily add custom, user-designed neurons using FramScript; an example is shown in Section 5.4.

The neural network can have any topology and complexity. Neurons can be connected with each other in any way (some may be unconnected). Inputs should be connected to outputs of another neuron (including sensors), while outputs should be connected to inputs of other neurons (including effectors). Sample control systems are shown in Fig. 5.8. Note that a single control system may contain many unconnected or independent subsystems.

Neurons can send multiple values in their output. This extension allows one to design complex neurons that provide a vector of output values, which is used, for example, in the Fuzzy and the Vector Eye neurons described later.

A section in Part I of the Framsticks tutorial [20] concerns understanding and designing control systems, and there are exercises in Part IV on the development of custom script-based neurons.

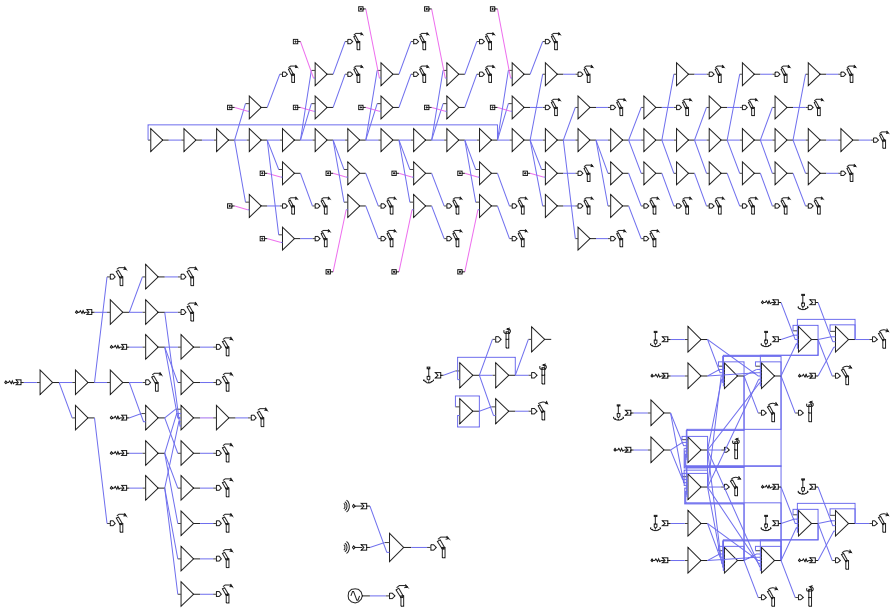

Fig. 5.8 Sample neural networks. Triangles are the standard signal-processing neurons ("N"). Receptors act as inputs (shown usually on the left side: gyroscope, touch, smell). Controlled muscles (rotating, bending) are usually on the right side. Some inputs are connected to the output of the sinus generator (⏦) or the constant signal generator (⊡). Note recurrent connections. Parallel connections are also allowed.

5.2.4 Receptors and Effectors

Receptors and effectors are interacting between body and brain. They must be connected to brain in order to be useful, but they also interact with creature's body and the world. The three basic Framsticks receptors (sensors) include "G" for orientation in space (equilibrium sense, gyroscope), "T" for detection of physical contact (touch), and "S" for detection of energy (smell); see Figs. 5.8 and 5.9.

The two basic Framsticks effectors are muscles: bending and rotating. Positive and negative changes of muscle control signal make adjacent sticks move in either direction, which is analogous to the natural systems of muscles, with flexors and extensors. The strength of a muscle determines its effective ability of movement and speed (acceleration). If energetic issues are considered in an experiment, then muscle strength affects energy consumption.

A sample framstick equipped with basic receptors and effectors is shown in Fig. 5.9. Other examples of receptors and effectors are energy level meter, water detector, vector eye, linear actuator, and thrust.

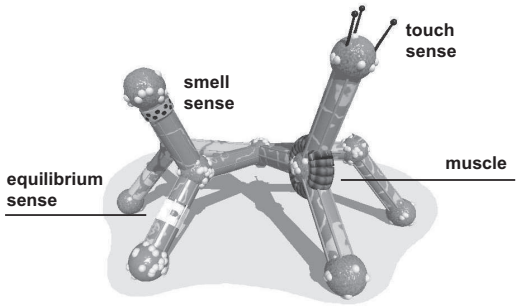

Fig. 5.9 Basic receptors (equilibrium, touch, smell) and effectors (muscles).

5.2.5 Communication

Since agents, their neurons, and the experiment logic are all script based, it is possible to implement virtually any kind of communication. Creatures, their components, and properties of these components are all accessible in script objects and can be modified by the experimenter according to their needs.

There are also dedicated objects and functions that facilitate implementation of the most common communication settings. The two basic communication objects are Channel and Signal.

- Channel objects are defined by unique names that represent either a physical medium (e.g., "light", "smell"), some abstract information ("danger", "goal"), or characterize a group of signal holders ("flock", "team_23").

- Signal objects store the value that is actually transmitted in a channel. The type of the value is arbitrary so that objects can be transmitted apart from simple values. For a signal, one can also set its power and flavor (which can be used to differentiate between multiple signals in a single channel). Signals belong to other objects – World, Creature, and Neuro – as members of their `signals` collections. World signals are stationary, while Creature signals and Neuro signals are carried by their owner.
- Functions used to receive signals can read the aggregated signal intensity from a channel (useful for physical quantities), find the strongest signal in a channel (taking into account location of transmitters and signal intensity), or enumerate all nearby signals.

Framsticks allows for visual presentation of signals in the world, which is useful both for debugging and as a part of the experiment visualization.

