Continuous Formal Systems: A Unifying Model in Language and Cognition

Bruce J. MacLennan

Computer Science Department
University of Tennessee, Knoxville*

July 3, 1995

1 Introduction

The idea of a *calculus* or *discrete formal system* is central to traditional models of language, knowledge, logic, cognition and computation, and it has provided a unifying framework for these and other disciplines. Nevertheless, research in psychology, neuroscience, philosophy and computer science has shown the limited ability of this model to account for the flexible, adaptive and creative behavior exhibited by much of the animal kingdom. Promising alternate models replace discrete structures by *structured continua* and discrete rule-following by *continuous dynamical processes*. However, we believe that progress in these alternate models is retarded by the lack of a unifying theoretical construct analogous to the discrete formal system.

In this paper we outline the general characteristics of *continuous formal systems* (*simulacra*), which we believe will be a unifying element in future models of language, knowledge, logic, cognition and computation.[1] Therefore, we discuss syntax, semantics, inference and computation in the context of continuous formal systems. In addition, we address an issue that the discrete models were inadequate to address: the gradual emergence of (approximately) discrete structures from a continuum. This is relevant to the emergence of linguistic structures, including semantics and syntax, and to the emergence of rule-like regularities in behavior.

---

*Email: `maclennan@cs.utk.edu`, URL: `http://www.cs.utk.edu/~mclennan`.

To appear in the proceedings of the IEEE Workshop on Architectures for Semiotic Modeling and Situation Analysis in Large Complex Systems, August 27–29, 1995, Monterey, CA.

[1]The reasons for our definitions of continuous formal systems are presented in earlier publications, including MacLennan (1993a, 1994a, 1994b, 1994c, 1994d)

2 Characteristics of Continuous Formal Systems

Since it is easiest to understand continuous formal systems (CFSs, simulacra) in contrast to the traditional, discrete formal systems (DFSs, calculi), for each characteristic I will describe the discrete system first and then the continuous. The two descriptions are parallel, so far as that is possible.

Overall, DFSs are definite: definite in syntax, semantics and computation. In contrast, continuous formal systems may be characterized, for the most part, by replacing "definite" by "continuous" in the description of DFSs. Continuity may be defined as follows: Infinitesimal changes have infinitesimal effects. Therefore:

> *A continuous formal system (*simulacrum*) is continuous in syntax, semantics and computation.*

The following sections explain the consequences of continuity in each of these domains.

2.1 Syntax

In a DFS an expression (or formula) is a definite arrangement of definite tokens. The definiteness of the tokens implies that they are discrete and may be definitely classified as to type. The definiteness of the expression means that we know definitely how the tokens are arranged and whether the arrangement is syntactically correct. All matters of syntax are definite; that is, they may be answered true or false.

As expressions are the concrete representational vehicles of a DFS, so *images* are the concrete representational vehicles of a CFS.[2]

> *An image displays a continuous pattern of bounded variation over a bounded continuum. The well-formedness (syntactic correctness) of an image depends continuously on its pattern of variation. All matters of syntax are continuous; that is, they are matters of degree.*

We will consider some examples of images. In the realm of written communication there are pictures, written language, and diagrams, such as graphs and maps, in which continuous variation has significance. In the foregoing examples variations of intensity, color and texture extend over a two-dimensional region. In other images, such as auditory signals (including speech), variations of intensity and pitch extend over an interval of time. Gestures are three-dimensional images that extend over time and

[2]This terminology is generally consistent with Peirce's semiotics, wherein he distinguishes three kinds of signs: *icons*, *indices* and *symbols*. Icons refer by virtue of their own character or form, that is, by some similarity with their referent. Peirce distinguishes three subkinds of icons: *images*, *diagrams* and *metaphors*. Many of what I call images are images in Peirce's sense, since they correspond to their referents in a direct way; other representations, which I call images (such as Fourier or wavelet transforms), might be better classified as diagrams, since their relationships to their referents are more abstract. See Peirce's *Collected Papers* (2.243–52, 274–302), Buchler (1955, pp. 101–7), and Goudge (1969, pp. 141–3).

three spatial dimensions. Images are also found in the brain: consider the continuous variation of electrochemical activity over the neuropil. (Even if we take discrete synapses to be the sites of activity, the distribution is practically continuous, since there are approximately a billion (10^9) synapses per mm^2 for cortex.) In general, any continuous quantity extended over space, time or other dimensions (e.g. frequency, energy) may serve as the substrate for an image.

2.1.1 Finiteness

Finiteness is an important characteristic of formal systems, whether discrete or continuous, because formal systems are abstractions of mechanical information processing. Such a process is physically realizable only if it uses finite resources (matter and energy). Thus, in a DFS for example, the formulas are required to be finite in length, and the computations should terminate (comprise a finite number of steps). (Consider Turing machines and context-free grammars for examples of the finiteness requirement.)

A reasonable standard for physically realizable images is to require that they be *finite energy functions* (i.e. functions with a finite $\mathcal{L}_2$ norm: $\int_\Omega \phi^2(x)\mathrm{d}x < \infty$). This is a mathematically convenient definition since it means that images are elements of Hilbert spaces. In fact, more restricted definitions are adequate in most cases. For example, images will be finite energy if we require that: (1) the *extent* of an image is finite (i.e., either it is defined over a bounded continuum, or if it defined over an unbounded continuum, then it is 0 outside of some bounded region of that continuum); and (2) the image has bounded variation over its extent. These requirements eliminate images that are either infinitely extended or unbounded in value. The requirements ensure that images can be represented with finite matter or energy. (The finiteness of continuous *computation* will be taken up later.)

2.1.2 Dimension

Another issue relevant to the physical realizability of images is their dimension (the degrees of freedom of their extent). Mathematically, images can extend over any number of dimensions, but practically they are limited to a few dimensions. Physical images can extend over time, over at most three space dimensions, or over various other physical dimensions (frequency, energy, orientation, color, etc.), either singly or in combination. Analogously, although in principle the formulas of a DFS may be of any dimension, in practice they are one-dimensional strings or occasionally two-dimensional arrays of characters. In the discrete case we know this is not a fundamental limitation, since higher-dimensional formulas can be represented as strings, so long as corresponding changes are made to the computational processes.

So also higher-dimensional images may be represented by lower dimensional images without loss of information. Specifically, a band-limited finite-energy image of any dimension can be represented by a finite number of zero-dimensional images, that is, by its generalized Fourier coefficients (or, equally well, by Gabor coefficients, wavelet coefficients, etc.). (If necessary this finite set of zero-dimensional images can be embedded in a single one-dimensional image.) Of course, the computational processes must be altered to operate on this indirect representation. This can always be done in principle, but, compared to the original process, the altered process may be either more or less amenable to mechanical implementation.

