Chapter 6: Computational Experiments

EXPERIMENT GROUP 1: SAMPLED TRANSIENTS OF COUPLED MAPS

Dynamical systems theory inherits from analytical mathematics an emphasis on asymptotic states, even though analytic solutions remains elusive for nonlinear, and particularly high dimensional nonlinear systems. Accordingly the literature on the time evolution of transients in essentially non-existent, though certain phenomena have been noted. For example, most treatments of bifurcations in period-doubling maps mention the phenomena of *critical slowing down*, in which the average time to reach the attractor increases in the vicinity of a branching critical point, where the attractor loses stability.

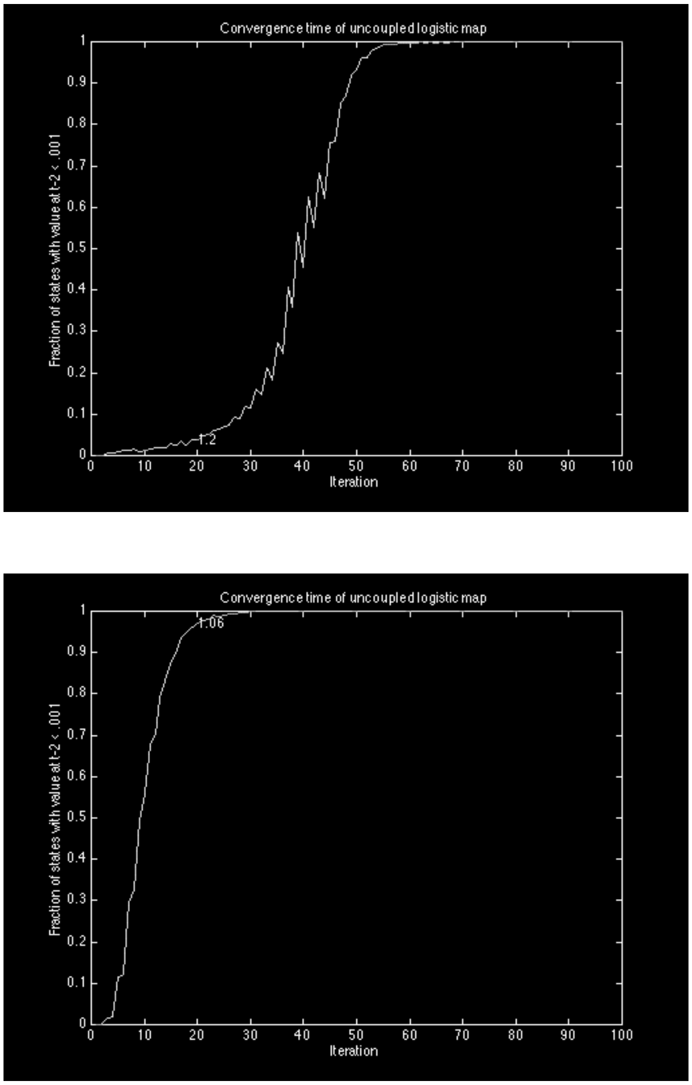
In order to study the evolution of transients in maps, several sets of simulations on asymmetric logistic maps were performed. These encompass both uncoupled, coupled, and two stage *synchronization opponent* systems. In some, a fixed initial condition is explored over a range of the {b (bifurcation), c (coupling)} parameter space. In others, an image generated by a parameterized function is given as an initial condition to the lattice. The image is systematically varied while one dynamical parameter is changed. Animations of the resulting distributions of states over the parameter and input space help to understand whether such dynamical behavior can be used in classification systems, and what limitations to expect.

In most cases these simulations were perfomed after the curve ordering experiments discussed in the next section; the parameters used were chosen from the set found to produce interesting behavior (i.e. the mapping of a curve family to distributions supporting ordering the curves by occupancy statistics defining a point in a partition cell metric space). The number of iterations in each study is low (5-9 iterations), relative to both standard practice in studying CML evolution from random initial conditions, and relative to the 10-20 iteration counts for recurrent neural ensembles to be considered biologically plausible. The lower numbers are chosen to emphasize the possible behaviors in a single processing stage in a two stage system.