Communication: The Fireflies Example

In this example, Framsticks communication features are used to simulate light transmission, and two specialized neural units (the SeeLight receptor and the Light effector) are required. Agents equipped with these units are able to synchronize their flashing patterns only using local information (aggregated light intensity). See Fig. 5.10 and the Framsticks Theater show for a live demonstration.

A manually designed neural network that handles synchronization of flashing is shown in Fig. 5.11. The output of the light receptor contributes to the charging potential of neuron #6, effectively shortening the flashing cycle. Agents that flash "too late" receive most of the incoming light signal during their charging phase. This makes them charge faster and catch up with the other agents.

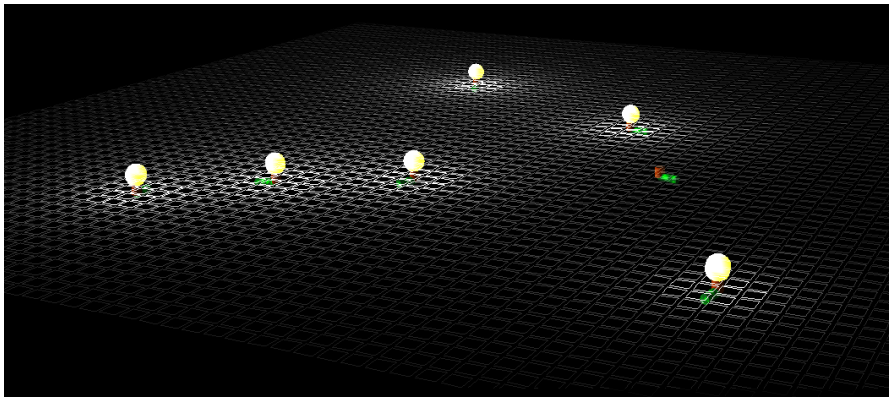

Fig. 5.10 The fireflies example and spatial intensity of the "light" signals.

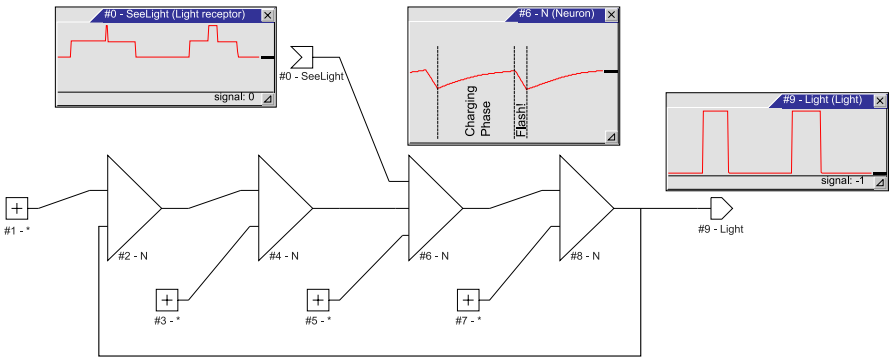

Fig. 5.11 A sample neural network that handles synchronization of flashing of a firefly.

The representation of this network in the *f1* genetic encoding (cf. Section 5.3) is as follows:

```
X [SeeLight,flavor:0] [*] [-1:2.26,6:-2,in:0.01,fo:0.01,si:1]
[*] [-2:1,-1:-0.5,si:9999,fo:1,in:0]
[*] [-2:2,-6:0.3,-1:-0.4,in:0.01,fo:0.01,si:1]
[*] [-2:1,-1:-0.5,si:9999,fo:1,in:0] [Light,-1:-1,flavor:0]
```

This genotype can be subject to evolution to obtain various flashing behaviors, depending on the fitness function used. The full source code for the the Light effector and SeeLight receptor is available in the "scripts" subdirectory of the Framsticks distribution (cf. Section 5.4 on scripting). In short, the Light effector, when created, registers one signal in the "light" channel:

```
Neuro.signals.add("light");
```

In each simulation step, this effector adjusts the power of this signal depending on its neural input value:

```
Neuro.signals[0].power=...;
```

The SeeLight receptor is very simple. In each step, it just sets its neural output to the amount of light perceived:

```
Neuro.state=Neuro.signals.receive("light");
```

Communication: The Boids Example

The implementation of boids [31] uses flexibility of the Framsticks communication to efficiently obtain the list of neighbor creatures. The data being communicated between agents are references to their own objects. The neighbor list is processed on each step by the creature handler to calculate aggregated direction of flight based on the motion of the neighbors. See Fig. 5.12 and the Framsticks Theater show for a live demonstration.

When an agent is created (the onBorn() function), it registers one signal in the "flock" channel and sets the signal value to reference its own body:

```
var signal=Creature.signals.add("flock");
signal.value=Creature.getMechPart(0);
```

To receive data, each agent in every simulation step receives a set of signals (i.e., references to neighbors within the specified range) and then can enumerate this array to perform necessary calculations:

```
var neighbors=creature.signals.receiveSet("flock",maxrange);
for(i=0;i<neighbors.size;i++) {...}
```

Fig. 5.12 The boids show in the Framsticks Theater. Creatures consist of one part only. They are assigned a visual style that uses a bird 3D model for display.

5.2.6 Environment

The world can be flat, built of smooth slopes, or built of blocks. It is possible to adjust the water level, so that not only walking/running/jumping creatures but also the swimming and amphibian ones are simulated. The boundaries of the virtual world can be one of three types:

- hard (surrounding wall: it is impossible to cross the boundary);
- wrap (crossing the boundary means teleportation to the other side of the world);
- none (the world is infinite).