PHYSICAL REVIEW E VOLUME 51, NUMBER 5 MAY 1995

Democratic reinforcement: A principle for brain function

Dimitris Stassinopoulos and Per Bak
Brookhaven National Laboratory, Upton, New York 11973
(Received 27 April 1994; revised manuscript received 10 November 1994)

We introduce a simple "toy" brain model. The model consists of a set of randomly connected, or layered integrate-and-fire neurons. Inputs to and outputs from the environment are connected randomly to subsets of neurons. The connections between firing neurons are strengthened or weakened according to whether the action was successful or not. Unlike previous reinforcement learning algorithms, the feedback from the environment is democratic: it affects all neurons in the same way, irrespective of their position in the network and independent of the output signal. Thus no unrealistic back propagation or other external computation is needed. This is accomplished by a global threshold regulation which allows the system to self-organize into a highly susceptible, possibly "critical" state with low activity and sparse connections between firing neurons. The low activity permits memory in quiescent areas to be conserved since only firing neurons are modified when new information is being taught.

PACS number(s): 87.22.As, 87.22.Jb, 82.20.Wt

I. INTRODUCTION

The mechanisms of the brain are poorly understood. The brain has billions of neurons, rather complicated themselves, each connected to thousands of other neurons. How then can one possibly generate a theory that deals with all these elements? Even to write down the map of the brain would require libraries of books. It has been said that the brain must necessarily be so complicated that it cannot possibly be understood by the brain.

Let us compare with the way we would understand a man-made object, namely a computer. First, one might suspect that the function is intimately related to the details. We would have to understand the quantum mechanical properties of the materials, silicon, etc., on which the transistors in the computer are based in order to explain how the various currents and potentials depend on each other. Actually the computer engineer could not care less about how the transistor works—it is irrelevant for his purposes. Whether the elements are of one type or another, whether they are of electrical, optical, or mechanical nature is irrelevant as long as they perform the correct function, that is, for instance, to carry out a simple logical operation such as an "AND" or "OR" logical operation on two bits.

Second, the truth could be in the complexity, or the intricacies of a vast number of interacting elements. Although one might think that the computer works because of its vast number of circuits, it is not so. The world's largest computers work the same way as the smallest pocket calculator. It simply has more storage, more processors, more input devices, etc. A computer is basically a simple device, sending numbers or bits from one location to another, and performing trivial operations with pairs of those numbers.

In order to understand the computer one has to understand the principles by which the elements are put together. The same, we argue, goes for the brain. The goal must be to understand the principles by which the neurons interact with each other, and with the environment—through muscles, sensory organs, etc. This does not mean that the study of the hardware, such as the flow of Ca^{2+} and Na^{+} ions at the synapses and axons, is irrelevant, but only that this study can be decoupled from a general study of the mechanisms of the brain. This premise forms the basis of the field of neural networks: that underlying principles for brain function can be understood with models that have simple structure.

There is one major conceptual difference between understanding the computer and understanding the brain. The computer was built by design. An engineer put together all the circuits and made it work. No engineer—no computer. However, there is no engineer around to correct all the synapses of the brain. Even more so, there is no engineer to make readjustments every time the outer world poses the brain with a new problem. One might imagine that the brain is ready and hard-wired at birth, with its connections formed through biological evolution, and all possible scenarios anticipated well ahead and coded into the DNA. This does not make any sense. Evolution is efficient, but not that efficient, certainly not for time scales much smaller than the lifetime of the individual. Indeed, the amount of information contained in the DNA is sufficient to determine the rules of the neural connections but vastly insufficient to specify the whole neural circuitry. While there is some hard-wiring—a lobster brain is different from a human brain—many important aspects of the network have to evolve during the lifetime of the individual. This means that the structure has to be *self-organized* rather than designed.

The fact that the architecture of the brain has to be self-organized puts some severe constraints on any brain model. We feel that this issue has in many cases been ignored in previous attempts to model brain function with neural networks. Conventional attractor neural network models (for reviews see Amit [1] and Hertz, Krogh, and Palmer [2]) work in two distinct modes: a learning mode where the strengths of the neural connections are computed by an outside agent, and a retrieving mode where the network recognizes input signals, i.e., provides the same pattern for several similar input patterns. More advanced models use complicated back-propagation algorithms which continuously update the connections by a computation *not performed by the neural network itself.* The organization is by design. These models have been important in the development of technologies for pattern recognition, and emphasis has been on maximizing their capacity for learning without regards to realism concerning brain function.

The importance of self-organization has previously been emphasized in the context of *reinforcement learning* models [3]. These models are more realistic with respect to brain function in the sense that there is no teacher explaining how to modify the neural connections, but only a critic telling the system whether its action were successful or not. A feedback signal from the environment is broadcast simultaneously to all elements. In most previous work on reinforcement learning, the updating of the synaptic connections is performed by back-propagation [4], or some other overseeing agent possessing prior knowledge of the problem.

There is one exception, however. Barto [5] has introduced a class of networks with "self-interested" elements which do not have access to information from elsewhere in the system. Barto's concept of self-interest coincides essentially with our use of the term "self-organization" [6]. The updating of connections is done through gradient descent methods. However, we find that regulation in such models, in particular for large networks, is inefficient. There is a weak cause-effect relationship between the variations of synaptic connections and the output.

Recently, Alstrom and Stassinopoulos [7] introduced a new class of neural networks, denoted *adaptive performance networks* which, among other things, can track a moving object. An evaluative feedback signal is sent democratically to all neurons simultaneously. The reinforcement rule depends only on the state of the neuron, and the neurons to which it is immediately connected, i.e., the neurons are self-interested. The neuron is rewarded, or penalized, whether or not its specific action had anything to do with the successful, or unsuccessful, output. In addition, there is a regulatory mechanism which keeps the output, and thus the activity in the network, minimal. This is crucial for the performance of the network.

