Field Computation and Nonpropositional Knowledge

Bruce J. MacLennan[*]
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943

Abstract

Most current AI technology has been based on propositionally represented theoretical knowledge. We argue that if AI is to accomplish its goals, especially in the tasks of sensory interpretation and sensorimotor coordination, then it must solve the problem of representing *embodied practical knowledge*. Biological evidence shows that animals use this knowledge in a way very different from digital computation. This suggests that if these problems are to be solved, then we will need a new breed of computers, which we call *field computers*. Examples of field computers are: neurocomputers, optical computers, molecular computers, and any kind of massively parallel analog computer. We claim that the principle characteristic of all these computers is their *massive parallelism,* but we use this term in a special way. We argue that true massive parallelism comes when the number of processors is so large that it can be considered a continuous quantity. Designing and programming these computers requires a new theory of computation, one version of which is presented in this paper. We describe a *universal field computer,* that is, a field computer that can emulate any other field computer. It is based on a generalization of Taylor's theorem to continuous dimensional vector spaces. A number of field computations are illustrated, including several transformations useful in image understanding, and a continuous version of Kosko's bidirectional associative memory.

Keywords: neurocomputers, neural networks, optical computers, molecular computers, field computers, universal field computer, associative memory, parallel processing, massive parallelism, parallel distributed processing, knowledge representation, nonpropositional knowledge, the new AI.

[*] Author's current address: Department of Computer Science, Ayres Hall, University of Tennessee, Knoxville, TN 37996-1301.

1. AN EXTENDED THEORY OF KNOWLEDGE

"Though it be allowed, that reason may form very plausible conjectures with regard to consequences of such a particular conduct in such particular circumstances; it is still supposed imperfect, without assistance of experience, which is alone able to give stability and certainty to the maxims, derived from study and reflection."

— David Hume

1.1 The "New" AI

We argue that AI is moving into a new phase characterized by a broadened understanding of the nature of knowledge, and by the use of new computational paradigms. A sign of this transition is the growing interest in neurocomputers, optical computers, molecular computers and a new generation of massively parallel analog computers. In this section we outline the forces driving the development of this "new" AI. In the remainder of the paper we present the theory of *field computers,* which is intended to be a comprehensive framework for this new paradigm.

The "old" AI has been quite successful in performing a number of difficult tasks, such as theorem proving, chess playing, medical diagnosis and oil exploration. These are tasks that have traditionally required human intelligence and considerable specialized knowledge. On the other hand, there is another class of tasks in which the old AI has made slower progress, such as speech understanding, image understanding, and sensorimotor coordination. It is interesting that these tasks apparently require less intelligence and knowledge than do the tasks that have been successfully attacked. Indeed, most of these recalcitrant tasks are performed skillfully by animals endowed with much simpler nervous systems than our own. How is this possible?

It is apparent that animals perform (at least some) cognitive tasks very differently from computers. Neurons are slow devices. The well-known "Hundred Step Rule"[1] says that there cannot be more than about a hundred sequential processing steps between sensory input and motor output. This suggests that nervous systems perform sensorimotor tasks by relatively shallow, but very wide (i.e., massively parallel) processing. Traditional AI technology depends on the digital computer's ability to do very deep (millions of sequential operations), but narrow (1 to 100 processors) processing. Neurocomputing is an attempt to obtain some of the advantages of the way animals do things by direct emulation of their nervous systems.

1.2 Theoretical and Practical Knowledge

One visible difference between the old and new AIs is in their computational strategies: the former stresses deep but narrow processing, that latter shallow but wide processing. Underlying this difference there is a deeper one: a difference in theories of knowledge. The old AI emphasizes *propositional (verbalizable) knowledge*. That is, it assumes that all knowledge can be represented by sentence-like constructs (i.e., finite ensembles of discrete symbols arranged in accord with definite syntactic rules). The propositional view is not new; it goes very far back, arguably to Pythagoras. Yet there is considerable evidence that nonpropositional knowledge is at least as important.[2]

The problems of practical action, as opposed to theoretical contemplation, are too complicated for propositional analysis. The real world is simply too messy for idealized theories to work. Representation in terms of discrete categories, and cognition by manipulation of discrete structures referring to these categories, may be appropriate to the idealized worlds of chess playing and theorem proving (although even this is doubtful[2]). However, in practical action the context looms large, as does the indefiniteness of categories and the other second order effects that propositional representation routinely idealizes away.

Of course, the approximations of propositional representation can be improved by a deeper theoretical analysis, but this greatly increases the computational burden. Traditional AI is faced with a dilemma: simple theories do not enable skillful behavior, but detailed theories are computationally infeasible. There might seem to be no way to avoid this tradeoff. But, recalling the Hundred Step Rule, and observing that animals behave skillfully, we realize that there must be a third alternative.

The limitations of traditional AI technology show us the limitations of *theoretical* knowledge, i.e., knowledge *that*. There is, however, another kind of knowledge, which we can call *practical* knowledge, or knowledge *how*. For example, a fish knows *how* to maintain its depth in the water, but it does not know *that* neutral buoyancy is achieved by adjusting its specific gravity to that of water. The fish does not have an explicit (propositional) *theory* of how temperature, dissolved substances, etc. affect the specific gravity of water, nor does it know equations describing the complex manner in which its specific gravity depends on the state of its body (food in gullet, air in air bladder, etc. etc.). Rather, the fish's knowledge how is represented nonpropositionally in its nervous system and body.

1.3 The Acquisition of Knowledge

The foregoing suggests that by discovering how to represent and manipulate practical knowledge the new AI may accomplish what the old could not. There are difficulties, however. How is practical knowledge acquired? There are several ways theoretical knowledge is acquired: for example, it may be taught. Since propositions can be encoded in verbal structures, language can be used to transfer theoretical knowledge from one person to another. (Of course, more detailed theories require correspondingly larger verbal structures for their encoding.) Thus, in principle, the representation of theoretical knowledge in a computer is straight forward; we merely have to design an appropriate knowledge representation language. In effect theoretical knowledge is transferred from human to computer in the same way it is transferred from human to human.

Before theoretical knowledge can be transferred it must be acquired in the first place. The original discovery of theoretical knowledge is beyond the scope of this paper. Here we restrict ourselves to the transfer of theoretical knowledge from one person to another; this is the case that is most important for expert systems and other applications of traditional AI technology.

Since practical knowledge is nonpropositional, it cannot be encoded verbally. This does not mean it cannot be taught, however, since we can teach by *showing* as well as by *saying*. Therefore, although theoretical knowledge is transferred by *telling,* practical knowledge is transferred is by *training*. Indeed we often speak of ‘‘training’’ a neural network to accomplish some task.

We have seen how practical knowledge may be transferred. How is it acquired in the first place? In a word, by *adaptation*. In nature adaptation occurs predominantly at two levels: at the species level it leads to innate practical knowledge; at the individual level it leads to learned practical knowledge. The foregoing suggests that where the old AI depended on verbal encoding and transfer, the new AI will emphasize training and adaptation as means of knowledge acquisition.

2. FIELD TRANSFORMATION COMPUTERS

2.1 Massive Parallelism

The preceding section suggests that the new AI will augment the traditional deep, narrow computation with shallow, wide computation. That is, the new AI will exploit *massive parallelism*. Now, massive parallelism means different things to different people; massive parallelism may begin with a hundred, a thousand, or a million processors. On the other hand, biological evidence suggests that skillful behavior requires a very large number of processors, so many in fact that it is infeasible to treat them individually; they must be treated *en masse*. This has motivated us to propose[3] the following definition of massive parallelism:

Definition (Massive Parallelism): A computational system is *massively parallel* if the number of processing elements is so large that it may conveniently be considered a continuous quantity.

That is, a system is *massively* parallel if the processing elements can be considered a *continuous mass* rather than a *discrete ensemble*.

How large a number is large enough to be considered a continuous quantity? That depends on the purpose at hand. A hundred is probably never large enough; a million is probably always large enough; a thousand or ten thousand may be enough. One of the determining factors will be whether the number is large enough to permit the application of continuous mathematics, which is generally more tractable than discrete mathematics.