In conclusion, the dimensionality of the representational medium does not limit the dimensionality of the images that can be represented in it (perhaps indirectly). For example, human primary visual cortex (area V1), which is physically two-dimensional, represents images extended over at least four dimensions (two spatial, one spatial frequency and one orientation; see MacLennan 1991 for a survey).

2.2 Semantics

The issue of syntactic correctness is one of interpretability. In traditional logic only the syntactically correct expressions are interpretable. In continuous logic, images are interpretable to the extent that they are well-formed. Continuity does not preclude there being some images that are entirely uninterpretable and others that are entirely interpretable, but there must also be intermediate images with intermediate degrees of interpretability. So also for well-formedness.

In a DFS every well-formed expression has a definite meaning. In particular, each token type has a definite meaning, and the meaning of an expression is a definite function of the arrangement and meanings of the constituent tokens. In contrast:

> *The meaning of an image depends continuously on the form of the image, and an image has a meaning to the degree that it is well-formed (syntactically correct).*

For example, normalized two-dimensional vectors might be interpreted to represent orientations, but the degree of well-formedness, and hence interpretability, must drop *continuously* to zero as the vector's magnitude deviates from 1 (Fig. 1).[3]

For a less abstract example, suppose that the well-formed images are the Times Roman letters 'T' and 'F', which are interpreted as **true** and **false**, respectively. As an image deviates more from these archetypes, it becomes progressively less interpretable, and may not be interpretable at all (Fig. 2). It will be apparent that the well-formed images are a fuzzy set (which may be defined by a "continuous grammar," which is described below.)

Given the membership functions μ_T and μ_F for the T and F sets (assumed disjoint), we can define a formal interpretation function. Suppose that the domain of

[3]Of course, there is nothing that stops us from interpreting *any* nonzero vector as an orientation. The point is that we may, for some reason, choose to limit the well-formed vectors to normalized vectors; for example, a computational process might require normalized input.

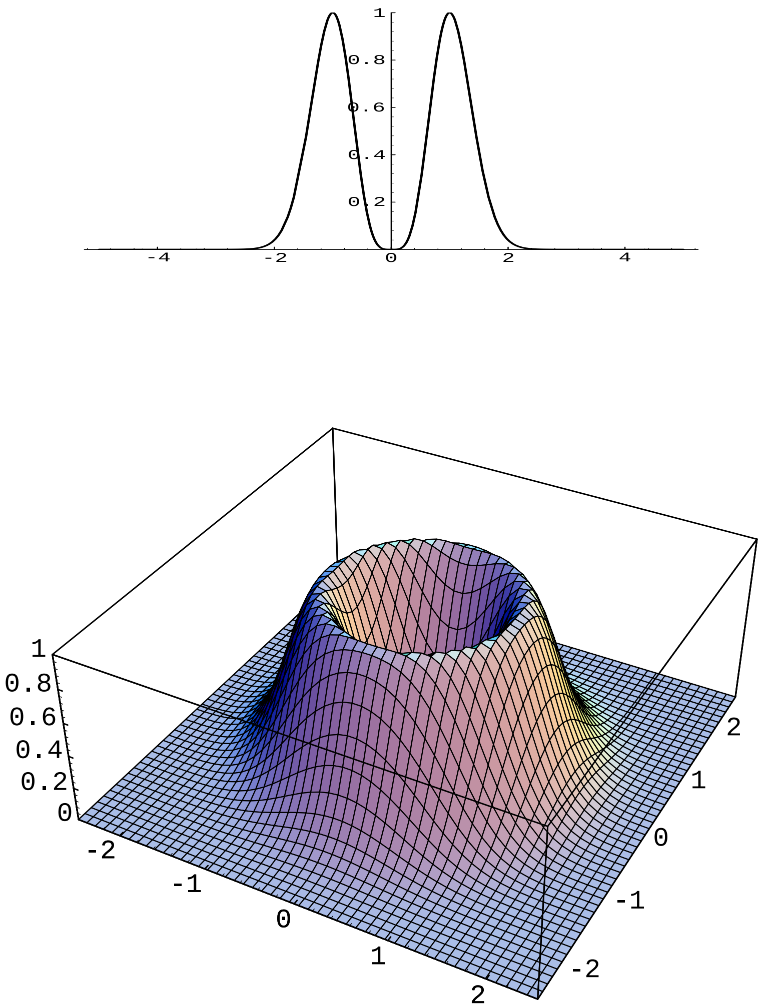

Figure 1: Example of degrees of well-formedness of two-dimensional vectors, wherein normalized vectors are considered perfectly well-formed. In this example the degree of well-formedness is measured by $f(\|\mathbf{v}\|^2)$, where $f(x) = x^2 e^{2(1-x)}$.

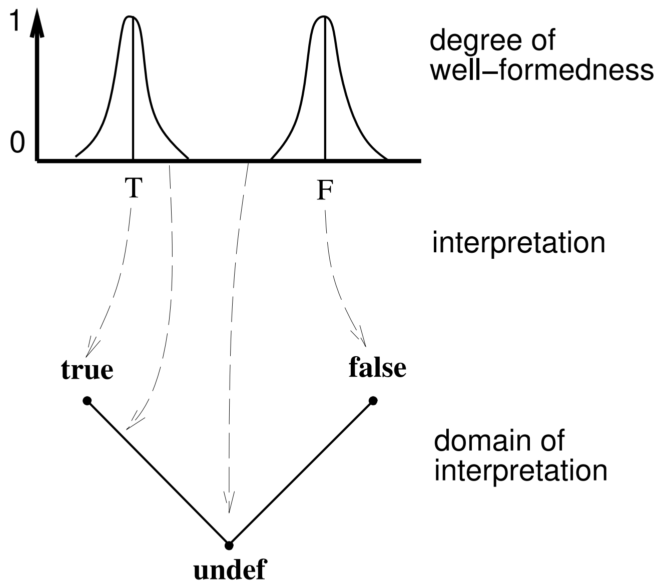

Figure 2: Inherent fuzziness of syntactic well-formedness. Just as the degree of well-formedness decreases continuously from 1 to 0, so also the interpretation must vary continuously from a defined interpretation to an undefined interpretation.

interpretation includes **true**, **false** and **undef**, as well as linear combinations of **undef** with the other two (Fig. 2). The interpretation of an image ϕ is defined:

$$\mathcal{I}(\phi) = \mu_{\mathrm{T}}(\phi)\mathbf{true} + \mu_{\mathrm{F}}(\phi)\mathbf{false} + [1 - \mu_{\mathrm{T}}(\phi) - \mu_{\mathrm{F}}(\phi)]\mathbf{undef}.$$

As required, this function satisfies:

$$\begin{aligned} \mathcal{I}(\text{`T'}) &= \mathbf{true}, \text{ since } \mu_{\mathrm{T}}(\text{`T'}) = 1, \\ \mathcal{I}(\text{`F'}) &= \mathbf{false}, \text{ since } \mu_{\mathrm{F}}(\text{`F'}) = 1, \\ \mathcal{I}(\phi) &= \mathbf{undef}, \text{ if } \mu_{\mathrm{T}}(\phi) = 0 \text{ and } \mu_{\mathrm{F}}(\phi) = 0. \end{aligned}$$

For imperfectly-formed images (e.g., $0 < \mu(\phi) < 1$) the interpretation will be between **undef** and the interpretation of the corresponding perfectly-formed image.