Here, we demonstrate that adaptive performance networks can be trained to react intelligently to external sensory signals. In a fashion analogous to the behaviorist techniques used in the training of animals we present our system with a set of external signals, each of which rewards a specific action. The system learns to recognize all signals and chooses the corresponding rewarding action. The network "remembers" the correct response. In contrast to traditional neural networks, "learning" and "retrieving" are two aspects of the same dynamical process: the neurons blindly perform the same task all the time. At the individual level they do not behave differently in the learning and retrieving phases. The question as to whether the network is in a learning phase or in a retrieving phase can be judged only by an outside observer. In fact, at the global system level there is a rather abrupt spontaneous phase transition between a learning phase with slow adjustment, and a retrieving phase with fast adjustment to switching inputs.

One might wonder how the network can remain robust, i.e., why old information is not erased when the system is subjected to new information, which affects

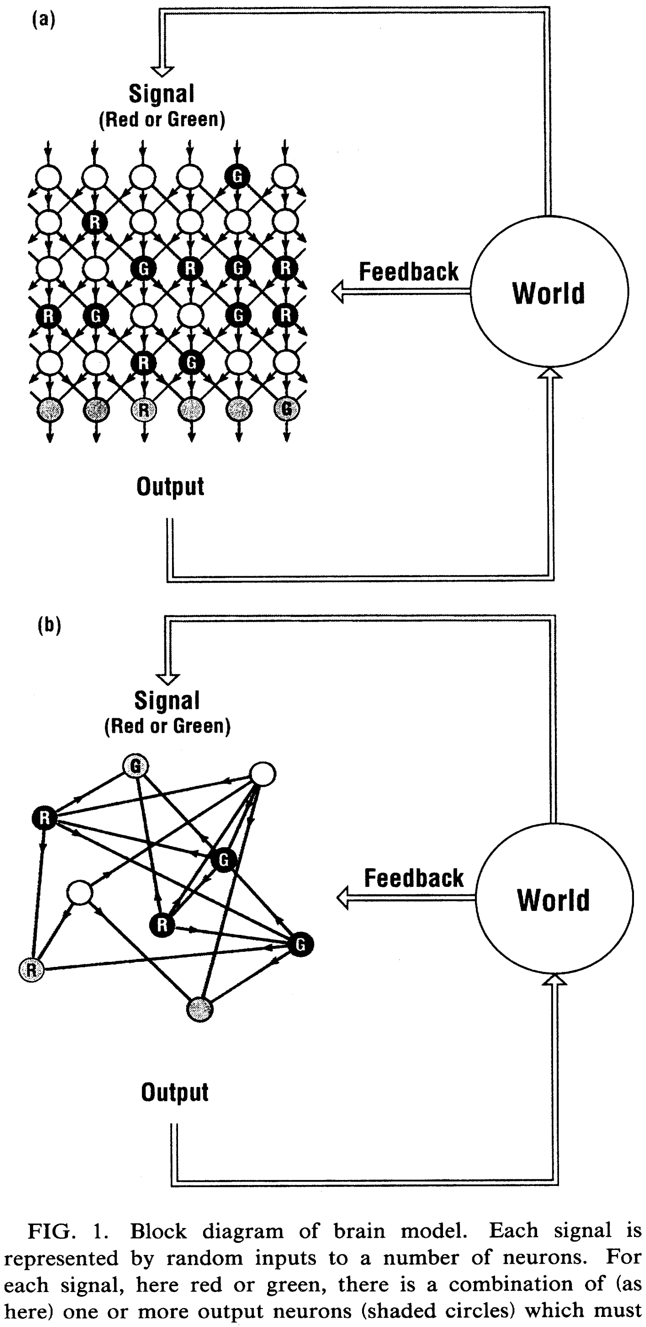


FIG. 1. Block diagram of brain model. Each signal is represented by random inputs to a number of neurons. For each signal, here red or green, there is a combination of (as here) one or more output neurons (shaded circles) which must fire in order to achieve success. The environment feeds back a signal indicating whether or not success was achieved. (a) Layered network; (b) random network.

neurons everywhere. This is precisely what is achieved by the global regulatory mechanism. In order to think clearly, one has to keep cool. If the output is too high the threshold is raised; if the output is too low, the threshold is lowered for *all* neurons. Biologically, this could be done by means of modulator chemicals. Since the reinforcement signal affects only the active neurons, information can be stored in the remaining vast areas of nonactive neurons. The network is kept near a highly susceptible critical point, or percolation threshold where the active neurons are only barely able to connect inputs with outputs. The connections are by means of linear, or fractal river networks, rather than through bulk lakes. The neurons work essentially as switches, with strong connections to output, in contrast to previous reinforcement algorithms.

To summarize: The performance of the network is based on the democratic reinforcement signal working in tandem with the global threshold regulation keeping activity low.

II. MODEL AND SIMULATIONS

The goal of any scientific theory or model is to capture the essential elements of experiments or observations in nature. Here we wish to model intelligent behavior at its simplest. To be concrete, consider the situation in which a system provides food to a monkey if the correct button is pressed. The correct button depends on whether a "red" or "green" light is on. This signal is all the information the monkey has in order to figure out which is the correct button at every instant. The action of the monkey is to press one or more buttons. The monkey learns the correct reaction after a "learning" period of trial and error. If the outside world changes, i.e., the correct buttons are switched, the monkey modifies its behavior. The animal is able to learn progressively more complicated patterns. The ability of a model to mimic the process of learning intelligent responses (leading to satisfaction) to outside signals is sometimes denoted "artificial intelligence."