We propose this definition of massive parallelism for a number of reasons. First, as noted above, skillful behavior seems to require significant neural *mass*. Second, we are interested in computers, such as optical computers and molecular computers, for which the number of processing elements *is* effectively continuous. Third, continuous mathematics is generally easier than discrete mathematics. And fourth, we want to encourage a new style of thinking about parallelism. Currently, we try to apply to parallel machines the thought habits we have acquired from thinking about sequential machines. This strategy works fairly well when the degree of parallelism is low, but it will not scale up. One cannot think individually about the 10^{20} processors of a molecular computer. Rather than postpone the inevitable, we think that we should begin now to develop a theoretical framework for understanding massively parallel computers. The principal goal of this paper is to propose such a theory.

2.2 Field Transformation

Our aim then is to develop a way of looking at massive parallelism that encompasses a variety of implementation technologies, including neural networks, optical computers, molecular computers and a new generation of analog computers. What these all have in common is the ability to process in parallel amounts of data so massive as to be considered a continuous quantity. This suggests that we structure our theory around the idea of a *field,* i.e. a continuous (dense) ensemble of data. We have in mind both scalar fields (such as potential fields) and vector fields (such as gravitational fields). Any operation on such a field, either to produce another field or to produce a new state of the field, can be considered massively parallel, since it operates on all the elements of the field in parallel. Indeed, it would not be feasible to serialize the processing of the field; modest degrees of parallelism cannot cope with the large number of field elements.

In the remainder of this paper we explore *field transformation computers,* that is, computers characterized by the ability to perform (in parallel) transformations on scalar and vector fields. We are not suggesting that field computers are unable to perform scalar calculations; in fact we assume that field transformation computers have the scalar capabilities of conventional digital and analog computers. Scalars have many uses in field computation. For example, we may want to use a scalar parameter to control the rate at which a field transformation takes place (e.g., a reaction rate in a molecular computer). Similarly, we may use a scalar representing the average intensity of a field to control the contrast enhancement of that field. A scalar threshold value may be used to suppress low level noise, and so forth.

An important reason for combining field computation with conventional digital computation is that it permits knowing *how* to be combined with knowing *that*, leading to knowledgeable, skillful behavior. The combined use of propositional and theoretical knowledge is unfortunately beyond the scope of this paper.[2]

2.3 Classes of Field Transformations

Field transformations, like filters, can be divided into two classes: *nonrecursive* and *recursive*. A nonrecursive transformation is simply a functional composition of more elementary transformations. The output of a nonrecursive transformation depends only on its input. A recursive transformation involves some kind of feedback. Hence, its output depends both on its input and on its prior state. Recursive transformations are ideal for simulating the temporal behavior of a physical system, for example in simulated annealing[4] and Boltzmann machines.[5]

2.4 General Purpose Field Computers

Many field computers are designed for special purposes; this has been the case with field computers to date, and we expect it to be the case in the future. In these computers, devices implementing field transformations (such as filters and convolutions) are assembled to solve a small class of problems (e.g., pattern recognition). On the other hand, our experience with digital computation has shown us the value of *general purpose* or *programmable* computers. This architectural feature permits one computer to perform a variety of digital computations, which eliminates the need to construct special purpose devices, and speeds implementation of digital algorithms.

The foregoing observations suggest that general purpose *field* computers will be similarly valuable. In these the connections between field transformation units and field storage units are programmable, thus facilitating their reconnection for a variety of purposes. In fact, we may want to make better use of our resources by *multiplexing* the use of field transformation units under the control of a program. Thus, a program for a general purpose field computer might look very much like a conventional program, except that the basic operations are field transformations rather than scalar arithmetic.

We cannot build into a general purpose field computer every transformation we might need. Instead we must choose a set of primitive operations that permit the programming of all others. How can such a set of primitive operations be chosen? How can we be guaranteed that we have provided all the necessary facilities? For digital computers this question is answered in part by computability theory. For example, this theory shows us how to construct a *universal Turing machine,* which, given an appropriate program, can emulate any Turing machine. Although the universal Turing machine is hardly a practical general purpose computer, consideration of it and other universal machines shows us the kinds of facilities a computer must have in order to be universal. There follows the hard engineering job of going from the theoretically sufficient architecture to the practically necessary architecture.

Can the same be accomplished for field computers? Is there a *universal field computer* that can emulate any field computer? If there is such a thing, then we can expect that it may form a basis for practical general purpose field computers in much the same way that Turing machines do for digital computers. In the next section we prove that general purpose field computation is possible.

3. A UNIVERSAL FIELD COMPUTER

3.1 Introduction

In this section we develop the general theory of field computation and prove the existence of a universal field computer. In particular, we show that with a certain set of built in field transformations we can implement (to a desired degree of accuracy) any field transformation in a very wide class. This is analogous to the result from Turing machine theory: The universal Turing machine allows us to implement (to a desired degree of accuracy) any function in a wide class (now known as the *computable functions*).

The phrase 'to a desired degree of accuracy' appears in both of the preceding statements. What does it mean? For the Turing machine it means that a given accuracy (e.g., precision or range of argument) can be achieved by providing a long enough tape. For the digital computer it means that computations are normally performed to a given precision (e.g., the word length), and that finite increments in the desired precision require finite increments in the resources required (e.g., additional registers and memory cells for double and multiple precision results, or stack space for recursion). The case is much the same for the universal field computer. Finite increments in the desired accuracy of a field transformation will require finite increments in the resources used (such as field transformation and storage units).

There are a number of theoretical bases for a universal field computer. We have investigated designs based on Fourier analysis, interpolation theory and Taylor's theorem, all generalized for field transformations. In this paper we present the design based on Taylor's theorem. There are no doubt as many principles upon which universal field computers can be based as there are bases for universal digital computers.

3.2 Taylor Series Approximation of Field Transforms

In this section we develop the basic theory of functions on scalar and vector fields and of their approximation by Taylor series. Most of the definitions and theorems in sections 3.2 and 3.3 have been previously published[3]; they are reproduced here for completeness. Once it is understood that fields are treated as continuous-dimensional vectors, it will seen that the mathematics is essentially that of finite-dimensional vectors. Note that the treatment here is heuristic rather than rigorous. First we consider scalar fields; later we turn to vector fields.

As usual we take a scalar field to be a function ϕ from an underlying set Ω to an algebraic field K, thus $\phi\colon \Omega \to K$. For our purposes K will be the field of real numbers, $\mathbb{R}$. We use the notation $\Phi(\Omega)$ for the set of all scalar fields over the underlying set Ω ($K = \mathbb{R}$ being understood). Thus, $\Phi(\Omega)$ is a function space, and in fact a linear space under the following definitions of field sum and scalar product:

$$\begin{aligned}(\phi + \psi)_t &= \phi_t + \psi_t \\ (\lambda\phi)_t &= \lambda(\phi_t)\end{aligned} \tag{1}$$

Note that we often write ϕ_t for $\phi(t)$, the value of the field at the point t. As a basis for this linear space we take the unit functions ω_t for each $t \in \Omega$. They are defined

$$\begin{aligned}\omega_t(t) &= 1 \\ \omega_t(s) &= 0, \text{ if } s \neq t\end{aligned} \tag{2}$$

Note that

$$\phi = \int_\Omega \omega_t \phi_t \,\mathrm{d}t \tag{3}$$

The preceding definitions show that we can think of scalar fields as vectors over the set Ω. Since we want to be quite general, we assume only that Ω is a measurable space. In practice, it will usually be a closed and bounded subspace of E^n, n-dimensional Euclidean space. Thus we typically have one, two and three dimensional closed and bounded scalar fields.

Since Ω is a measure space, we can define an inner product between scalar fields:

$$\phi \cdot \psi \equiv \int_\Omega \phi_t \psi_t \,\mathrm{d}t. \tag{4}$$

We also define the norm:

$$||\phi|| = \int_\Omega |\phi_t| \, \mathrm{d}t. \tag{5}$$

Thus $\Phi(\Omega)$ is the function space $L_1(\Omega)$. Note that the ω_t are not an orthogonal set under this norm, since $||\omega_t|| = 0$.

We first consider scalar valued functions of scalar fields, that is functions $f\colon \Phi(\Omega) \to \mathbb{R}$. We prove some basic properties of these functions, culminating in Taylor's theorem.

Definition (Differentiability): Suppose f is a scalar valued function of scalar fields, $f\colon \Phi(\Omega) \to \mathbb{R}$, and that f is defined on a neighborhood of $\phi \in \Phi(\Omega)$. Then we say that f is *differentiable at* ϕ if there is a field $G \in \Phi(\Omega)$ such that for all α in this neighborhood

$$f(\phi + \alpha) - f(\phi) = \alpha \cdot G + \eta ||\alpha|| \tag{6}$$

where $\eta \to 0$ as $||\alpha|| \to 0$. We will later call G the *gradient* of f at ϕ.