If D is the domain of interpretation of a DFS, it can be made the domain of interpretation of a CFS as follows. Let the unit interval $U = [0, 1]$ represent degrees of well-formedness. Then a pair $(x, y) \in U \times D$ combines a degree of interpretability x and the interpretation y of a perfectly-formed image. Since all uninterpretable images are equivalent, define an equivalence relation on $U \times D$:

$$E[(x_1, y_1), (x_2, y_2)] \equiv (x_1 = x_2) \wedge (x_1 = 0 \vee y_1 = y_2).$$

Thus all pairs (interpretations) with a zero first component are equivalent, no matter what their second component. Finally we may define $\mathcal{D}$, the continuous domain of interpretation corresponding to D, as the quotient set $\mathcal{D} = (U \times D)/E$ of equivalence classes under E. Then **undef** $= [(0, y)]$, the equivalence class of interpretations of completely ill-formed images. We can see that $\mathcal{D}$ has a continuum of interpretations between **undef** and each of the members of D (the interpretations of perfectly-formed images).

If $I(\text{'x'})$ is the interpretation of 'x' in a DFS, and if in a corresponding CFS 'x' has a fuzzy set with membership function μ_{x}, then the interpretation of an image ϕ in this set (i.e., $\mu_{\text{x}}(\phi) \neq 0$) is

$$\mathcal{I}(\phi) = [(\mu_{\text{x}}(\phi), I(\text{'x'}))].$$

Note that a "propositional image," that is, an image with a truth value, must have a continuous domain of interpretation, but this is not the same as saying it has a continuous truth value (as in fuzzy logic). As depicted in Fig. 2, there may be only two truth values (**true** and **false**), but varying degrees of interpretability (corresponding, for example, to confidence of interpretation). The interpretation is **undef** in the absence of any confidence.

2.3 Computation

In a DFS the computational process, by which one expression is derived from another, is definite; that is, the computation comprises a definite sequence of discrete steps of definite type. At each step it is definite which rule (type of computation) may be applied, which is definitely constrained by the syntax of the expression, and the result of the step is a definite result of the expression to which the rule is applied. On the other hand:

> *In a CFS the computational process, by which one image is derived from another, is continuous; that is, the computation comprises a continuous process of continuously changing direction. At each point the infinitesimal change effected by the process is a continuous function of the form (shape) of the image, and the result of the change is a continuous function of the image at that time.*

As in a DFS, computation depends on the form, but not the meaning, of the image.

Examples of continuous computations include: continuous approximations, continuous optimization processes (such as hill-climbing), continuous control (e.g., robotics, sensory-motor coordination), continuous deformations and transformations, continuous mental image manipulation (as in Roger Shepard's well-known experiments), gradual learning processes, and computation on analog computers.[4]

[4]A more detailed discussion of the relation of continuous (analog) and discrete (digital) computation, including an exploration of the philosophical issues, can be found in earlier publications (MacLennan 1993b, 1994b, 1994c, 1994d).

Formally, a continuous computational process has at each time t a state ψ_t, which is an image drawn from a continuum. In a deterministic computation the state at any time interval u in the future is a continuous function of the current state, that is,

$$\psi_{t+u} = P(\psi_t, u).$$

Clearly, the process function P satisfies the group properties:

$$\begin{aligned} P(\psi, 0) &= \psi, \\ P[P(\psi, u), v] &= P(\psi, u + v). \end{aligned}$$

In all ordinary cases the state images will form a linear space, which means that we can differentiate them with respect to time:

$$\dot{\psi}_t = \lim_{u \to 0} \frac{\psi_{t+u} - \psi_t}{u} = \left. \frac{\partial P(\psi_t, u)}{\partial u} \right|_{u=0}.$$

Thus, a deterministic continuous computation can be expressed by a differential equation, $\dot{\psi}_t = Q(\psi_t)$, where $Q(\psi) = \partial P(\psi, u)/\partial u|_{u=0}$. Further, continuous computations (such as control processes) can depend, in addition, on one or more input signals (images) ϕ_t, e.g., $\dot{\psi}_t = Q(\psi_t, \phi_t)$.

2.3.1 Termination

There are ordinarily two ways in which computation in a DFS can halt. First it may enter a state to which no rules are applicable; thus no further state change is possible. Second, the computation may enter one of a designated set of terminal states, which signal the completion of the computation; no attempt is made to apply rules to terminal states. In a well-structured DFS the two kinds of termination ordinarily coincide; that is, it is arranged so that a state is terminal if and only if no rules are applicable to it. In any case, whether a computation has terminated is always a definite matter.

> *In a CFS there are two ways a computation can terminate. First, the computation may asymptotically approach or reach a* point attractor, *in which case the computation is said to have* stabilized. *Second, the computation may enter a designated subspace of terminal states, in which case no further computation takes place; such a process is said to have* completed. *Ordinarily the two methods of termination coincide; that is, the computation stabilizes if and only if it completes by reaching a terminal state.*
>
> *In either case, the computation approaches termination continuously (and so, in effect, termination is a matter of degree).*

Figure 3 illustrates the two kinds of termination.

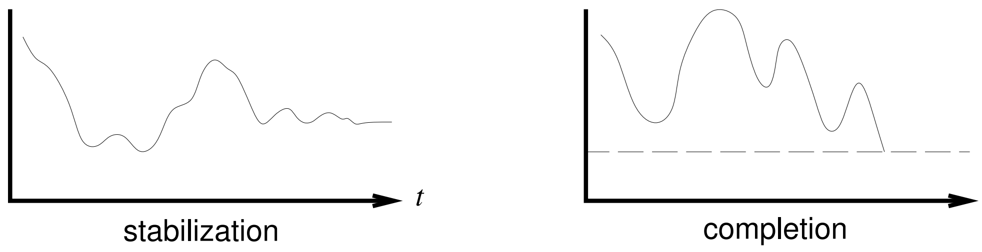

Figure 3: Kinds of termination of continuous computation. The diagram on the left depicts termination by *stabilization*: the state asymptotically approaches or reaches a point attractor. The diagram on the right depicts termination by *completion*: the state enters a predesignated set of *terminal states* (in this example, those below the dotted line).