We start by visualizing our model-brain in its embryonic state: a network of neurons with random connections. Sensory signals are fed into the brain randomly. The outputs, such as connections with muscle fibers, are also connected randomly. The result of the output, or action on the environment, such as whether or not food was obtained, is fed back to the "brain" through some global signal, such as a change in the level of a hormone,

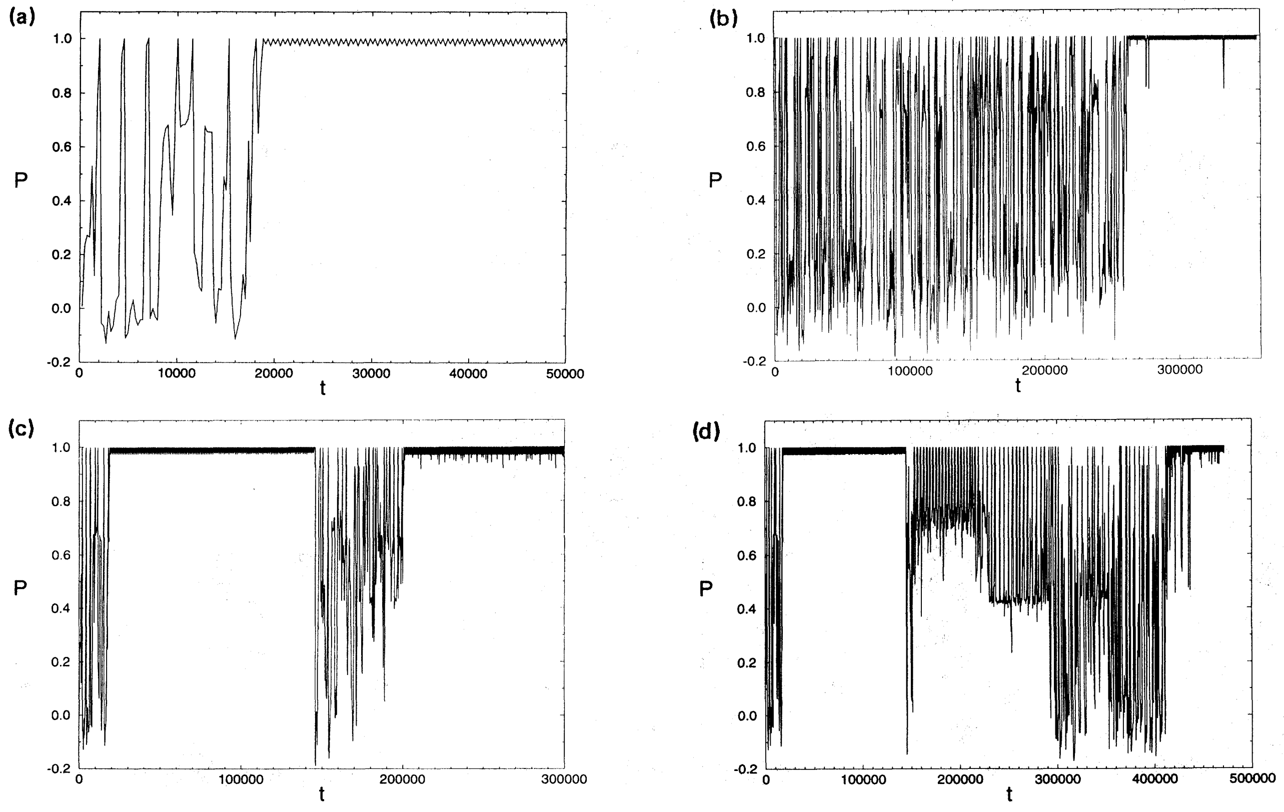


FIG. 2. (a) Performance vs time for system with two input signals which are switched every 2000 time units, or when the system is consistently successful. Consistency is checked for a period of 250 time steps. After a training period during which the network self-organizes, the system enters an "intelligent" state with fast switching between the correct outputs. (b) Same for random network; the two input signals are presented for 5000 time units unless consistent success has occurred. (c) Performance for the same system, but with 30 neurons damaged after 150 000 time steps. The system has relearned the correct response after 210 000 time steps. (d) Same system with a third input added after 150 000 steps. After a "confused" learning period, the correct output for all three inputs is learned after 450 000 time steps.

or an increase in the blood-sugar content. There is no mechanism by which the information can be fed back selectively to the individual neurons.

In this kind of picture the interplay with the environment is essential in organizing the brain's ability to explore and become more experienced, allowing it to react intelligently. In order to represent this, our model interacts with the "outer world" in three different ways (Fig. 1). There is (i) an input signal giving information about the state of the outer world, (ii) an action by the brain to the environment, and (iii) a global feedback signal indicating whether this action was successful or not in accomplishing the goal. The model is necessarily grossly oversimplified; its sole purpose is to demonstrate some simple general principles.

We have studied two network topologies: a layered one and a random one. In the latter version, both *inputs*, *outputs*, and *internal connections* are completely random. N neurons are each connected randomly with C other neurons. The neurons can be either in a firing state, $n_i=1$, or a nonfiring state, $n_i=0$. The input to the ith neuron from other neurons is $h_i=\sum_j J_{i,j}n_j$, where the summation is over the C interacting neighbors. Initially, the j's are randomly chosen in the interval $0<J<1$. The neuron fires if the input exceeds a threshold T. The interactions with the environment are implemented as follows.

(i) The sensory signal is represented by an additional contribution h' to the input signal of a number of random neurons. These various branches can be thought of as different features of the input signal such as sound, shape, color, smell, position, size, etc. Different inputs are represented by different sets of random input neurons (see Fig. 1) [8]. (ii) The output signal is the firing condition of a set of randomly selected output neurons, i_o. For each input signal the action is considered successful if one or