When the term *final state* is used below, it simply means an arbitrary sampling time in the transient stage of an ongoing dynamical evolution, and does *not* imply that any asymptotic state or attractor has been reached. This sampling time is one of the parameters evolved by evolutionary search for the other experiments in shape similarity and object recognition. The sampling time is, rather, implicit in the sum of the two stage iteration parameters {t1 ,t2} of the Soca network.

Convergence Time Distribution for a Single Logistic Map

For constructing engineering systems and modeling biological processes, it is essential to study the rapid evolution of networks of chaotic units (i.e. the behavior in the attractor basin *transients)*. The best known measure of convergence and divergence for a particular instantiation (bifurcation parameter) of the map, Lyapunov exponents, is defined for times approaching infinity (Ingraham 1991). However, for computing with transients, knowledge or this number may in itself be inadequate; to estimate lower or uppers bounds on the number of iterations expected to play a role in computations, we can only perform numerical studies on the short term convergence behavior for uncoupled or coupled maps. The following diagrams illustrate convergence times for two bifurcation parameter values in the period 2 limit cycle regime. For the purposes of constructing representations during the transients, it is crucial that the coupled systems should not converge too rapidly from random initial conditions; hence, it is encouraging that such a large range of absolute convergence times are possible in the ensemble.



a)

b)

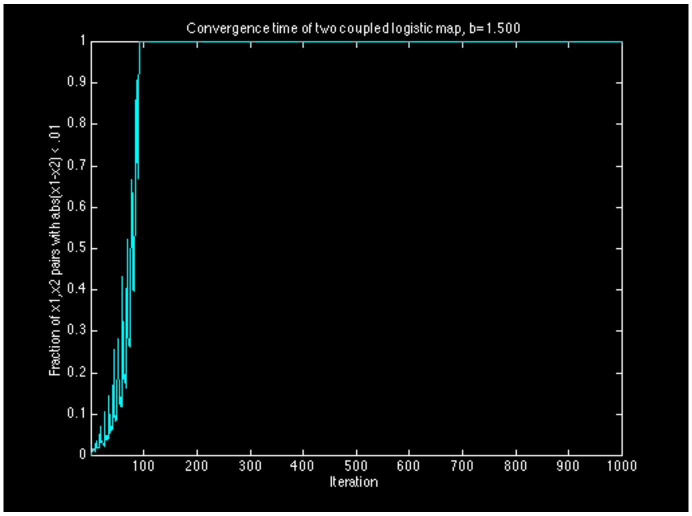
Fig. 26. Convergence times for an ensemble of 1000 initial state pairs.uniformly distributed from the interval [-1,1]. The number of iterations to reach x_t - x_{t-2} < .001 is plotted. a) average convergence time for b=1.2 b) average convergence time for b=1.06. the steep slope and lower absolute time to reach any given fractional threshold indicates a greater negative Lyapunov exponent.

Synchronization Time for Two Coupled Maps

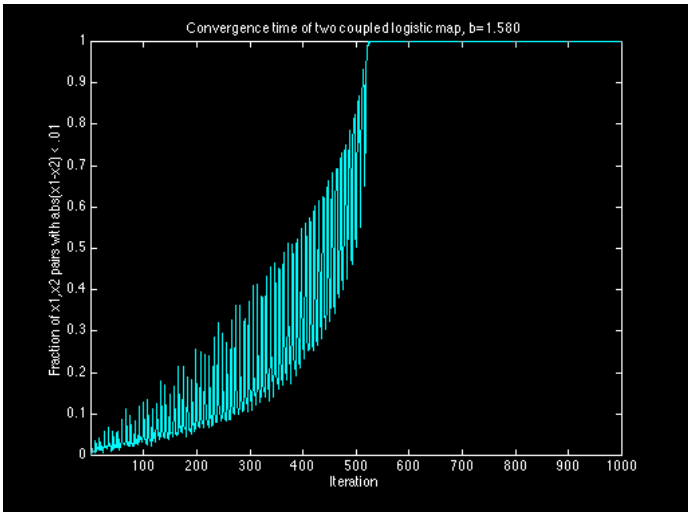
The previous section illustrated the time for a single map to reach stable values of period 2 oscillations, until the period 2 oscillations break down. In this experiment, we examine two coupled map units with the same logistic equation but symmetric coupling. Between iterations, a diffusive coupling step adds to each map’s state a fraction of the other map's state, scaled to avoid exceeding the stable domain of the map.