Theorem: If f is differentiable at ϕ then f is continuous at ϕ.

Proof: Since f is differentiable at ϕ we know

$$f(\psi) - f(\phi) = (\psi - \phi) \cdot G + \eta ||\psi - \phi||.$$

Therefore,

$$\begin{aligned} |f(\psi) - f(\phi)| &= |\, (\psi - \phi) \cdot G + \eta ||\psi - \phi|| \,| \\ &\le |(\psi - \phi) \cdot G| + |\eta| \, ||\psi - \phi|| \\ &\le ||G|| \, ||\psi - \phi|| + |\eta| \, ||\psi - \phi|| \\ &= (||G|| + |\eta|) \, ||\psi - \phi|| \quad . \end{aligned}$$

Thus f is continuous at ϕ. □

The quantity $\alpha \cdot G$ whose existence is guaranteed by differentiability is called the directional derivative of f with respect to α at ϕ. It is defined directly as follows.

Definition (Directional Derivative): The directional derivative in the "direction" α is given by the following limit:

$$\nabla_\alpha f(\phi) = \frac{\partial f}{\partial \alpha}(\phi) = \lim_{h \to 0} \frac{f(\phi + h\alpha) - f(\phi)}{h} \tag{7}$$

We use $\nabla_\alpha f$ and $\partial f / \partial \alpha$ interchangeably for the directional derivative. Note that if f is differentiable at ϕ then $\nabla_\alpha f(\phi) = \alpha \cdot G$.

Lemma: If f is differentiable in a neighborhood of ϕ, then

$$\frac{\mathrm{d}}{\mathrm{d}x} f(\phi + x\alpha) = \nabla_\alpha f(\phi + x\alpha). \tag{8}$$

Proof: By the definition of the derivative:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x} f(\phi + x\alpha) &= \lim_{h\to 0} \frac{f[\phi + (x+h)\alpha] - f(\phi + x\alpha)}{h} \\ &= \lim_{h\to 0} \frac{f(\phi + x\alpha + h\alpha) - f(\phi + x\alpha)}{h} \\ &= \nabla_\alpha f(\phi + x\alpha)\end{aligned}$$

The last step follows by the definition of the directional derivative. □

Theorem (Mean Value): Suppose $f\colon \Phi(\Omega) \to \mathbb{R}$ is continuous on a neighborhood containing ϕ and ψ. Then, there is a θ, $0 \le \theta \le 1$, such that

$$\begin{aligned} f(\psi) - f(\phi) &= (\psi - \phi) \cdot \nabla f(\chi) \\ \text{where}\quad \chi &= \phi + \theta(\psi - \phi) \end{aligned}. \tag{9}$$

Proof: Let $\alpha = \psi - \phi$ and consider the function

$$F(x) = f(\phi + x\alpha) - f(\phi) - x[f(\psi) - f(\phi)].$$

Since f is continuous, so is F. Now, since $F(0) = F(1) = 0$, we have by Rolle's Theorem that there is a θ, $0 \le \theta \le 1$, such that $F'(\theta) = 0$. Note that

$$\begin{aligned} F'(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\{f(\phi + x\alpha) - f(\phi) - x[f(\psi) - f(\phi)]\} \\ &= \frac{\mathrm{d}}{\mathrm{d}x} f(\phi + x\alpha) - [f(\psi) - f(\phi)] \end{aligned}.$$

By the preceding lemma

$$F'(x) = \nabla_\alpha f(\phi + x\alpha) - [f(\psi) - f(\phi)]$$

Hence, substituting θ for x,

$$0 = F'(\theta) = \nabla_\alpha f(\phi + \theta\alpha) - [f(\psi) - f(\phi)].$$

Therefore, transposing we have

$$f(\psi) - f(\phi) = \nabla_\alpha f(\phi + \theta\alpha)$$

and the theorem is proved. □

Theorem (Taylor): Suppose that f and all its directional derivatives through order $n+1$ are continuous in a neighborhood of ϕ. Then for all α such that $\phi + \alpha$ is in that neighborhood there is a θ, $0 \le \theta \le 1$, such that

$$f(\phi + \alpha) = f(\phi) + \nabla_\alpha f(\phi) + \frac{1}{2}\nabla_\alpha^2 f(\phi) + \cdots + \frac{1}{n!}\nabla_\alpha^n f(\phi) + \frac{1}{(n+1)!}\nabla_\alpha^{n+1} f(\phi + \theta\alpha). \tag{10}$$

Proof: By the Taylor theorem on real variables,

$$f(\phi + t\alpha) = f(\phi) + \frac{\mathrm{d}}{\mathrm{d}t} f(\phi)t + \frac{1}{2}\frac{\mathrm{d}^2}{\mathrm{d}t^2} f(\phi)t^2 + \cdots + \frac{1}{n!}\frac{\mathrm{d}^n}{\mathrm{d}t^n} f(\phi)t^n + \frac{1}{(n+1)!}\frac{\mathrm{d}^{n+1}}{\mathrm{d}t^{n+1}} f(\phi + \theta\alpha)t^{n+1} .$$

Observe that by the preceding lemma

$$\frac{\mathrm{d}^n}{\mathrm{d}t^n} f(\phi + t\alpha) = \nabla_\alpha^n f(\phi + t\alpha).$$

Therefore,

$$f(\phi + t\alpha) = f(\phi) + \nabla_\alpha f(\phi)t + \frac{1}{2}\nabla_\alpha^2 f(\phi)t^2 + \cdots + \frac{1}{n!}\nabla_\alpha^n f(\phi)t^n + \frac{1}{(n+1)!}\nabla_\alpha^{n+1} f(\phi + \theta\alpha)t^{n+1} .$$

Setting $t = 1$ gives the desired result. □

The extension to a function of *several* scalar fields is routine.

Since our ‘‘vectors’’ are continuous dimensional, partial derivatives are with respect to a ‘‘coordinate’’ $t \in \Omega$ rather than with respect to a coordinate variable. To define partial derivatives it's convenient to make use of the Dirac delta functions, δ_t, for $t \in \Omega$:

$$\begin{aligned} \delta_t(t) &= \infty \\ \delta_t(s) &= 0, \text{ for } s \ne t \end{aligned} \tag{11}$$

Of course, by the first equation above we mean $\delta_t(s) = \lim_{\varepsilon \to 0} \varepsilon^{-1}$ for $|s - t| < \varepsilon/2$. Note the following properties of the delta functions (fields):

$$\begin{aligned} &\|\delta_t\| = 1 \\ &\delta_t \cdot \phi = \phi_t \\ &\int_\Omega \omega_t\, \delta_t \cdot \phi \,\mathrm{d}t = \phi \end{aligned} \tag{12}$$

Given the delta functions the partial derivative of f at coordinate t is simply expressed:

$$\frac{\partial f}{\partial \delta_t} = \nabla_{\delta_t} f \tag{13}$$

Theorem: If f is differentiable at ϕ then the first order partial derivatives exist at ϕ.

Proof: First observe that by differentiability

$$\begin{aligned}\frac{f(\phi + h\delta_t) - f(\phi)}{h} &= \frac{f(\phi) + h\delta_t \cdot G + \eta \| h\delta_t \| - f(\phi)}{h} \\ &= \delta_t \cdot G + \eta \| \delta_t \| \, |h|/h \\ &= \delta_t \cdot G + \eta |h|/h \\ &= G_t + \eta |h|/h\end{aligned}$$

Recalling that $\eta \to 0$ as $h \to 0$, observe

$$\begin{aligned}\lim_{h\to 0} \left| \frac{f(\phi + h\delta_t) - f(\phi)}{h} - G_t \right| &= \lim_{h\to 0} |G_t + \eta |h|/h - G_t| \\ &= \lim_{h\to 0} |\eta| \\ &= 0\end{aligned}$$

Hence, $\frac{\partial}{\partial \delta_t} f(\phi) = G_t$, where G is the field whose existence is guaranteed by differentiability. Thus the partial derivative exists. □

What is the field G whose points are the partial derivatives? It is just the gradient of the function.