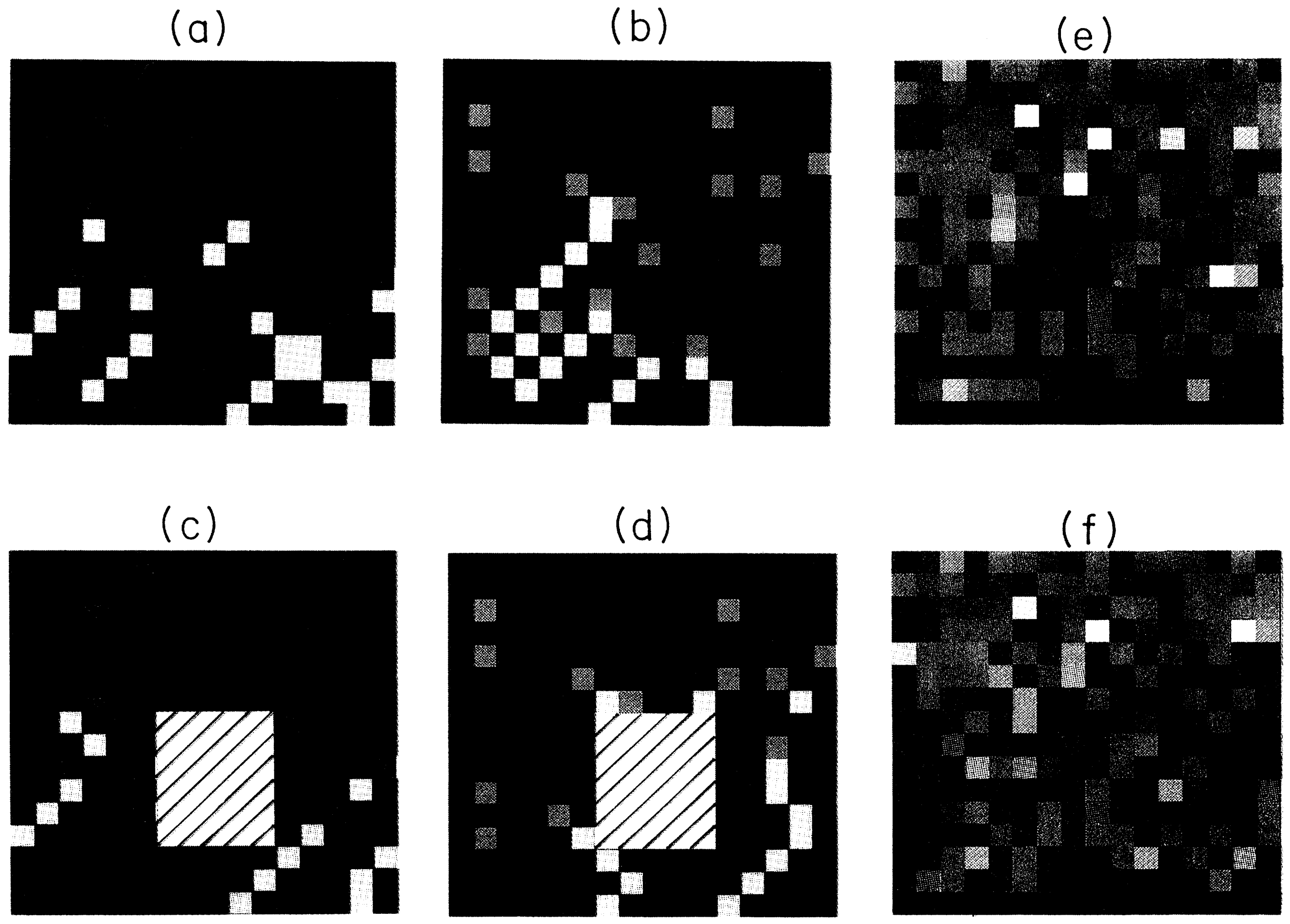


FIG. 3. Successful firing patterns in the fast switching phase. The two sets of input neurons are colored red and green, respectively. For (a) the red input, output cells 10 and 15 of the bottom row, counting from left to right, must be triggered simultaneously in order to achieve success; for (b) the green input, the output cells 7 and 12 must be triggered. The yellow squares indicate neurons which are firing for the two inputs. (c) and (d) The same as above, but in the case where the system has relearned the correct response after removal of a block of 30 neurons (shaded area). Note the difference from the original response. (e) and (f) Arbitrarily, the configurations of $J_{i,j}$'s pointing to the right were chosen for display, for the two cases a-b, and c-d. The different values are depicted with a rainbow-colored map ranging from black and dark blue for the lowest values to red for the highest.

more specific but randomly selected neurons, belonging to the set of output neurons, are all firing. (iii) If the action is successful, a positive reinforcing signal $r \ll 1$ is fed back to all firing neurons. If the action is unsuccessful a negative signal is fed back. The reinforcement modifies all connections between firing neurons, $J_{i,j} \rightarrow J_{i,j} + [rJ_{i,j}(1-J_{i,j}) + \eta_{i,j}]n_i' n_j$, where n_i' denotes the state of the ith neuron at the next time step and $\eta_{i,j}$ is a random noise between $-\eta_0$ and η_0. The outputs are normalized, $J_{i,j} \rightarrow J_{i,j} / \sum_i J_{i,j}$.

This rule differs from the well known Hebb rule in one crucial aspect: the connection is strengthened only if the simultaneous firing of two connected neurons is accompanied by a simultaneous successful output; otherwise the connection is weakened.

In addition to the input-output functions described above, the model has a global control mechanism for the overall activity [7] for the total number of firing output neurons $A = \sum_{i_0} n_{i_0}$. It is important that the output activity be kept to a minimum. If A exceeds a value A_0 the threshold T is increased, while if A is smaller than A_0 the threshold is reduced, $T \rightarrow T + \delta \,\mathrm{sgn}(A - A_0)$. Thus, if there is no output, or the output is too low, the system is "thinking," that is, its sensitivity is increased until an appropriate output is achieved. If the system is "confused," i.e., there is too much output, the sensitivity is lowered. A similar regulation takes place locally (through the normalization of $J_{i,j}$'s): when r is positive the neuron tends to activate as few as possible of its postsynaptic neurons (in our particular implementation one); if, however, r is negative it spreads its activity to as many directions as possible. Conceivably, modulatory chemicals released into the brain from elsewhere could help performing these functions for the real brain, in addition to participating in the formation of the synapses, i.e., the J's discussed above.

In the layered version, the neurons are arranged in rows, with each neuron firing to the three nearest neighbors in the next row. Inputs are random, but output neurons are those in the bottom row. At each time step the system is updated in parallel following the algorithm above. The performance P of the network is measured by the time average of the activity at the selected neurons minus the activity at the rest of the output sites. Appropriate normalization assures that best performance ($P=1$) corresponds to constant firing of the selected sites only, whereas worst performance ($P=-1$) corresponds to constant firing everywhere except at the selected sites. Figures 2 and 3 show the results for a number of different tasks.