$$x1_{t+1} = 1 - b((1-c)(x1_t + c * x2_t)^2$$
$$x2_{t+1} = 1 - b((1-c)(x2_t + c * x1_t)^2$$

Here, the time for two coupled maps orbits to approach to one another to some distnce ε from an ensmble of random initial conditions (x1-x2 <.01) is examined for a coupling value (.38) which achieves synchronization until strong chaos is induced in each map. The bifurcation parameters are varied from sub-critical 1.5 to the maximum 2.0. Illustrated below are two values of b, first sub-critical (i.e. below the uncoupled transition) and the second beyond the uncoupled transition to chaos. While the maps may not be fully synchronized at this first approach, they certainly are *not* synchronized prior to this time.



a)



b)

Fig. 27. First approach time for a) b=1.5 , c=.38 b) b=1.58, c=.38. The transition to chaos for uncoupled maps is approximately 1.54.

System of Two Coupled Discrete Maps: Basic Behavior As Bifurcation And Coupling Parameters Are Varied

Time Evolution of Single Map in a System of Two Coupled Logistic Maps with Constant Parameters

Two maps are iterated from initial conditions .0001 and .9999. Each map iterates the logistic equation

$$x_{t+1} = 1 - bx_t{}^2$$

Between iterations, a diffusive coupling step adds to each map’s state a fraction of the other map's state, scaled to avoid exceeding the stable domain of the map.

$$x1_{t+1} = 1 - b((1-c)(x1_t + c * x2_t)^2$$

$$x2_{t+1} = 1 - b((1-c)(x2_t + c * x1_t)^2$$

This animated surface shows the time evolution of the .9999 map, with the axis indicating fixed time parameters for bifurcation and coupling. In this early transient portion of the map's orbit, the variation between adjacent points in the parameter space is seen to be relatively smooth. With increasing time, the surface will become irregular as the bifurcations implicit in the dynamics separate the orbits into attractor basins. Note that the procedure is carried out on a 200 x 200 matrix of states, with each matrix element value computed from corresponding matrices of *b* and *c* values.

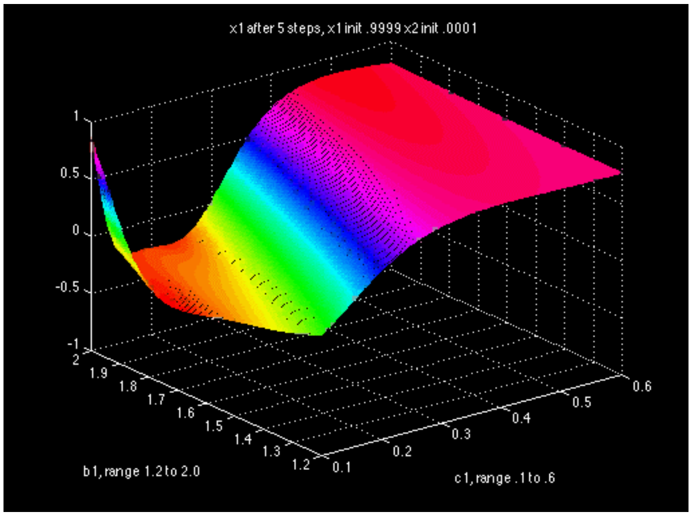


Fig. 28. A snapshot of the parametric variation in the time evolution of one coupled map cell after 5 steps, a typical single stage iteration count used in subsequent experiments for the formation of representation spaces. The smoothness of the surface suggests that network operation should be robust in the face of small variations in the parameters.

Final State in First Map of Two Map, Two Stage Constant Parameter System as Initial Conditions Vary

To investigate whether the particular initial conditions chosen to translate binary images into the logistic map phase space plays a large role in the outcome, we examine the state of one of two coupled maps over a range of bifurcation parameters when the initial conditions are varied to decrease the initial distance between the maps. The dynamics here are representative of the synchronization opponent style, with the second stage held constant (b2=1.3694, c2=0.19134, 4 iterations) while the first stage b and c are varied for 5 iterations. (The second stage values were drawn from parameters found to produce a distribution supporting metric distance functions matching a parametric curve).