Definition (Gradient): The gradient of $f\colon \Phi(\Omega) \to \mathbb{R}$ at ϕ is a field whose value at a point t is the partial derivative at that point, $\frac{\partial}{\partial \delta_t} f(\phi)$:

$$[\nabla f(\phi)]_t = \frac{\partial}{\partial \delta_t} f(\phi). \tag{14}$$

Since, by Eq. 3, $\nabla f(\phi) = \int_\Omega \omega_t [\nabla f(\phi)]_t \, \mathrm{d}t$, the gradient can also be expressed in terms of the basis functions and the partial derivatives:

$$\nabla f(\phi) = \int_\Omega \omega_t \frac{\partial}{\partial \delta_t} f(\phi) \, \mathrm{d}t. \tag{15}$$

When no confusion will result, we use the following operator notations:

$$\begin{aligned} \nabla f &= \int_\Omega \omega_t \; \partial f / \partial \delta_t \, \mathrm{d}t \\ \nabla &= \int_\Omega \omega_t \; \partial / \partial \delta_t \, \mathrm{d}t \\ \partial / \partial \delta_t &= \delta_t \cdot \nabla = \nabla_{\delta_t} \end{aligned} \tag{16}$$

Finally, since $\nabla f(\phi) = G$, the field guaranteed by differentiability, and $\nabla_\alpha f(\phi) = \alpha \cdot G$, we know

$$\frac{\partial f}{\partial \alpha}(\phi) = \nabla_\alpha f(\phi) = \alpha \cdot \nabla f(\phi) \tag{17}$$

or, in operator form, $\partial / \partial \alpha = \nabla_\alpha = \alpha \cdot \nabla$.

3.3 A Universal Field Computer Based on Taylor Series Approximation

We can use Taylor's Theorem to derive approximations of quite a general class of scalar valued functions of scalar fields. Thus, if we equip our universal field computer with the hardware necessary to compute Taylor series approximations, then we will be able to compute any of a wide class of functions (namely, those functions whose first n partial derivatives exist and are continuous). Therefore, consider the general form of an n-term Taylor series:

$$f(\phi) \approx \sum_{k=0}^{n} \frac{1}{k!} \nabla_\alpha^k f(\phi_0), \quad \text{where } \alpha = \phi - \phi_0 \tag{18}$$

What hardware is required? Clearly we will need a field subtractor for computing the difference field $\alpha = \phi - \phi_0$. We will also need a scalar multiplier for scaling each term by $1/k!$; we will also need a scalar adder for adding the terms together. The harder problem is to find a way to compute $\nabla_\alpha^k f(\phi_0)$ for a vector α that depends on the (unknown) input ϕ. The trouble is that the αs and the ∇s are interleaved, as can be seen here:

$$\begin{aligned} \nabla_\alpha^k f(\phi_0) &= (\alpha \cdot \nabla)^k f(\phi_0) \\ &= (\alpha \cdot \nabla)^{k-1} \, [\alpha \cdot \nabla f(\phi_0)] \\ &= (\alpha \cdot \nabla)^{k-1} \int_\Omega \alpha_{t_1} \frac{\partial}{\partial \delta_{t_1}} f(\phi_0) \, \mathrm{d}t_1 \\ &\;\vdots \\ &= \int_\Omega \cdots \int_\Omega \int_\Omega \alpha_{t_1} \alpha_{t_2} \cdots \alpha_{t_k} \frac{\partial^k}{\partial \delta_{t_1} \partial \delta_{t_2} \cdots \partial \delta_{t_k}} f(\phi_0) \, \mathrm{d}t_1 \, \mathrm{d}t_2 \cdots \mathrm{d}t_k \end{aligned}$$

We want to separate everything that depends on α, and is thus variable, from everything that depends on $f(\phi_0)$, and is thus fixed. This can be accomplished (albeit, with extravagant use of our dimensional resources) by means of an outer product operation. Therefore we define the outer product of two scalar fields:

$$(\phi \curlywedge \psi)_{s,t} \;=\; \phi_s \psi_t \tag{19}$$

Note that if $\phi,\ \psi\ \in \Phi(\Omega)$ then $\phi \curlywedge \psi\ \in \Phi(\Omega^2)$.

To see how the outer product allows the variable and fixed parts to be separated, consider first the case ∇_α^2:

$$\begin{aligned}
\nabla_\alpha^2 f(\phi_0) &= \int_\Omega \int_\Omega \alpha_s \alpha_t \, \frac{\partial}{\partial \delta_s} \, \frac{\partial}{\partial \delta_t} \, f(\phi_0) \, \mathrm{d}t \, \mathrm{d}s \\
&= \int_\Omega \int_\Omega (\alpha \curlywedge \alpha)_{s,t} \, (\nabla)_s (\nabla)_t \, f(\phi_0) \, \mathrm{d}t \, \mathrm{d}s \\
&= \int_\Omega \int_\Omega (\alpha \curlywedge \alpha)_{s,t} \, (\nabla \curlywedge \nabla)_{s,t} \, f(\phi_0) \, \mathrm{d}t \, \mathrm{d}s \\
&= \int_{\Omega^2} (\alpha \curlywedge \alpha)_x (\nabla \curlywedge \nabla)_x \mathrm{d}x \, f(\phi_0) \\
&= (\alpha \curlywedge \alpha) \cdot (\nabla \curlywedge \nabla) \, f(\phi_0)
\end{aligned}$$

Now we can see how the general case goes. First we define the k-fold outer product:

$$\begin{aligned}
\phi^{[1]} &= \phi \\
\phi^{[k+1]} &= \phi \curlywedge \phi^{[k]}
\end{aligned} \tag{20}$$

Then,

$$\nabla_\alpha^k f(\phi) \;=\; \alpha^{[k]} \cdot \nabla^{[k]} f(\phi) \tag{21}$$

The n-term Taylor series then becomes

$$f(\phi) \;\approx\; \sum_{k=0}^{n} \frac{1}{k!} \, (\phi - \phi_0)^{[k]} \cdot \nabla^{[k]} f(\phi_0) \tag{22}$$

Since ϕ_0 is fixed, we can compute each $\nabla^{[k]} f(\phi_0)$ once, when the field computer is programmed. Then, for any given input ϕ we can compute $(\phi - \phi_0)^{[k]}$ and take the inner product of this with $\nabla^{[k]} f(\phi_0)$. Thus, in addition to the components mentioned above, computing the Taylor series approximation also requires outer and inner product units that will accommodate spaces up to those in $\Phi(\Omega^n)$.

We consider a very simple example of Taylor series approximation. Suppose we want to approximate $\mathrm{defint}\ \phi$, which computes the definite integral of ϕ, $\mathrm{defint}\ \phi = \int_\Omega \phi_s \ \mathrm{d}s$. First we determine its partial derivative at t by observing:

$$\begin{aligned}\lim_{h\to 0}\frac{\operatorname{defint}(\phi + h\delta_t) - \operatorname{defint}\phi}{h} &= \lim_{h\to 0}\frac{\int_\Omega \phi_s + h\delta_t(s)\,\mathrm{d}s - \int_\Omega \phi_s\,\mathrm{d}s}{h}\\ &= \lim_{h\to 0}\frac{\int_\Omega \phi_s\,\mathrm{d}s + h\int_\Omega \delta_t(s)\,\mathrm{d}s - \int_\Omega \phi_s\,\mathrm{d}s}{h}\\ &= \lim_{h\to 0} h\|\delta_t\|/h = 1\end{aligned}$$

Thus, $\frac{\partial}{\partial \delta_t}$ defint $\phi = 1$, and we can see that

$$\nabla \operatorname{defint}\phi = \mathbf{1}, \tag{23}$$

where $\mathbf{1}$ is the constant 1 function, $\mathbf{1}_t = 1$. This leads to a one term Taylor series, which is exact:

$$\operatorname{defint}\phi = \phi \cdot \mathbf{1} \tag{24}$$

Note that $\mathbf{1}$ is a fixed field that must be loaded into the computer.

3.4 Transformations on Scalar and Vector Fields

The previous results apply to *scalar* valued functions of scalar fields. These kinds of functions are useful (e.g., to compute the average value of a scalar field), but they do not exploit the full parallelism of a field computer. Achieving this requires the use of functions that accept a (scalar or vector) field as input, and return a *field* as output. We briefly sketch the theory for scalar field valued functions of scalar fields; transformations on vector fields are an easy extension of this.