2.3.2 Programs

In a DFS the permissible types of computation at each step are given by rules, which together constitute an expression (as defined above); this expression is called a *program* for the computation. Further, there is a simple, mechanical relationship between the rule set and each step of the computational process.

It is difficult to be more precise than this, when the variety of traditional formal systems is considered. Take for examples the predicate calculus, Turing machines, the lambda calculus, Markov algorithms, Post productions, combinatory logic, recursive functions and digital computers. Almost any definite process is permissible provided it is mechanical, i.e. can be carried without the exercise of intelligence.

> *In a CFS the moment-to-moment changes are guided continuously by an image, called a* guiding image *for the computation.*
>
> *There is a simple, mechanical relationship between the guiding image and the trajectory of the computed image.*

The notion of a guiding image is especially interesting, since it amounts to a program "written" (better: "sculpted") in a continuous language. It is difficult to be more precise about the mechanical relationship, except that it should not require intelligence, and we should be able to see how, at least in principle, it could be implemented by a machine or physical process.

A simple example of a guiding image is a potential surface. Gradient descent is a deterministic computation guided by the image. Formally, if the potential surface P is the guiding image, then the computation is given by $\dot{\psi} = -r\nabla P(\psi)$. Of course, the trajectory of the computation also depends on its initial state ψ_0, which may represent the input to the computation.

More interesting guiding images determine the trajectory in a less direct way, e.g., by defining Fourier coefficients, the coefficients of a differential equation, or the boundary conditions of partial differential equations. For another example, consider any one-dimensional image ϕ such that $x\phi_x > 0$ for $x \neq 0$. It is the guiding image of the computation $\ddot{x} = -\phi(x)$, which causes x to oscillate in a manner determined by ϕ.

For a final example, Jonathan Mills (1995) has developed *Kirkhoff machines*, which solve the two-dimensional diffusion equation ($\partial\psi/\partial t = a^2\nabla^2\psi$) given guiding images in the form of boundary conditions, sources and sinks. The computation is implemented by the actual diffusion of charge carriers in a spatially-continuous semiconducting mass.

2.3.3 Nondeterministic Computation

In a DFS a nondeterministic computational process defines sequences of allowed (discrete) changes to an expression, which permit one expression to be derived from another, in accord with a rule set. In the usual formulation, inference rules constrain, but do not determine, the steps of a (valid) derivation (computation).

> *In a CFS a nondeterministic computational process associates degrees of facility with possible infinitesimal changes in an image; the facilities of changes are determined by the guiding image. The probabilities of various computational trajectories depend on their cumulative facilities.*

Notice that continuity requires that there be a continuum between allowed and disallowed trajectories. For example, a nondeterministic computation might permit any energy-decreasing trajectory, with facility being determined by the rate of decrease.

Specifically, suppose the potential surface P is the guiding image of the computation. The infinitesimal energy change $\dot{P}$ resulting from an infinitesimal state change $\dot{\psi}$ is given by $\dot{P}(\psi) = \nabla P(\psi) \cdot \dot{\psi}$. Since we require $\dot{P}(\psi) < 0$ whenever $\dot{\psi} \neq 0$, we can define the facility of change $\dot{\psi}$ from state ψ as follows:

$$F(\psi, \dot{\psi}) = \left[-\nabla P(\psi) \cdot \dot{\psi}\right]^+,$$

that is, the positive part of $-\nabla P(\psi) \cdot \dot{\psi}$. Thus, $F(\psi, \dot{\psi}) > 0$ to the extent that $\dot{\psi}$ decreases energy, and $F(\psi, \dot{\psi}) = 0$ if $\dot{\psi}$ would increase energy or leave it unchanged (Fig. 4). Indeed, we can see that the facility of an infinitesimal change is proportional to the angle between the change and the negative gradient of the guiding image:

$$\begin{aligned} F(\psi, \dot{\psi}) &= \left[\| - \nabla P(\psi)\| \, \|\dot{\psi}\| \cos\theta\right]^+, \\ &\propto \|\nabla P(\psi)\| \cos\theta, \quad \text{for } -\pi/2 \leq \theta \leq \pi/2. \end{aligned}$$

2.4 Grammars

In a DFS a *generative grammar* is a rule set describing a nondeterministic computational process capable of generating from a single token of fixed type any and only

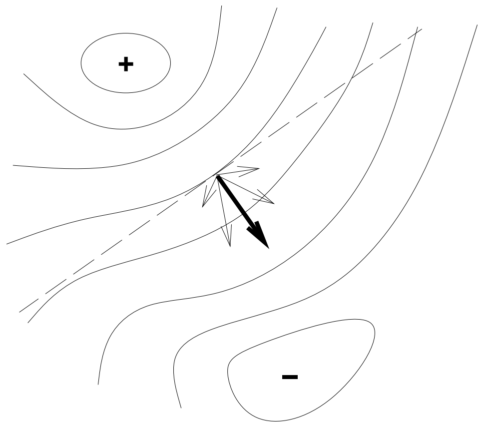

Figure 4: Nondeterministic computation by descent on potential surface. The facility of change in a given direction is shown by the length of the arrow (which is proportional to the cosine of its angle with the negative gradient. The dotted line separates impossible changes to the left from possible changes to the right.

the well-formed expressions, which are among the set of terminal states. The process may be made deterministic by specifying, by an expression, the choice of rule to be applied at each step. A *recognitive grammar* is an computational process that reduces any arrangement of tokens of the allowed types to a single token of one of two types, meaning that the arrangement is or is not well-formed.

> *A* continuous generative grammar *is the guiding image of a nondeterministic computational process capable of generating, with varying facility, terminal images from a fixed starting image. There will be a continuum between images generated by the grammar and those not so generated, so well-formedness is a matter of degree. The process may be made deterministic by specifying, by an image, the direction of change at each instant.*
>
> *A* continuous recognitive grammar *is a computational process that reduces an image to a real number in* $[0, 1]$ *representing its degree of well-formedness.*

A continuous recognitive grammar is an example of a guiding image (continuous program) for a (fuzzy) decision problem.

A simple example of a continuous grammar uses the nondeterministic descent on the potential surface in Fig. 5 to generate normalized two dimensional vectors from the starting image $(0, 0)$. In a simple example such as this the well-formedness surface can be used directly as the guiding image, that is, as the "grammar." The "energy" of the terminal image represents its degree of well-formedness (zero energy = perfectly-formed, higher energy = less-well-formed).