We can see that the large separation in initial conditions plays a significant role in creating a complex surface in the transient state, supporting. adaptive behavior by delaying rapid synchronization.

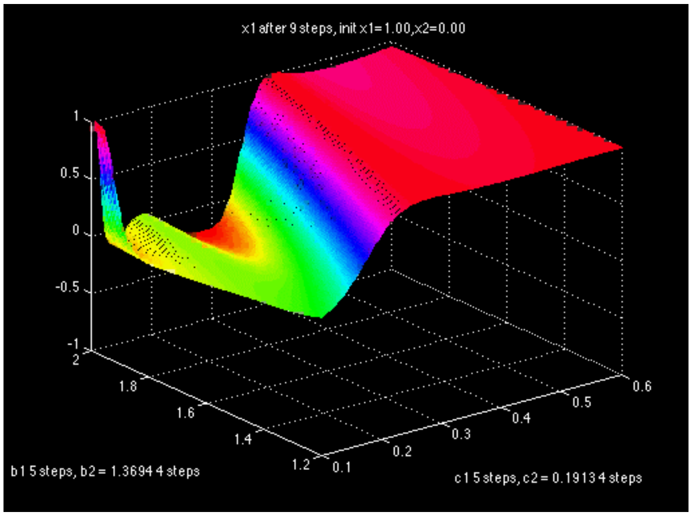


Fig. 29. Sampled orbit at 9 iterations with initial conditions 1 mapped to 1.0, 0 mapped to 0.0001. b2=1.3694, c2=0.19134 for 4 iterations, b1 and c1 as plotted for 5 iterations.

When the initial states are not well separated, synchronization occurs rapidly for most c values, thus only a small coupling parameter range results in any separation, thus the ability of the sampled distribution to construct state flows mapping curves or arbitrary shapes to a metric output distribution is reduced.

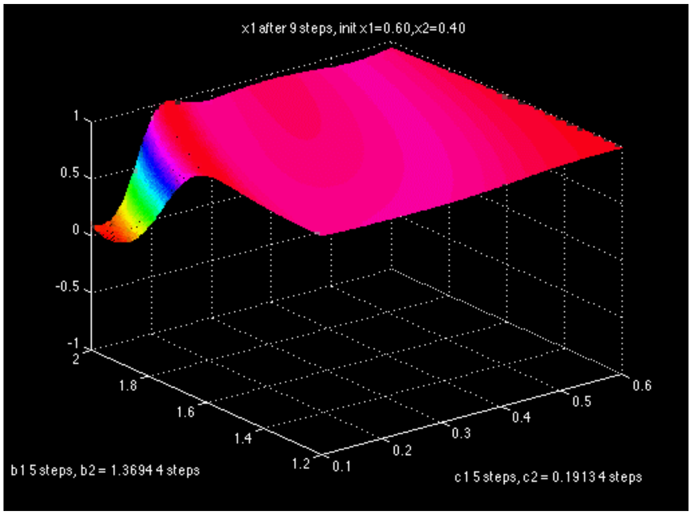


Fig. 30. Sampled orbit at 9 iterations with initial conditions 1 mapped to 0.6, 0 mapped to 0.4.

Effect Of Coupling And Initial Condition Separation On Final State In A Two Stage Parameter Cycle

High coupling values result in the eventual convergence or synchronization of maps, even when their individual bifurcation parameters would result in chaotic trajectories.

These two animations show the effect of relatively weak and strong coupling on two maps, for a two-stage dynamics. The surface shown is the difference between the final states after 5 steps of c1 and b1, 4 steps of b2 and c2.

In this case, the bifurcation parameter in the second stage is a fixed value (b2= 1.3694), while the first stage parameters range over the values shown. Each subsequent frame in the animation shows a different set of initial conditions, with the difference between x1 and x2 decreasing. The first frame corresponds to the values actually used in the subsequent experimental work. The b2 value chosen is one found in the evolutionary computation process for similarity ordering.