By a scalar field valued transformation of scalar fields we mean a function $\mathbf{F}\colon \Phi(\Omega_1) \to \Phi(\Omega_2)$. Such a transformation is considered a family of scalar valued functions $f_t\colon \Phi(\Omega_1) \to \mathbb{R}$ for each $t \in \Omega_2$; these are the *component functions* of $\mathbf{F}$. Note that $\mathbf{F}$ can be expressed in terms of its components:

$$\mathbf{F}(\phi) = \int_{\Omega_2} f_t(\phi)\,\omega_t\,\mathrm{d}t \tag{25}$$

More briefly, $\mathbf{F} = \int_{\Omega_2} f_t\omega_t\,\mathrm{d}t$. $\mathbf{F}$ is decomposed into its components by $\delta_t \cdot \mathbf{F}(\phi) = f_t(\phi)$.

Next we turn to the differentiability of field transformations. To define this it is necessary to first define an analog to the inner product for fields of different dimension. Thus, we define the continuous-dimensional analogue of a vector-matrix product:

$$(\Psi\phi)_s = \int_\Omega \Psi_{st}\phi_t \mathrm{d}t = \Psi_s \cdot \phi \qquad (26)$$
$$(\phi\Psi)_t = \int_\Omega \phi_s \Psi_{st} \mathrm{d}s = \phi \cdot \Psi_t^{\mathrm{T}}$$

where the transpose of a field is defined $\Psi_{ts}^{\mathrm{T}} = \Psi_{st}$. With this notation differentiability can be defined.

Definition (Differentiability of Field Transformations): Suppose $\mathbf{F}\colon \Phi(\Omega_1) \rightarrow \Phi(\Omega_2)$ is a field valued function of scalar fields defined on a neighborhood of $\phi \in \Phi(\Omega_1)$. We say that $\mathbf{F}$ is *differentiable at* ϕ provided there is a field $\Gamma \in \Phi(\Omega_1 \times \Omega_2)$ such that for all α in the neighborhood

$$\mathbf{F}(\phi + \alpha) - \mathbf{F}(\phi) = \alpha\Gamma + \mathrm{H}\|\alpha\| \qquad (27)$$

where $\mathrm{H} \in \Phi(\Omega_2)$ and $\|\mathrm{H}\| \rightarrow 0$ as $\|\alpha\| \rightarrow 0$. We will show that Γ is the gradient of $\mathbf{F}$ at ϕ.

Next we consider the directional derivative of a field transformation. For a scalar function f, $\nabla_\alpha f(\phi)$ is a scalar that describes how much $f(\phi)$ changes when its argument is perturbed by a small amount in the "direction" α. For a field transformation $\mathbf{F}$, $\boldsymbol{\nabla}_\alpha \mathbf{F}(\phi)$ should be a *field,* each component of which reflects how much the corresponding component of $\mathbf{F}(\phi)$ changes when ϕ moves in the "direction" α. That is, $[\boldsymbol{\nabla}_\alpha \mathbf{F}(\phi)]_t = \nabla_\alpha\ \delta_t \cdot \mathbf{F}(\phi)$. Hence,

$$\boldsymbol{\nabla}_\alpha \mathbf{F}(\phi) = \int_{\Omega_2} \omega_t\ \nabla_\alpha\ \delta_t \cdot \mathbf{F}(\phi)\ \mathrm{d}t, \qquad (28)$$

or, more briefly, $\boldsymbol{\nabla}_\alpha \mathbf{F} = \int_{\Omega_2} \omega_t\ \nabla_\alpha\ \delta_t \cdot \mathbf{F}\ \mathrm{d}t$. The directional derivative is defined directly by a limit.

Definition (Directional Derivative of Field Transformations): If $\mathbf{F}\colon \Phi(\Omega_1) \rightarrow \Phi(\Omega_2)$ is differentiable at ϕ then

$$\boldsymbol{\nabla}_\alpha \mathbf{F}(\phi) = \frac{\partial \mathbf{F}}{\partial \alpha}(\phi) = \lim_{h \rightarrow 0} \frac{\mathbf{F}(\phi + h\alpha) - \mathbf{F}(\phi)}{h} \qquad (29)$$

It is obvious that $\boldsymbol{\nabla}_\alpha \mathbf{F}(\phi) = \alpha\Gamma$, where Γ is the field whose existence is guaranteed by differentiability. This suggests the definition:

Definition (Gradient of Field Transformation): The gradient of $\mathbf{F}$ at ϕ is the field whose value at a point t is the partial derivative of $\mathbf{F}$ at that point:

$$[\boldsymbol{\nabla}\mathbf{F}(\phi)]_t = \frac{\partial \mathbf{F}}{\partial \delta_t}(\phi) \qquad (30)$$

Since $\boldsymbol{\nabla}_{\delta_t} \mathbf{F}(\phi) = \delta_t \Gamma = \Gamma_t$, it follows that $\boldsymbol{\nabla}\mathbf{F}(\phi) = \Gamma$, as expected.

Notice that the gradient is of higher dimension than the field transformation. That is, if $\mathbf{F}\colon \Phi(\Omega_1) \rightarrow \Phi(\Omega_2)$, then $\nabla\mathbf{F}\colon \Phi(\Omega_1) \rightarrow \Phi(\Omega_1 \times \Omega_2)$. Higher order gradients will have similarly higher order:

$$\nabla^k\mathbf{F}\colon\ \Phi(\Omega_1) \ \rightarrow\ \Phi(\Omega_1{}^k \times \Omega_2),\ \text{for}\ \mathbf{F}\colon\ \Phi(\Omega_1) \ \rightarrow\ \Phi(\Omega_2) \tag{31}$$

A derivation similar to that for scalar field functions yields the Taylor series for field transformations:

$$\mathbf{F}(\phi) \ \approx\ \sum_{k=0}^{n} \frac{1}{k!}\ [(\phi - \phi_0)\ \nabla]^k\mathbf{F}(\phi_0) \tag{32}$$

As before, outer products can be used to separate the variable and fixed components.

We illustrate the Taylor series computation for a simple but important class of field transformations, the integral operators. For example, the derivative and difference transformations, which are very useful in image processing, are integral operators, as will be shown below.

Definition (Integral Operator): A field transformation $\mathbf{F}\colon \Phi(\Omega_1) \rightarrow \Phi(\Omega_2)$ is an *integral operator* if there is a field $\Psi \in \Phi(\Omega_2 \times \Omega_1)$, called its *kernel,* such that

$$\mathbf{F}(\phi) \ =\ \Psi\phi \tag{33}$$

Recall $(\Psi\phi)_s = \int_{\Omega_1} \Psi_{st}\phi_t \mathrm{d}t$.

Since $\mathbf{F}(\phi + \psi) = \Psi(\phi + \psi) \ =\ \Psi\phi + \Psi\psi \ = \mathbf{F}(\phi) + \mathbf{F}(\psi)$, it's clear that integral operators are linear. Thus, their gradients are especially easy to compute:

Theorem (Gradient of Integral Operator): The gradient of an integral operator is the transpose of its kernel.

Proof: Suppose $\mathbf{F}(\phi) = \Psi\phi$ is an integral operator. Observe $\mathbf{F}(\phi + \alpha) - \mathbf{F}(\phi) = \Psi\alpha = \alpha\Psi^{\mathrm{T}}$, where $\Psi^{\mathrm{T}}_{ts} = \Psi_{st}$. Hence $\nabla\mathbf{F}(\phi) = \Psi^{\mathrm{T}}$. □

Since the gradient of a constant function is zero, all higher order gradients vanish, so the Taylor series for an integral operator is exact: $\mathbf{F}(\phi) = \phi\nabla\mathbf{F} = \phi\Psi^{\mathrm{T}}$.

Next we show that the derivative and difference operations on scalar fields are integral operators and we compute their Taylor series. First consider the finite difference transformation $\Delta\phi$ defined so that

$$\Delta\phi_t \; = \; \phi_{t+h} - \phi_t \tag{34}$$

for a given $h > 0$. To see that this is an integral operator observe:

$$\Delta\phi_t \; = \; \delta_{t+h} \cdot \phi - \delta_t \cdot \phi \; = \; (\delta_{t+h} - \delta_t) \cdot \phi \tag{35}$$

Define the field Ψ by $\Psi_{tu} = \delta_{t+h}(u) - \delta_t(u)$ and it follows that $\Delta\phi = \Psi\phi$, since

$$(\Psi\phi)_t \; = \; \int_{\Omega_1} \Psi_{tu}\phi_u \mathrm{d}u \; = \; \int_{\Omega_1} [\delta_{t+h}(u) - \delta_t(u)]\phi_u \, \mathrm{d}u \; = \; (\delta_{t+h} - \delta_t) \cdot \phi \tag{36}$$

The only trouble with this formula for the finite difference is that the field Ψ is not physically realizable, since it makes use of the Dirac functions. In practice we have to replace δ_{t+h} and δ_t by finite approximations, but the resulting approximate difference transformation is still an integral operator. The same applies to the derivative transformation, which can be approximated by the approximations to the first and higher order derivatives.