A more complex continuous grammar, comparable to a regular expression, is shown in Fig. 6. The state $\psi_t = (y_t, a_t)$ circles at a constant rate $\dot{a} = r$, and any

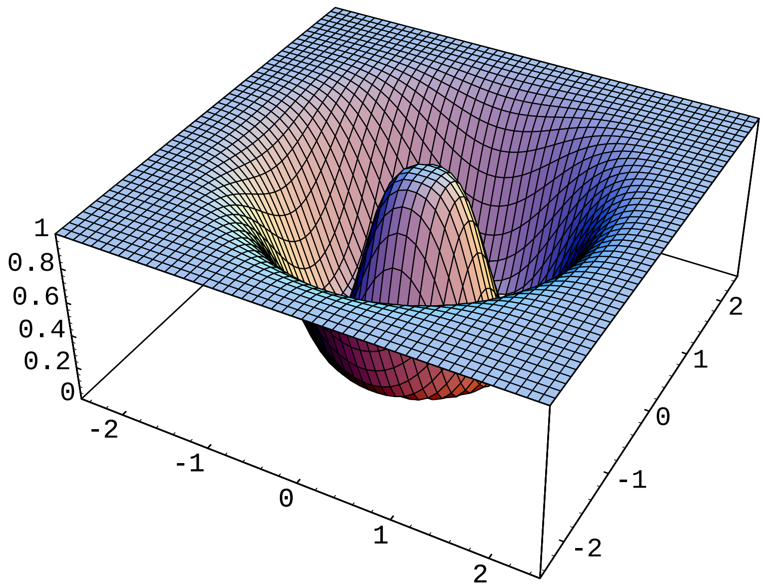

Figure 5: A simple continuous grammar. Nondeterministic descent on this potential surface from $(0, 0)$ generates normalized two-dimensional vectors.

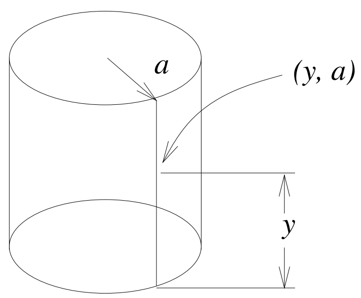

Figure 6: Continuous grammar for generating damped sinusoids.

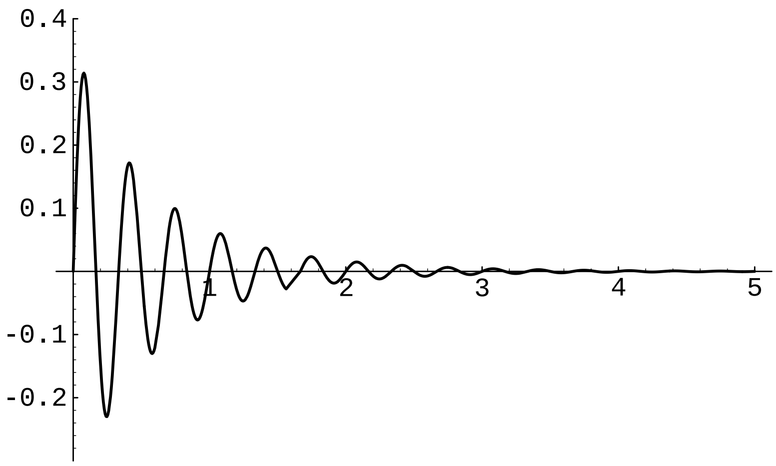

Figure 7: Typical damped sinusoid generated by simple continuous grammar

potential-decreasing path is allowed (i.e., $\dot{y} \leq 0$, and $\dot{y} = 0$ only if $y = 0$). The terminal images are at the bottom of the cylinder. Thus this nondeterministic computation sequentially generates damped sinusoids of the form $\phi_t = y_t \sin a_t$ where $0 \leq y$ and $\dot{y} \leq 0$ (Fig. 7). Similar techniques can be used for generating nested images.

There are of course many mechanisms by which nondeterministic computation can generate terminal images under the control of guiding images, just as there are many different discrete grammatical formalisms (e.g., regular expressions, context-free grammars, various "normal forms"). So also, important classes of continuous "languages" (fuzzy sets of well-formed images) may have different grammatical mechanisms that generate them.

3 Objects, Time and Change

Certainly, language and cognition sometimes resemble a DFS, and the dominant paradigm in linguistics, cognitive science and artificial intelligence has been that they *are* DFSs. Although this paper is based on the premise that CFSs provide better models of language and cognition, we must nevertheless ask how it is that a CFS can look enough like a DFS that investigators have been fooled into thinking that language and cognition are discrete. In one sense the answer is trivial: one can design a CFS that approximates a given DFS as closely as one likes; and vice versa, a DFS can be found that approximates a given CFS to any desired degree. However, the more interesting issue is: How do (approximately) discrete formal systems emerge from continuous representations by means of continuous computation? The answer to this question is a step towards understanding the origin of apparently discrete faculties, such as language and formal reasoning, from the underlying continuous sensory-motor processes.

3.1 Objects

An important part of this process is the perception of *objects*; indeed, objects are the prototypes of the tokens manipulated by calculi, as we can see even in the etymology of the word (Latin *calculus* = small stone used in counting, voting, games, etc.). An object can be defined as a stable bundle of properties (cf. elementary particles in physics as bundles of quantum numbers). That is, an object retains its properties when its context is changed, which is what allows an object to be separated from its background. In other terms, an object is equivalent to a set of *invariances* (MacLennan 1994b, 1994c). For example, the spatial relations among the parts of a rigid object are invariant in spite of its motion; also, a melody has an invariant pitch-contour in spite of the key into which it's transposed. To understand how stable (or invariant) bundles of properties can be detected, we must consider how CFSs represent change (for invariance presupposes variation).

3.2 Gabor Representations

I think there is a fallacy in attempting to reduce motion to space and time (or change to some measurable property and time). Zeno showed us long ago the paradoxes inherent in the concept of instantaneous velocity (the derivative). More recently Einstein in his special theory of relativity showed the inseparability of space and time; indeed, relative motion is the primary, from which the time and space axes derive their position.

Consider a changing image such as a sound. In the time domain such a signal can be considered a variation of amplitude extended over time, but in the frequency domain it may be considered a spectrum: variations of amplitude (and phase) extended over frequency.

Gabor (1946) observed that neither view accurately reflects our perception of sound, which is simultaneously of pitch and duration (frequency and time). By applying the mathematics of the Heisenberg Uncertainty Principle, he showed how any finite signal (i.e. signal of finite duration and bandwidth) could be decomposed into a finite number of "elementary signals," now known as Gabor functions.[5] Each elemen-

[5] MacLennan (1991) provides an intuitive presentation of Gabor's proof.