First, the "monkey" experiment defined above was simulated. A network with 256 neurons was studied, with $C=3$ ($\eta_0=0.01$, $r=0.1$, $\delta=0.01/16$). Two input signals, each acting on 16 randomly chosen (but fixed) input neurons, were considered. For each input, a pair of output cells was defined. The input signals were switched every 2000 time steps (or when success has been achieved over 250 consecutive steps). Figure 2(a) shows the performance vs time. First, there is a period which we can identify as a learning phase in which the success rate is low and changes in a rather noisy and unpredictable manner. At some point, the network locks into a state where success is obtained very quickly in response to the switching of inputs. In this phase, the system recognizes (or retrieves) the input signals. The two phases are separated by a rather sharp phase transition. The ability to recognize emerges very suddenly. Figures 3(a) and 3(b) show the firing patterns in the fast switching phase for the two inputs, respectively. Inputs and outputs are connected with orange firing neurons. Figure 2(b) shows the performance for the random-topology case. Again a sharp, self-organized transition from a learning mode to a retrieval mode is observed.

We emphasize that the dynamics of this learning phase is distinctly different from the learning dynamics characterizing back-propagation based algorithms. To illustrate this point we have applied back-propagation to solving the simple recognition problem described above. To make the comparison more manageable we considered a smaller 4×4 system for the layered architecture and we confined our input to the top row (the output was confined to the bottom row as before). The performance P for the two learning algorithms is shown in Fig. 4. In the back-propagation algorithm, improvements in performance are gradual and smooth, in the adaptive performance algorithm they are not: a rapid improvement in performance can be temporarily interrupted by sudden dips [9]. In fact this "erratic" nature of the learning procedure makes the algorithm biologically plausible. The comparison is too sketchy to have any weight as a performance test of the two algorithms. The most important difference, of course, is that the back propagation algorithm is not self-organized in the sense that it requires outside computation.

What happens inside the network during the learning

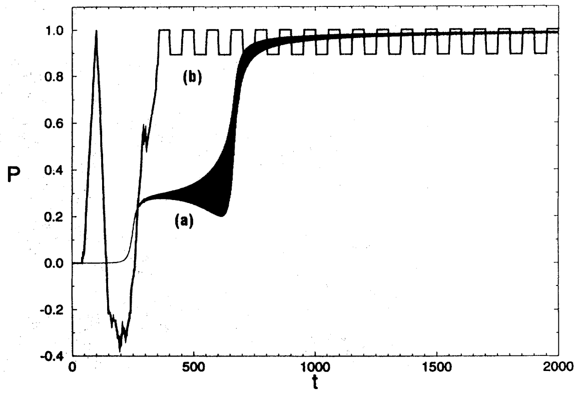


FIG. 4. (a) Performance vs time for the back-propagation learning algorithm. For the layered architecture ($C=3$) and for a 4×4 size system two input-output pairs, (1,0,0,0)-(0,0,0,1) and (0,0,0,1)-(0,1,0,0), were considered. At each trial the inputs were switched, and the system updated. The activation function $\tanh\beta h$ was used ($\beta=0.5$) and the "learning" coefficient $\eta=0.1$ involved in determining the $J_{i,j}$'s from the back-propagated errors (for details see Hertz, Krogh, and Palmer in Ref. [2]). (b) Performance vs time for the adaptive performance learning algorithm (i)–(iii). Input signals were switched every 1000 time units or if consistently successful for 50 time steps ($\eta_0=0.05$, $r=0.1$, $\delta=0.01/4$).