To further illustrate the Taylor series for scalar field valued transformations, we consider *pointwise* transformations. A pointwise transformation applies a scalar function (but not necessarily the same function) to every element of a field. That is, $\mathbf{F}\colon \Phi(\Omega) \to \Phi(\Omega)$ is a pointwise transformation if for some $f_t\colon \mathbb{R} \to \mathbb{R}$,

$$[\mathbf{F}(\phi)]_t = f_t(\phi_t) \tag{37}$$

Note that

$$\delta_t \cdot \mathbf{F}(\phi) = f_t(\phi_t). \tag{38}$$

Lemma: If $\mathbf{F}\colon \Phi(\Omega) \to \Phi(\Omega)$ is a pointwise transformation, then

$$\nabla_\alpha \, \delta_t \cdot \mathbf{F}(\phi) \; = \; \delta_t \cdot \alpha\mathbf{F}'(\phi) \tag{39}$$

where $[\mathbf{F}'(\phi)]_t = f_t{}'(\phi_t)$, and $f_t{}'$ is the derivative of f_t.

Proof: By the definition of the derivative:

$$
\begin{aligned}
\nabla_\alpha\ \delta_t \cdot \mathbf{F}(\phi) &= \lim_{h\to 0} \frac{\delta_t \cdot \mathbf{F}(\phi + h\alpha) - \delta_t \cdot \mathbf{F}(\phi)}{h} \\
&= \lim_{h\to 0} \frac{f_t[(\phi + h\alpha)_t] - f_t(\phi_t)}{h} \\
&= \lim_{h\to 0} \frac{f_t(\phi_t + h\alpha_t) - f_t(\phi_t)}{h} \\
&= \lim_{h\to 0} \alpha_t f_t{}'(\phi_t) + \varepsilon |h\alpha_t|/h \\
&= \alpha_t f_t{}'(\phi_t) \\
&= \delta_t \cdot \alpha \mathbf{F}'(\phi)
\end{aligned}
$$

□

Theorem (Directional Derivative of Pointwise Transformation): If $\mathbf{F}\colon \Phi(\Omega) \to \Phi(\Omega)$ is a pointwise transformation, then

$$\nabla_\alpha \mathbf{F}(\phi) = \alpha \mathbf{F}'(\phi) \tag{40}$$

Proof: Applying the preceding lemma:

$$
\begin{aligned}
\nabla_\alpha \mathbf{F}(\phi) &= \int_\Omega \omega_t\ \nabla_\alpha\ \delta_t \cdot \mathbf{F}(\phi)\, \mathrm{d}t \\
&= \int_\Omega \omega_t\ \delta_t \cdot \alpha \mathbf{F}'(\phi)\, \mathrm{d}t \\
&= \alpha \int_\Omega \omega_t\ \delta_t \cdot \mathbf{F}'(\phi)\, \mathrm{d}t \\
&= \alpha \mathbf{F}'(\phi)
\end{aligned}
$$

The last step follows by Eq. 12. □

Corollary: The directional derivative of a pointwise transformation is a pointwise transformation.

Proof: This follows immediately from the preceding theorem. □

The theorem and its corollary lead to the Taylor series expansion of a pointwise transformation:

Theorem (Taylor Series for Pointwise Transformation): If $\mathbf{F}\colon \Phi(\Omega) \to \Phi(\Omega)$ is a pointwise transformation, and its component functions and all their derivatives through $n + 1$ are continuous in a neighborhood of ϕ, then for all α such that $\phi + \alpha$ is in that neighborhood there is a field θ, $0 \le \theta_t \le 1$, such that

$$\mathbf{F}(\phi + \alpha) = \sum_{k=0}^{n} \frac{1}{k!} \alpha^k \mathbf{F}^{(k)}(\phi) + \frac{1}{(n+1)!} \alpha^{n+1} \mathbf{F}^{(n+1)}(\phi + \theta\alpha) \tag{41}$$

Here $\mathbf{F}^{(k)}$ is the kth derivative of $\mathbf{F}$:

$$[\mathbf{F}^{(k)}(\phi)]_t = f_t^{(k)}(\phi_t) = \left. \frac{\mathrm{d}^k f_t(x)}{\mathrm{d}x^k} \right|_{x=\phi_t} \tag{42}$$

where $f_t = \delta_t \cdot \mathbf{F}$.

Proof: By the Taylor theorem for functions of reals:

$$\begin{aligned}
[\mathbf{F}(\phi + \alpha)]_t &= f_t(\phi_t + \alpha_t) \\
&= \sum_{k=0}^{n} \frac{1}{k!} \alpha_t^k f_t^{(k)}(\phi_t) + \frac{1}{(n+1)!} \alpha_t^{n+1} f_t^{(n+1)}(\phi_t + \theta_t \alpha_t) \\
&= \sum_{k=0}^{n} \frac{1}{k!} [\alpha^k \mathbf{F}^{(k)}(\phi)]_t + \frac{1}{(n+1)!} [\alpha^{n+1} \mathbf{F}^{(n+1)}(\phi + \theta\alpha)]_t
\end{aligned}$$

□

This results permits converting Taylor series for scalar functions to the Taylor series for the corresponding pointwise field transformations. Notice also that the resulting series do not require outer products, provided we have a field multiplier, $(\phi\psi)_t = \phi_t \psi_t$.

We illustrate this theorem by determining a Taylor series approximation for **ln**, the function that computes the natural logarithm of each field element.

Theorem (Taylor Series for Pointwise Logarithm): Suppose $(\mathbf{ln}\ \phi)_t = \ln \phi_t$. Then

$$\begin{aligned}
\mathbf{ln}\ \phi &= (\phi - 1) - \frac{1}{2}(\phi - 1)^2 + \frac{1}{3}(\phi - 1)^3 - \cdots + \frac{(-1)^{n-1}}{n}(\phi - 1)^n \\
&\quad + \frac{1}{(n+1)!} \mathbf{ln}\ [1 + \theta(\phi - 1)]
\end{aligned} \tag{43}$$

provided $|\phi_t - 1| \le 1$ and $\phi_t \ne 0$, for all $t \in \Omega$.

Proof: Note that for $k > 0$, $\mathbf{ln}^{(k)}\ \phi = (-1)^{k-1}(k-1)!\ /\ \phi^k$. Therefore, for $k > 1$, $\mathbf{ln}^{(k)}\ 1 = (-1)^{k-1}(k-1)!$. By the Taylor theorem,

$$\begin{aligned}\mathbf{ln}\,(1+\alpha) &= \mathbf{ln}\,1 + \sum_{k=1}^{n} \frac{1}{k!}\,\alpha^k \mathbf{ln}^{(k)}\,1 + \frac{1}{(n+1)!}\,\alpha^{n+1}\mathbf{ln}^{(k)}(1+\theta\alpha) \\ &= \sum_{k=1}^{n} \frac{1}{k!}\,\alpha^k(-1)^{k-1}(k-1)! + \frac{1}{(n+1)!}\,\alpha^{n+1}\mathbf{ln}^{(k)}(1+\theta\alpha) \\ &= \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k}\,\alpha^k + \frac{1}{(n+1)!}\,\alpha^{n+1}\mathbf{ln}^{(k)}(1+\theta\alpha)\end{aligned}$$

To prove the theorem let $\alpha = \phi - 1$. □

We consider vector fields briefly. Recall that any three-dimensional vector field Φ can be considered three scalar fields ϕ, ψ, χ where

$$\Phi_t = \phi_t \mathbf{i} + \psi_t \mathbf{j} + \chi_t \mathbf{k} \tag{44}$$

Similarly, a function that returns a three-dimensional vector field can be broken down into three functions that return scalar fields. Thus, we see that a transformation on finite dimensional vector fields can be implemented by a finite number of transformations on scalar fields.

To ensure the continuity of field valued functions, certain restrictions must be placed on the fields permitted as arguments. Although these restrictions are still under investigation, we believe that it is sufficient that the input field's gradient be bounded at each stage. This will be the case for all physically realizable fields. This restriction on allowable inputs finds its analogy in digital computers: legal input numbers are restricted to some range; numbers outside that range may cause underflow or overflow in the subsequent computation. In the same way here, fields whose gradients are too large may lead to incorrect results.