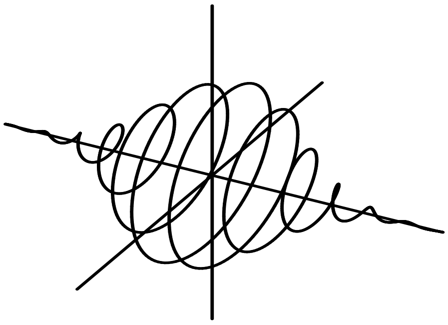

Figure 8: Gabor elementary function, which is a complex-valued function of time (or any other real quantity). The time axis extends from left to right through the center of the spiral; the imaginary axis is vertical and the real axis is horizontal.

tary signal represents the maximum localization in time and frequency permitted by the uncertainty principle. That is, rather than reducing a signal to either the time or frequency domain, Gabor reduces the signal to elementary conjuncts of time and frequency. Gabor's elementary functions differ from other elementary functions in that they are localized in both time and frequency; in contrast sinusoidal basis functions are not localized in time, and radial basis functions are not localized in frequency.

To put it differently, Gabor's analysis shows us how a finite sound is composed of a finite number of "elementary sounds," which is more informative than either the time or frequency domain descriptions. In a similar way we can decompose a time varying image into a set of "elementary changes" from which we may detect the covariances and contravariances that separate an object from its context. Next we'll consider one way of accomplishing this separation.

As a starting point consider a one-dimensional image extended in time, ϕ_t. The Gabor transform of this will be a two-dimensional image extended in time and space (representing frequency), Φ_{tf}. The magnitude of Φ_{tf} measures the degree to which at time near t the image contains frequencies near f. Formally, the Gabor transform is given by the inner product $\Phi_{tf} = \langle \phi, \gamma_{tf}^{(\beta)} \rangle$, where $\gamma_{tf}^{(\beta)}$ is a Gabor function spread (as determined by β) around t and f (Figs. 8-9):

$$\gamma_{tf}^{(\beta)}(\tau) = e^{-\pi(\tau-t)^2/\beta^2} e^{2\pi i f \tau}.$$

The first exponential is a Gaussian with standard deviation $\beta/2\sqrt{\pi}$; that is, β is proportional to its spread. The second, complex exponential is the conjugate exponential form of the trigonometric functions, and is periodic with frequency f.

The Gabor transform is not limited to temporal images. If ϕ_x were an image extended in space (or some other dimension), then its Gabor transform Φ_{xu} represents the presence in the image at locations near x of spatial frequencies near u.

Now suppose we have a two-dimensional image ϕ_{xt} extended over space and time, that is, a time-varying one-dimensional image. Its Gabor transform is a four-dimensional image Φ_{xutf}, where x is spatial location, u is spatial frequency, t is time

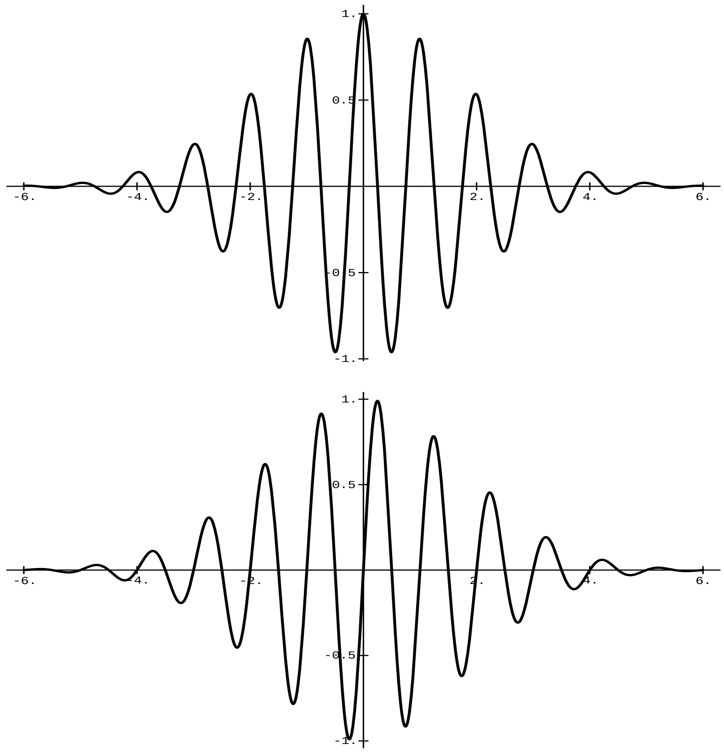

Figure 9: Real and imaginary components of Gabor elementary function. The upper diagram is the real component, a Gaussian-modulated cosine; the lower diagram is the imaginary component, a Gaussian modulated sine.

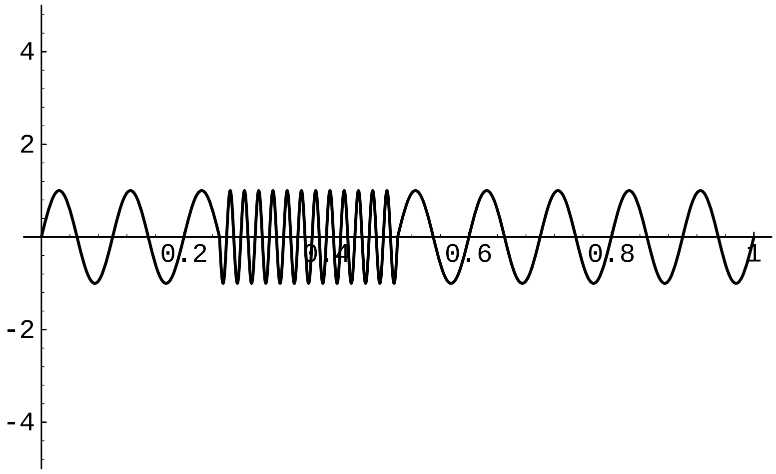

Figure 10: Object detection example. An image of an object with a high spatial frequency moving in front of a background with a low spatial frequency.

and f is (temporal) frequency. Thus the magnitude of Φ_{xutf} measures the degree to which at locations near x and times near t there is a grating of frequency near u oscillating at a rate near f. The two-dimensional Gabor transform is given by the inner product $\Phi_{xutf} = \langle \phi, \gamma_{xutf}^{(\alpha\beta)} \rangle$, where the two-dimensional Gabor elementary function is defined:

$$\gamma_{xutf}^{(\alpha\beta)}(\xi, \tau) = \gamma_{xu}^{(\alpha)}(\xi)\ \gamma_{tf}^{(\beta)}(\tau).$$

To see how the Gabor transform facilitates object detection, consider the Gabor representation of an image of an object with spatial frequency u_{o} moving at a velocity v_{o} across a background with a spatial frequency u_{b} (Fig. 10). The background will cause energy to be concentrated in $\Phi_{xu_{\mathrm{b}}t0}$ for any places x and times t where it is not occluded by the moving object. The object will cause energy to be concentrated near $\Phi_{xu_{\mathrm{o}}tf_{\mathrm{v}}}$, where $f_{\mathrm{v}} = u_{\mathrm{o}}v$, for all locations x and times t where the object occludes the background. Figure 11 depicts $\Phi_{xu_{\mathrm{o}}tf}$ at fixed t; it shows how the Gabor representation Φ separates the moving object from its background.