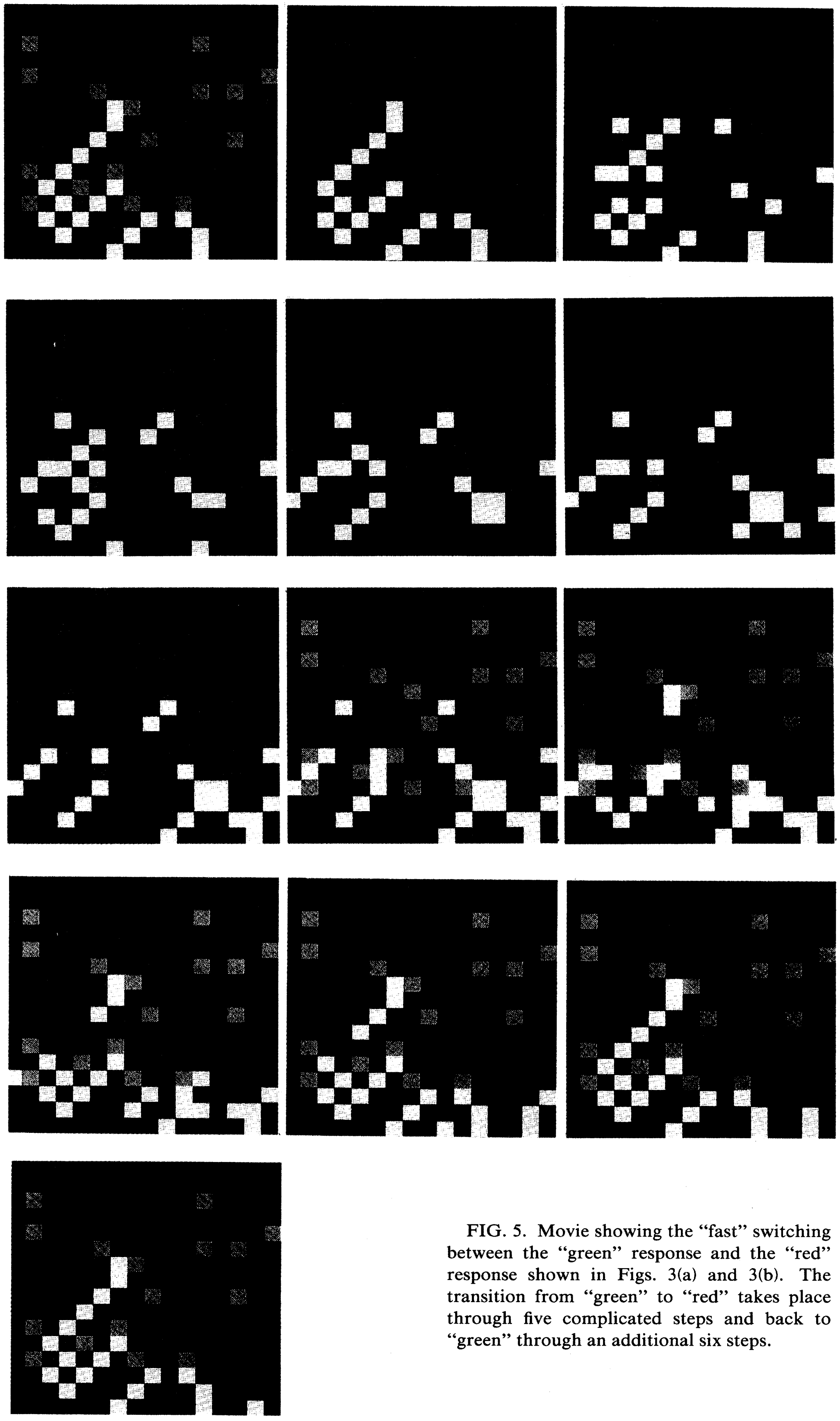


FIG. 5. Movie showing the "fast" switching between the "green" response and the "red" response shown in Figs. 3(a) and 3(b). The transition from "green" to "red" takes place through five complicated steps and back to "green" through an additional six steps.

phase? Through a complicated organizational process, the system creates internal contacts or connections between selected parts of the input signal and the correct output cell(s). The process can be thought of as the formation of a river network connecting output with input. When the output is incorrect, the river beds are raised and at the same time the flow is weakened (to minimize flooding). When the flow is correct, the river beds are deepened and as the result the pattern of the flow becomes more one dimensional.

The state of the system after completion of the learning phase cannot be calculated by means of a simple algorithm, and it is generally not reproducible in the sense that it depends on the actual random numbers. Note that there is a small activity not connected to the output. The corresponding neurons do not "know" that they are not responsible for the success, and receive, undeservedly, a reward similarly to all other firing neurons, including the ones connecting to the correct output. Figure 5 shows a movie of the switching from the "red" response [Fig. 3(a)] to the "green" response. The switching from "red" to "green" and back takes place through eleven intermediate steps. We doubt that any engineer would have come up with such a solution. If we were free to construct the network by "design" rather than by self-organization we could obviously come up with a much simpler and more efficient design. Traditional neural networks have no real dynamics taking them from one output to the next in the retrieval phase: the output is simply a unique function of the input.

The ability of fast switching is probably related to the system being tuned to a critical percolation threshold by keeping the output low. The signal is barely able to propagate through the system. The large susceptibility of the system to modifications of the input is related to the criticality. In "sand" models of self-organized criticality [10], the criticality is achieved by keeping the input rate, rather than the output, low.

Figure 2(c) also shows the response to "damage" done to the network [Figs. 3(c) and 3(d)]. After $\sim 150\,000$ time steps, a block of 30 neurons was removed from the network. After a transient period [Fig. 2(c)], the network has relearned the correct response, carving new connections in the network. In other words, instead of using some features of the input signal the system learns to use other features. Think of this as replacing "vision" with "smell." The memory is distributed and robust, as it should be in order to represent real brain function.

Figure 2(d) shows the situation where a third input (and corresponding pair of output cells) was added after the first two responses had been learned. After a transient period where the system is confused and the success rate is low, the network eventually learns the three appropriate responses.

The complexity of the switching behavior of the system hints to the fact that it is not only the well developed "river beds," where most of the activity takes place, that are important for the function of the network, but also the relatively silent regions in between. Evidence of this can be seen in the rather complicated landscape of the $J_{i,j}$'s. Figures 3(e) and 3(f) show the strengths of the connections with a color code. The landscape is strikingly rugged. One might have expected well-carved riverbeds and isolated switches. Where it would seem that this configuration is more efficient from an engineering point of view, it is not compatible with the self-organization process since it is unlikely for the system to find such rare optional configurations on its own. The ruggedness, with many neurons working near the threshold, makes the network susceptible to changes.

Of course, our "pocket-calculator" brain model is not a serious model of any real brain. Its sole purpose is to demonstrate that aspects of brain function can be modeled with a structure that has a minimum of complexity, as is imperative for a real biological brain. A brain working according to these principles could be a relatively simple organ without much structure. Little information is needed to construct the simple network with essentially arbitrary connections.

[1] D. J. Amit, *Modelling Brain Function: The World of Attractor Neural Networks* (Cambridge University Press, Cambridge, 1989).

[2] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation* (Addison-Wesley, Redwood City, CA, 1991).

[3] A. G. Barto and P. Anandan, IEEE Trans. Syst. Man, Cybern. **15**, 360 (1985).

[4] Based on the partial information provided by the critic a target pattern is determined and the output-weight errors computed. The rest of the weights can then be updated by back-propagating this error-signal through the network (see Ref. [2]).

[5] A. G. Barto, Human Neurobiology **4**, 229 (1985).

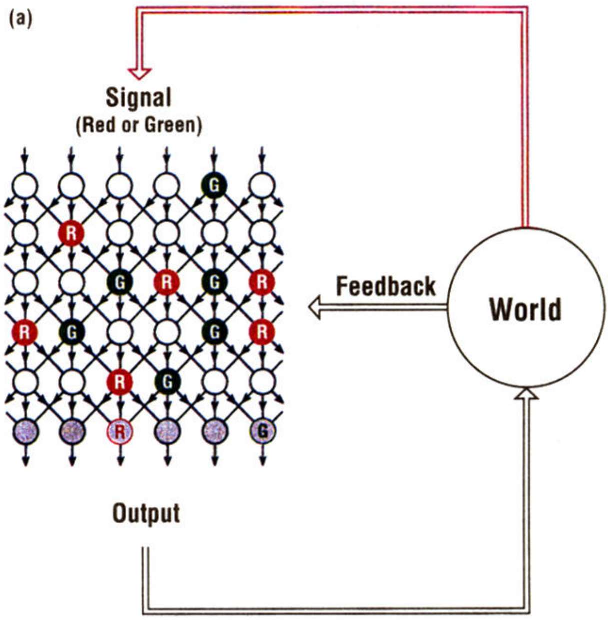
[6] Kohonen [T. Kohonen, *Self-organization and Associative Memory* (Springer, Heidelberg, 1984)] has introduced what he calls "self-organizing" maps in which there is no feedback or supervision whatsoever. We do not see how to utilize such maps to model intelligent behavior as defined here.