4. EXAMPLE APPLICATION: BIDIRECTIONAL ASSOCIATIVE FIELD MEMORY

In this section we illustrate the theory of field computation by analyzing a continuous version of Kosko's *bidirectional associative memory.*[6] The system operates as follows (see Figure 1).

[figure omitted]

Figure 1. Bidirectional Associative Field Memory

The goal is to store a number of associative pairs $(\psi^{(k)}, \phi^{(k)})$, where $\phi^{(k)} \in \Phi(\Omega_1)$ and $\psi^{(k)} \in \Phi(\Omega_2)$. (We assume these fields are *bipolar,* that is, take values in $\{-1, +1\}$, although physical realizability actually requires continuous variation between -1 and $+1$.) Presentation of a field $\phi \in \Phi(\Omega_1)$ at in_1

eventually yields at out_1 and out_2 the pair j for which $\phi^{(j)}$ is the closest match for ϕ. Similarly, presentation of ψ at in_2 will yield the pair for which $\psi^{(j)}$ is the closest match. The pairs are stored in a distributed fashion in the field $\Psi \in \Phi(\Omega_2 \times \Omega_1)$ computed as follows:

$$\Psi = \psi^{(1)} \wedge \phi^{(1)} + \cdots + \psi^{(n)} \wedge \phi^{(n)} \tag{45}$$

Note that $\psi^{(k)} \wedge \phi^{(k)}$ reflects the cross correlations between the values of $\psi^{(k)}$ and the values of $\phi^{(k)}$.

Two of the boxes Fig. 1 perform "matrix-vector" multiplications, $\Psi\phi$ or $\psi\Psi$. Thus presentation of a field ϕ at in_1 yields

$$\psi' = S(\psi, \Psi\phi) \tag{46}$$

at out_1. Here S is a nonlinear function that helps to suppress crosstalk between stored pairs by forcing field values back to -1 or $+1$ (it is described below). Computation of ψ in turn yields

$$\phi' = S(\phi, \psi'\Psi) \tag{47}$$

at out_2. On the next iteration we get $\psi'' = S(\psi', \Psi\phi')$ and $\phi'' = S(\phi', \psi''\Psi)$, and so forth. Each iteration will yield a closer approximation of the desired $(\psi^{(k)}, \phi^{(k)})$. We cannot in general guarantee that the system will stabilize (i.e., that the ϕ and ψ will become constant), but we will show that the changes will become as small as we like. Kosko can show stability, since the discreteness of his fields places a lower bound on nonzero change.

The nonlinear function S updates the output field in the following way [illustrated for the computation $\psi' = S(\psi, \Psi\phi)$]:

$$\psi'_t = \begin{cases} +1 & \text{if } \Psi_t \cdot \phi > \theta_1 \\ \psi_t & \text{if } -\theta_2 \le \Psi_t \cdot \phi \le \theta_1 \\ -1 & \text{if } \Psi_t \cdot \phi < -\theta_2 \end{cases} \tag{48}$$

Thus, the value of a field element ψ_t is not changed if $-\theta_2 \le \Psi_t \cdot \phi \le \theta_1$, where the thresholds $\theta_1, \theta_2 > 0$. The rule for $\phi' = S(\phi, \psi\Psi)$ is analogous.

Following Hopfield[7] we show that the stored pairs $(\psi^{(k)}, \phi^{(k)})$ are stable states in the dynamic behavior of the memory.

Theorem (Stability of Stored Pairs): Any stored pair is stable, provided $|\Omega_1| > \theta$, where

$\theta = \max(\theta_1, \theta_2)$. (This condition holds for realistic values of the thresholds.)

Proof: Suppose that $(\psi^{(j)}, \phi^{(j)})$ is one of the stored pairs and observe:

$$\begin{aligned} \Psi\phi^{(j)} &= (\sum_k \psi^{(k)} \wedge \phi^{(k)})\phi^{(j)} \\ &= \sum_k (\psi^{(k)} \wedge \phi^{(k)})\phi^{(j)} \\ &= \sum_k \psi^{(k)}(\phi^{(k)} \cdot \phi^{(j)}) \end{aligned}$$

The expression $\phi^{(k)} \cdot \phi^{(j)}$ measures the correlation between $\phi^{(k)}$ and $\phi^{(j)}$, which varies between $-|\Omega_1|$ and $+|\Omega_1|$. Notice that for $j \neq k$ this expression has mean value 0. On the other hand, when $j = k$ its value is $|\Omega_1|$. Hence,

$$\Psi\phi^{(j)} \approx \psi^{(j)}|\Omega_1| \tag{49}$$

Now consider the crucial expression from Eq. 48:

$$\Psi_t \cdot \phi^{(j)} \approx \psi_t^{(j)}|\Omega_1| \tag{50}$$

If $\psi_t^{(j)} = +1$, then

$$\Psi_t \cdot \phi^{(j)} \approx |\Omega_1| > \theta_1$$

and so $\psi'_t = +1$. Similarly, if $\psi_t^{(j)} = -1$, then

$$\Psi_t \cdot \phi^{(j)} \approx -|\Omega_1| < -\theta_2$$

and so $\psi'_t = -1$. In both cases $\psi'_t = \psi_t^{(j)}$. Since $\psi' = \psi^{(j)}$ and (by similar reasoning) $\phi' = \phi^{(j)}$, we see that any stored pair $(\psi^{(j)}, \phi^{(j)})$ is stable. □

The preceding theorem shows that the stored pairs are stable, but it does not show that they will ever be reached. Therefore we prove several theorems establishing conditions under which correct recall takes place. The following theorem shows that if ϕ is sufficiently close to one $\phi^{(j)}$ and sufficiently far from all the others, then perfect recall occurs in one step.

Theorem (Close Matches Lead to Correct Recall): $\psi' = \psi^{(j)}$, provided that

$$\phi^{(j)} \cdot \phi - \theta > \sum_{k \neq j} \phi^{(k)} \cdot \phi \tag{51}$$

where $\theta = \max(\theta_1, \theta_2)$. Similarly, $\phi' = \phi^{(j)}$ provided $\psi^{(j)} \cdot \psi - \theta > \sum_{k \neq j} \psi^{(k)} \cdot \phi$.

Proof: For notational convenience let $\sigma_k = \phi^{(k)} \cdot \phi$ be the similarity of $\phi^{(k)}$ and ϕ. Thus we are assuming

$$\sigma_j - \theta > \sum_{k \neq j} \sigma_k \tag{52}$$

We need to show $\psi' = \psi^{(j)}$, so consider $\Psi\phi$:

$$\begin{aligned}
\Psi\phi &= (\sum_k \psi^{(k)} \times \phi^{(k)})\phi \\
&= \sum_k (\psi^{(k)} \times \phi^{(k)})\phi \\
&= \sum_k \psi^{(k)}(\phi^{(k)} \cdot \phi) \\
&= \sum_k \sigma_k \psi^{(k)}
\end{aligned} \tag{53}$$

This equation is plausible: it says that each $\psi^{(k)}$ contributes to the result to the extent the corresponding $\phi^{(k)}$ is similar to ϕ. From Eq. 53 we have

$$\Psi_t \cdot \phi = \sigma_j \psi_t^{(k)} + \sum_{k \neq j} \sigma_k \psi_t^{(k)} \tag{54}$$

Now we consider the cases $\psi_t^{(j)} = +1$ and $\psi_t^{(j)} = -1$. If $\psi_t^{(j)} = +1$ then

$$\begin{aligned}
\Psi_t \cdot \phi &= \sigma_j + \sum_{k \neq j} \sigma_k \psi_t^{(k)} \\
&\geq \sigma_j - \sum_{k \neq j} \sigma_k \\
&> \theta \geq \theta_1
\end{aligned}$$

Hence $\psi_t' = +1 = \psi_t^{(j)}$. If $\psi_t^{(j)} = -1$ then

$$\begin{aligned}
\Psi_t \cdot \phi &= -\sigma_j + \sum_{k \neq j} \sigma_k \psi_t^{(k)} \\
&\leq -\sigma_j + \sum_{k \neq j} \sigma_k \\
&< -\theta \leq -\theta_2
\end{aligned}$$

Hence $\psi_t' = -1 = \psi_t^{(j)}$. In either case $\psi_t' = \psi_t^{(j)}$. The proof that $\phi' = \phi^{(j)}$ is analogous. □

The preceding proof makes very pessimistic assumptions, namely, that the other $\psi^{(k)}$ are negatively correlated with $\psi^{(j)}$, and thus maximally interfering. In the more likely case, where they are uncorrelated, or even better, where closeness in the $\phi^{(k)}$ implies closeness in the $\psi^{(k)}$, we can get correct recall in spite of the other $\phi^{(k)}$ being similar to ϕ.