I have said vaguely that Φ_{xutf} measure movement "near" x, u, t and f. How near? This is given by the Gabor Uncertainty Principle, which is just the Heisenberg Uncertainty Principle applied to arbitrary finite signals. If Δx represents any function's localization in space, Δt its localization in time, Δu its localization in spatial

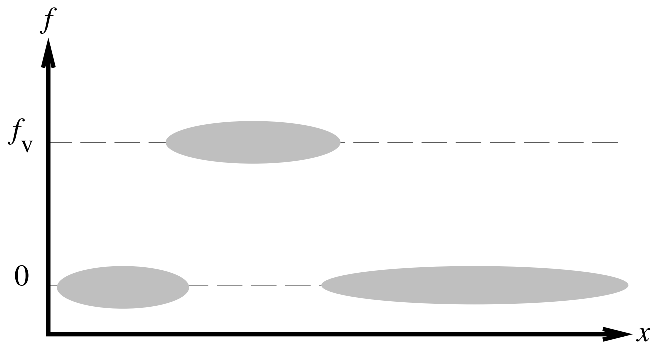

Figure 11: Depiction of Gabor representation of object moving across background. Activity at f_v represents the moving object, activity at $f = 0$ represents the background.

frequency, and Δf its localization in temporal frequency, then we must have (Fig. 12):

$$\begin{aligned} \Delta x \, \Delta u &\geq 1/4\pi, \\ \Delta t \, \Delta f &\geq 1/4\pi. \end{aligned}$$

Thus, if change is better localized in time, it is more poorly localized in temporal frequency (and hence velocity), and vice versa; the same applies for space and spatial frequency. We can trade localization in one variable for localization in its conjugate variable, but the joint minimum uncertainty cannot be better than $1/4\pi$.[6]

The uncertainties Δx, Δt, Δu and Δf, are determined by the shape parameters, α and β, of the Gabor elementary function:

$$\begin{aligned} \Delta x &= \alpha/2\sqrt{\pi}, \\ \Delta u &= \alpha^{-1}/2\sqrt{\pi}, \\ \Delta t &= \beta/2\sqrt{\pi}, \\ \Delta f &= \beta^{-1}/2\sqrt{\pi}. \end{aligned}$$

Thus α determines the most something can be localized in space and therefore the minimum size objects that can be detected; conversely it determines how acurately "textures" (spatial frequencies) are distinguishable, which affects its ability to separate objects from the background. In effect the choice of α and β determines the mesh of the perceptual fish-net, and therefore the size of the fish that inevitably slip through it.[7]

[6] MacLennan (1991) provides an intuitive presentation of Gabor's proof.

[7] Specifically Δx, Δt, Δu and Δf are the standard deviations of the Gabor elementary function over each of its dimensions (both extension and frequency).

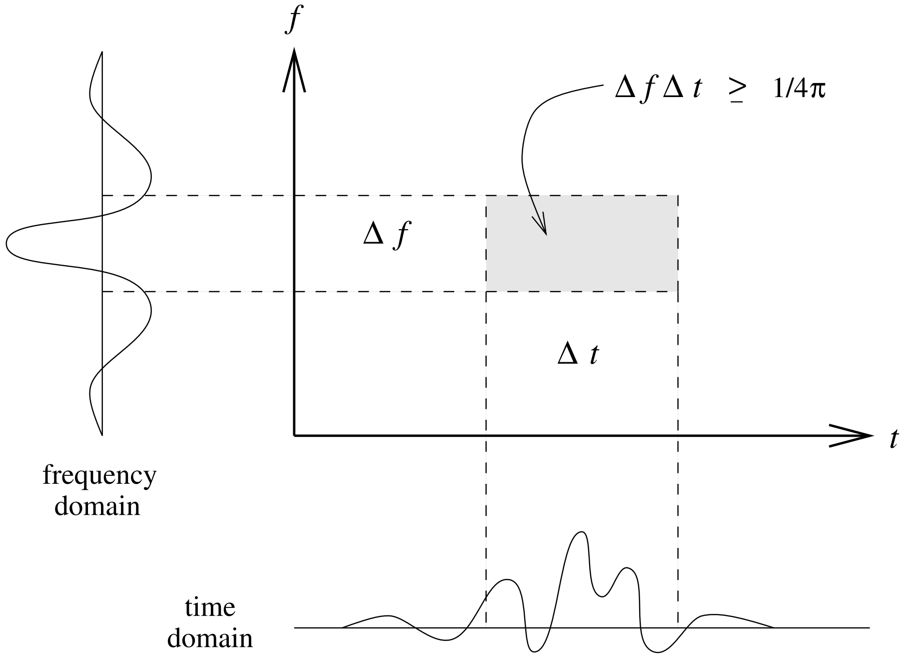

Figure 12: Minimum joint localization in time and frequency domain given by Gabor Uncertainty Principle.

3.3 Multiresolution Representations

One solution to the fixed resolution of the Gabor representation is to use a *multiresolution representation*, which uses different tradeoffs between conjugate variables at different scales. In particular, smaller scale features are associated with higher frequencies and therefore with a wider tolerance in uncertainty in the frequency domain. Figure 13 compares the division of "Fourier space" by the Gabor transform and multiresolution analyses. *Wavelet analysis* produces such a multiresolution representation by scaling a single "mother wavelet," usually by factors of 2, to achieve differing scale sensitivities. One common mother wavelet, the Morlet wavelet, which is a slight modification of the Gabor elementary function, is defined:

$$\psi_{xu}(\xi) = \frac{1}{\sqrt[4]{\pi}} \left(e^{2\pi i u \xi} - e^{-u^2/2} \right) e^{-(\xi - x)^2/2}$$

The one-dimensional Morlet transform is then given by the inner product $\Phi_{xu} = \langle \phi, \psi_{xu} \rangle$.