[7] P. Alstrøm and D. Stassinopoulos, Phys. Rev. E **51**, 5027 (1995).

[8] The original AS model [7] is "blind" in the sense that it operates with a fixed input signal at the upper layer.

[9] This aspect of the dynamics is more apparent in Fig. 2(a) where we consider a larger system: the bigger the system the larger the number of paths needed to be explored and the larger the number of necessary adjustments or switchings between path configurations. Nevertheless we find that our adaptive performance algorithm deals relatively well in situations where more than a few intermediate neurons are involved. For instance, when the size of the network was doubled, 8×8, and for the same parameter values as in Fig. 4, the adaptive performance algorithm solved the problem in ~ 5000 time steps while the back-propagation required $\sim 500\,000$ trials.

[10] P. Bak, C. Tang, and K. Wiesenfeld, Phys. Rev. Lett. **59**, 381 (1987).



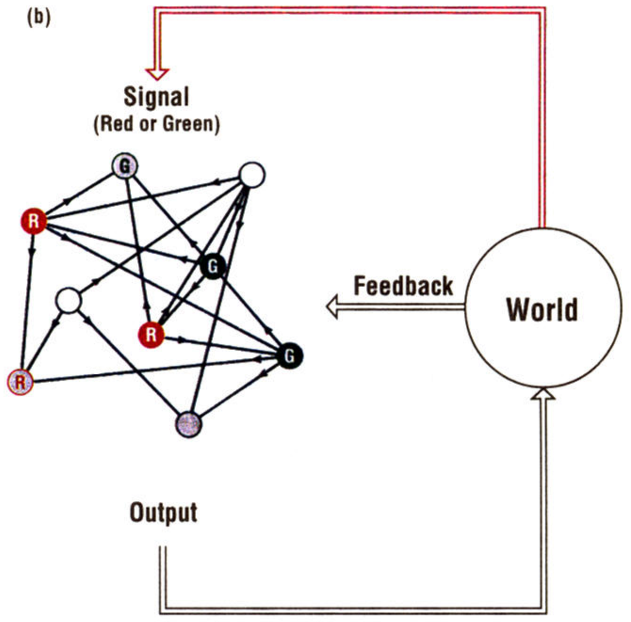


FIG. 1. Block diagram of brain model. Each signal is represented by random inputs to a number of neurons. For each signal, here red or green, there is a combination of (as here) one or more output neurons (shaded circles) which must fire in order to achieve success. The environment feeds back a signal indicating whether or not success was achieved. (a) Layered network; (b) random network.