Our next goal is to show that the system converges to a stable state. To accomplish this, following Kosko[6] and Hopfield[7], we investigate how a Lyapunov function for the system changes in time. The function is defined:

$$E(\psi, \phi) \ = \ -\tfrac{1}{2}\,\psi \cdot (\Psi\phi - \theta_1) - \tfrac{1}{2}\,\phi \cdot (\psi\Psi - \theta_2) \tag{55}$$

(We write θ_i ambiguously for $\theta_i \mathtt{1}$.) This formula can be understood by observing that $\psi \cdot \Psi\phi$ measures the correlation between ψ and $\Psi\phi$, and $\phi \cdot \psi\Psi$ measures the correlation between ϕ and $\psi\Psi$. Thus E represents the "energy" of the system, which decreases as these correlations increase. Eq. 55 can be simplified to:

$$E(\psi, \phi) \ = \ -\psi\Psi\phi + \psi \cdot \theta_1 + \phi \cdot \theta_2 \tag{56}$$

where we write $\psi\Psi\phi = \psi \cdot \Psi\phi = \psi\Psi \cdot \phi$. We now use this energy function to prove some important results.

Theorem (Monotonicity of Change of Energy): Changes of state always decrease the energy. That is, $\Delta E(\psi) < 0$ and $\Delta E(\phi) < 0$.

Proof: The change in energy resulting from alteration of ψ to ψ' is:

$$\begin{aligned} \Delta E(\psi) \ &= \ E(\psi', \phi) - E(\psi, \phi) \\ &= \ (-\psi'\Psi\phi + \psi' \cdot \theta_1 + \phi \cdot \theta_2) \ - \ (-\psi\Psi\phi + \psi \cdot \theta_1 + \phi \cdot \theta_2) \\ &= \ (-\psi' + \psi) \cdot (\Psi\phi - \theta_1) \\ \Delta E(\psi) \ &= \ -\Delta\psi \cdot (\Psi\phi - \theta_1) \end{aligned} \tag{57}$$

The analysis for the change in ϕ is analogous. Expanding the inner product in Eq. 57 yields:

$$\Delta E(\psi) \ = \ -\int_{\Omega_2} \Delta\psi_t(\Psi_t \cdot \phi - \theta_1)\,\mathrm{d}t \tag{58}$$

We next investigate the integrand.

The updating equation for ψ (Eq. 48) yields the following possibilities:

$$\begin{cases} \Delta\psi_t \in \{0, +2\} & \text{if } \Psi_t \cdot \phi - \theta_1 > 0 \\ \Delta\psi_t = 0 & \text{if } -\theta_2 \le \Psi_t \cdot \phi \le \theta_1 \\ \Delta\psi_t \in \{0, -2\} & \text{if } \Psi_t \cdot \phi + \theta_2 < 0 \end{cases} \tag{59}$$

Since $\Psi_t \cdot \phi - \theta_1 < \Psi_t \cdot \phi + \theta_2 < 0$, note that in all three cases:

$$\Delta\psi_t(\Psi_t \cdot \phi - \theta_1) \ge 0 \tag{60}$$

Furthermore, note that if $\Delta\psi_t \ne 0$ then this inequality is strict. Observe that since $\Delta\psi \ne 0$, this strict inequality must hold for a set of $t \in \Omega_2$ with nonzero measure. This implies the strict monotonicity of $\Delta E(\psi)$:

$$\Delta E(\psi) \; = \; -\int_{\Omega_2} \Delta\psi_t(\Psi_t \cdot \phi - \theta_1)\, \mathrm{d}t \; < \; 0 \tag{61}$$

Hence, until the system stabilizes, every $\phi \rightarrow \psi$ step of the iteration decreases the energy. Analogous reasoning shows that the energy also decreases with the $\psi \rightarrow \phi$ steps. □

We have shown that every step decreases the energy. We now show that the energy cannot decrease forever, because it has a lower bound.

Theorem (Energy Bounded From Below): The energy is bounded from below by a quantity that depends only on Ψ.

Proof: Recall (Eq. 56) that the energy is defined: $E(\psi,\phi) \; = \; -\psi\Psi\phi + \psi \cdot \theta_1 + \phi \cdot \theta_2$. Since Ψ, θ_1 and θ_2 are fixed, there are clearly bipolar ψ and ϕ that minimize this expression.

We can derive a formula for a explicit lower bound as follows. Since $\psi_s \leq 1$ and $\phi_t \leq 1$, we have the inequality:

$$\begin{aligned} \psi\Psi\phi &= \int_{\Omega_1}\int_{\Omega_2} \psi_s \Psi_{st} \phi_t \, \mathrm{d}s \, \mathrm{d}t \\ &\leq \int_{\Omega_1}\int_{\Omega_2} M \, \mathrm{d}s \, \mathrm{d}t \\ &= M|\Omega_1||\Omega_2| \end{aligned}$$

where $M = \max\limits_{st} \Psi_{st}$. Therefore,

$$\begin{aligned} E(\psi,\phi) &= -\psi\Psi\phi + \psi \cdot \theta_1 + \phi \cdot \theta_2 \\ &\geq -M|\Omega_1||\Omega_2| + \psi \cdot \theta_1 + \phi \cdot \theta_2 \\ &\geq -M|\Omega_1||\Omega_2| + \theta_1|\Omega_2| + \theta_2|\Omega_1| \end{aligned}$$

This is our lower bound. It is fixed since it depends only on Ψ. □

The preceding theorems show that the bidirectional associative field memory approaches one of the states characterizing a stored associative pair.

5. CONCLUSIONS

We have argued that AI is moving by necessity into a new phase that recognizes the role of nonpropositional knowledge in intelligent behavior. We also argued that the "new" AI must make use of massive parallelism to achieve its ends. We proposed a definition of massive parallelism, namely that the number of processing elements can be taken as a continuous quantity. We believe that this definition will encourage the development of the necessary theoretical basis for neurocomputers, optical computers, molecular computers, and a new generation of analog computers. We claimed that these computing technologies can be profitably viewed as *field computers,* computers that operate on entire fields of data in parallel. We discussed the importance of general purpose field computers, and related them to universal field computers. This was followed by a theoretical model of field computation, including the derivation of several generalizations of Taylor's theorem for field transformations. These theorems provide one theoretical basis for universal field computers. Finally, we illustrated our theory of field computation by analyzing a continuous field version of Kosko's bidirectional associative memory.

6. ACKNOWLEDGEMENTS

The research reported herein was supported by the **Office of Naval Research** under contract N00014-87-WR-24037.

7. REFERENCES

1. J. A. Feldman and D. H. Ballard, "Connectionist models and their properties," *Cognitive Science,* Vol. 6, pp. 205-254, 1982.

2. B. J. MacLennan, "Logic for the New AI," in *Aspects of Artificial Intelligence,* J. H. Fetzer (ed.), D. Reidel, 1988, in press, pp. 163-192..

3. B. J. MacLennan, "Technology-Independent Design of Neurocomputers: The Universal Field Computer," *Proceedings of the IEEE First Annual International Conference on Neural Networks,* in press, San Diego, CA, June 21-24, 1987.

4. S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by Simulated Annealing," *Science,* Vol. 220 (1983), pp. 671-680.

5. G. E. Hinton and T. J. Sejnowski, "Optimal Perceptual Inference," in *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition* (Washington, D.C., 1983), pp. 448-453.

6. Bart Kosko, "Optical Bidirectional Associative Memories," *Society for Photo-optical and Instrumentation Engineers (SPIE) Proceedings: Image Understanding,* Vol. 758, January 1987.

Bart Kosko, "Competitive Adaptive Bidirectional Associative Memories," *Proceedings of the IEEE First Annual International Conference on Neural Networks,* in press, San Diego, CA, June 21-24, 1987. Bart Kosko, "Adaptive Bidirectional Associative Memories," *Applied Optics,* to appear, November 1987.

7. J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. USA,* Vol. 79, pp. 2554-2558, April 1982. J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Natl. Acad. USA,* Vol. 81, pp. 3088-3092, May 1984.