4 Emergence of Rules

Certainly, people and other animals sometimes exhibit rule-like behavior, and until recently both cognitive science and artificial intelligence made the assumption that

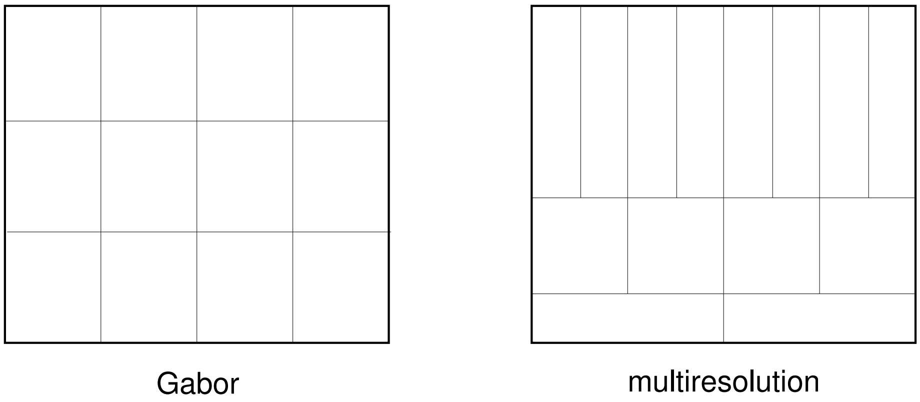

Figure 13: Comparison of Gabor and multiresolution representations. The Gabor transform divides "Fourier space" (frequency cross time) into identical cells, whereas multiresolution transforms, such as wavelet transforms, allow the frequency/time localization tradeoff to scale with frequency. At larger scales, time resolution is worse, but frequency resolution is better, and vice versa.

rule-like behavior is a consequence of following rules. Now connectionism has shown how rule-like behavior can emerge from processes that are not following rules. In addition, it demonstrates the possibility of *approximately rule-like* behavior, which exhibits rule-like behavior when it is appropriate, but is able to bend or adapt the rules when that is more effective. In this way we can begin to understand and imitate the *flexible* rule-like behavior of the natural world. We'll take a brief look at the emergence of rule-like behavior from the perspective of CFSs.

In typical rule-like behavior the situation (the "input") is classified into one of a small number of cases. From each of these cases a small amount of *index information* is extracted, which is sufficient to identify the particulars to which the action for that case applies.[8] Thus a rule takes the form: "If some *things* are in this kind of situation, then take this corresponding action regarding those *things*." Thus the response (or "output") depends only on the (low-dimensional) classification of the situation and the (low-dimensional) indices required to identify the objects to which it applies.

In other words, a rule reduces a situation, with its (perhaps) complex internal relationships, into a simple classification and the indices. Aside from the particulars represented by the indices, the entire situation is reduced, in effect, to a point, so the rule cannot be sensitive to any aspects of the situation not represented in its classification (i.e., in the low-dimensional features extracted from the situation). Thus a rule cannot be context-sensitive, that is, sensitive to aspects of the situation not explicitly represented in the low-dimensional features. Other than the particularization of the actions by the indices, the number of possible actions (responses) is limited to the

[8] An *index* is a sign that points at an object (Peirce's terminology; see footnote 2). For concreteness, think of it as the object's coordinates.

number of kinds of situations; there can be no continuous sensitive dependence of the response on the nuances of the situation.

Behavior appears rule-like to the extent that the functional dependence of outputs on inputs can be factored through a low-dimensional space (representing the kind of situation and the indices). However, it is not necessary that rule-like behavior actually be generated in his way, that is, by means of an intermediate low-dimensional representation. Indeed, flexibility depends on avoiding this representational bottleneck, either by there being no intermediate representation, or by the intermediate representation being of comparatively high dimension. Then we have the possibility of flexibility and adaptation for, as the situation demands, the representation can expand beyond the low-dimensional subspace (which manifests in the rule-like behavior) of the intermediate space, perhaps later settling into a different low-dimensional subspace (and therefore constellating as different rule-like behavior).

5 Conclusions

We have shown that the concept of a *continuous formal system* (or *simulacrum*) is analogous to a discrete formal system (or calculus), but that the former is continuous where the latter is definite. Nevertheless, the theory of CFSs forces revisions in our notions of syntax, semantics, computation, program, grammar and rule. It promises a new theory of representations in artificial intelligence, cognitive science, linguistics and philosophy, which can better address the emergence of discrete entities, such as objects and rules, from the underlying continuous processes. In particular, we have considered how a multiresolution representation of change can aid the detection of discrete structures. It is our hope that continuous formal systems will provide a theoretical framework for understanding emergent and adaptive information processing in all its manifestations.

6 References

Buchler, Justus (ed.). (1955). *Philosophical Writings of Peirce.* New York, NY: Dover. (Previously published as *The Philosophy of Peirce: Selected Writings.* London: Routledge and Kegan Paul, 1940.)

Gabor, D. (1946). Theory of Communication. *Journal of the Institution of Electrical Engineers*, vol. 93 (III), pp. 429–457.

Goudge, Thomas A. (1969). *The Thought of C. S. Peirce* New York: Dover. (Originally published by University of Toronto Press, Toronto, 1950.)

MacLennan, B. J. (1991). Gabor Representations of Spatiotemporal Visual Images. Technical report CS-91-144, University of Tennessee, Knoxville, Computer Science Department. Accessible via URL: `http://www.cs.utk.edu/~mclennan`.

MacLennan, B. J. (1993a). Characteristics of Connectionist Knowledge Representation. *Information Sciences*, vol. 70, pp. 119–143.

MacLennan, B. J. (1993b). Grounding Analog Computers. *Think*, vol. 2, pp. 48–51, 74–77.

MacLennan, B. J. (1994a). Continuous Symbol Systems: The Logic of Connectionism. In: D. S. Levine and M. Aparicio IV (Eds.), *Neural Networks for Knowledge Representation and Inference*, pp. 83–120. Hillsdale: Lawrence Erlbaum.

MacLennan, B. J. (1994b). Image and Symbol: Continuous Computation and the Emergence of the Discrete. In: Vasant Honavar and Leonard Uhr (Eds.), *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration, Volume I: Basic Paradigms; Learning Representational Issues; and Integrated Architectures*, pp. 207–240. New York: Academic Press.

MacLennan, B. J. (1994c) Continuous Computation and the Emergence of the Discrete. In: Karl Pribram (Ed.), *Origins: Brain & Self-Organization*, pp. 121–151. Hillsdale: Lawrence Erlbaum.

MacLennan, B. J. (1994d). Words Lie in Our Way. *Minds and Machines*, vol. 4, pp. 421–437.

Mills, Jonathan W. (1995), Kirkhoff Machines. Technical report, Indiana University, Bloomington, Computer Science Department.