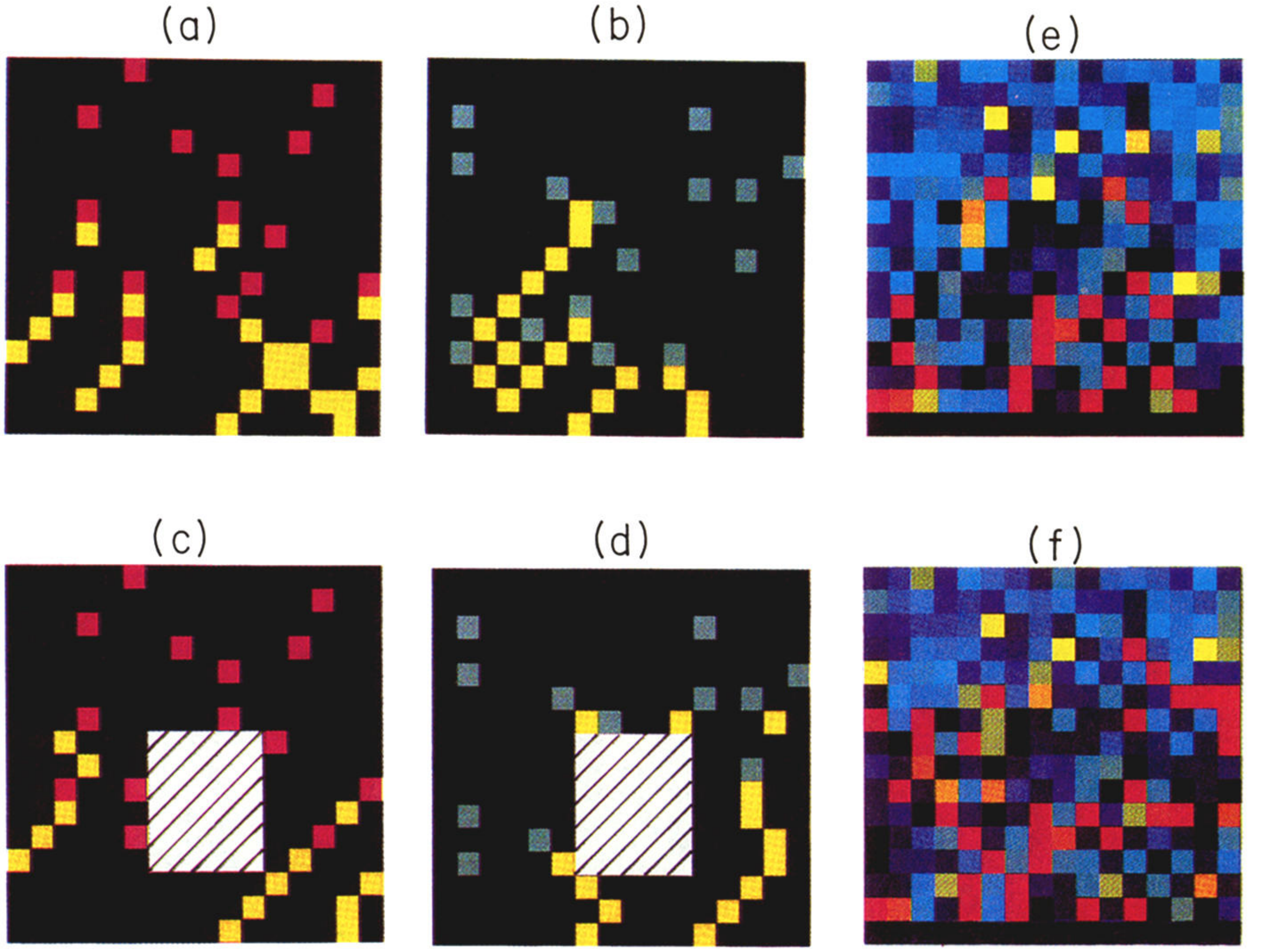


FIG. 3. Successful firing patterns in the fast switching phase. The two sets of input neurons are colored red and green, respectively. For (a) the red input, output cells 10 and 15 of the bottom row, counting from left to right, must be triggered simultaneously in order to achieve success; for (b) the green input, the output cells 7 and 12 must be triggered. The yellow squares indicate neurons which are firing for the two inputs. (c) and (d) The same as above, but in the case where the system has relearned the correct response after removal of a block of 30 neurons (shaded area). Note the difference from the original response. (e) and (f) Arbitrarily, the configurations of $J_{i,j}$'s pointing to the right were chosen for display, for the two cases *a*-*b*, and *c*-*d*. The different values are depicted with a rainbow-colored map ranging from black and dark blue for the lowest values to red for the highest.

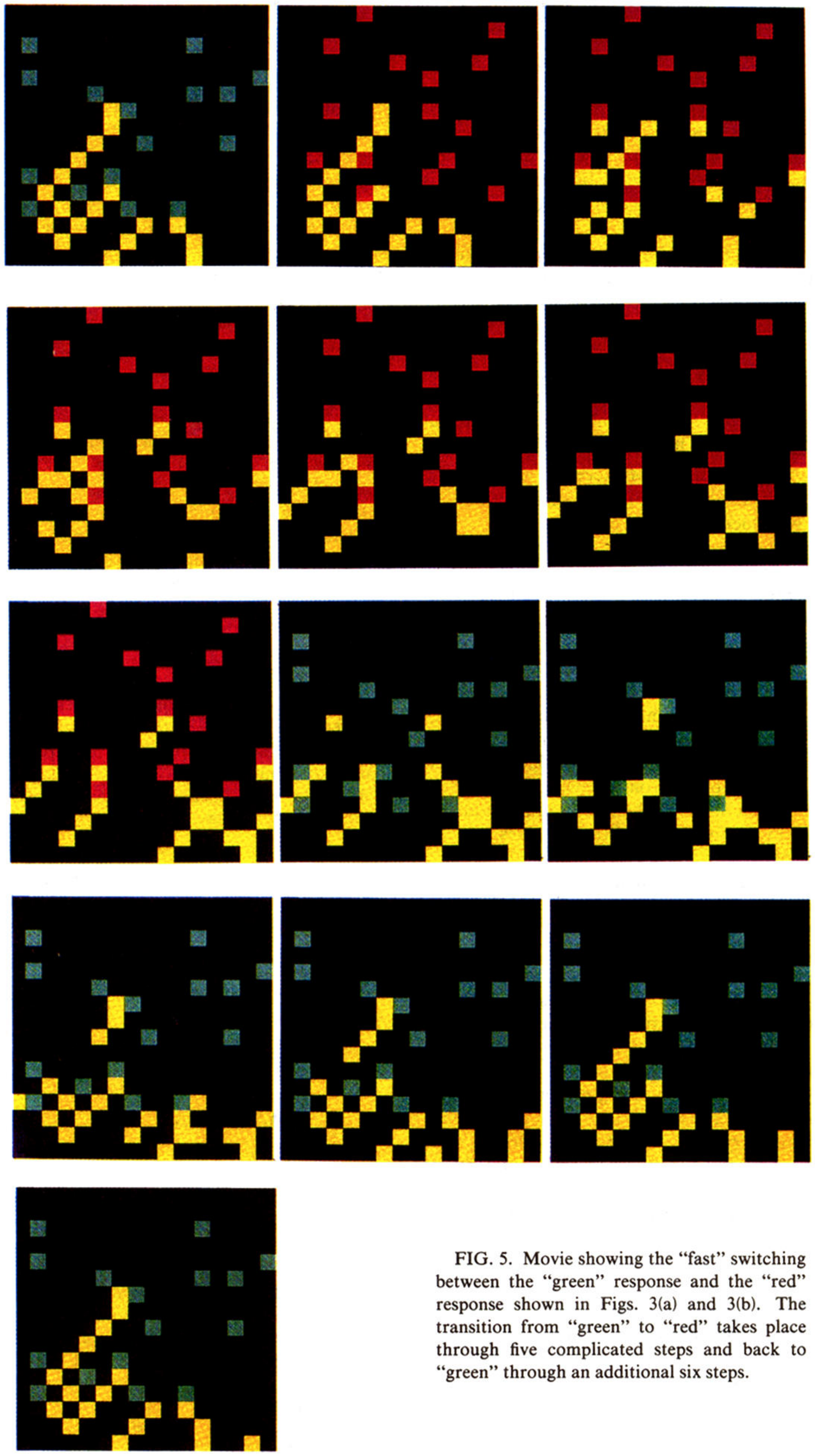


FIG. 5. Movie showing the "fast" switching between the "green" response and the "red" response shown in Figs. 3(a) and 3(b). The transition from "green" to "red" takes place through five complicated steps and back to "green" through an additional six steps.