Difference Between Final States of Two Unit Synchronization Opponent System With Medium Strength Coupling (c2=.1913)

Note that the complexity of the surface shape in the low c, high b regime supports the ability of the lattice dynamics to preserve a linear relationship between the generating parameter producing an initial distribution of adjacent states and the final state. High values of c1 are seen to synchronize in 9 steps, even with low c2 coupling. Bifurcations values are as plotted with b1=b2.

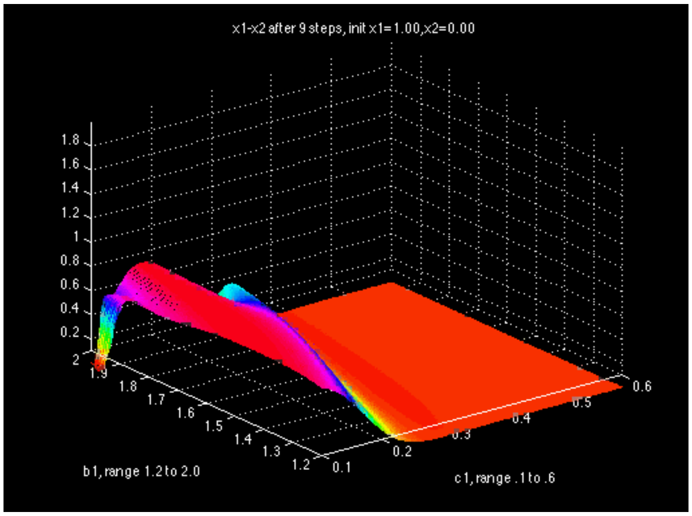


Fig. 31. Difference between two coupled logistic maps in synchronization opponent stage with b2= 1.3694, c2=.1913 and b1, c1 as plotted.

Difference Between Final States of Two Unit Synchronization Opponent Sytem With Stronger Coupling (c2 = .3913)

With stronger coupling in the second stage (steps 6-9) the final states are nearly synchronized for all values, whether the initial states are separated or not.
Obviously no distinctions can be made in this regime, after only 9 time steps.

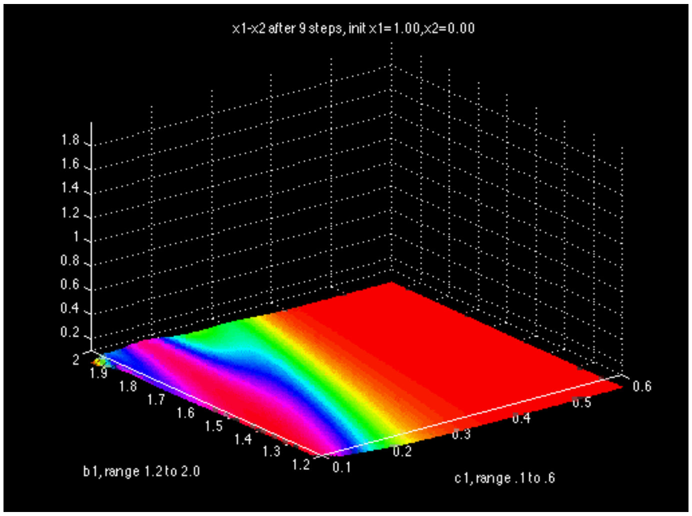


Fig. 32. Difference between two coupled logistic maps in synchronization opponent stage with b2= 1.3694, c2=.3913 and b1, c1 as plotted.

Sampled Transient Evolution in Lattices with Varying Parameters, Random and Structured Initial Conditions

Final State Distribution After 8 Iterations Varying b and c with Random Initial Conditions

A 200 x 200 matrix is initialized with random values uniformly drawn from the set {0.0001, 0.9999}. After 8 iterations for a particular pair of b and c values, the instantaneous distribution of states is measured with 256 partition cell bins. The procedure is repeated 20 times with new random matrices, and the average population at each {b, c} pair is recorded. Each frame of the animation shows a family of 256 bin histograms ranging across the bifurcation value b for a constant value of c. For each b value, the entire distribution is plotted (i.e. any bins with equal population would be overlaid by the last drawn bin). The first frame illustrates that with no coupling, the two initial states evolve independently to state values determined by the b parameter; with the different colors indicating the region of phase space visited in the 8th iterate.

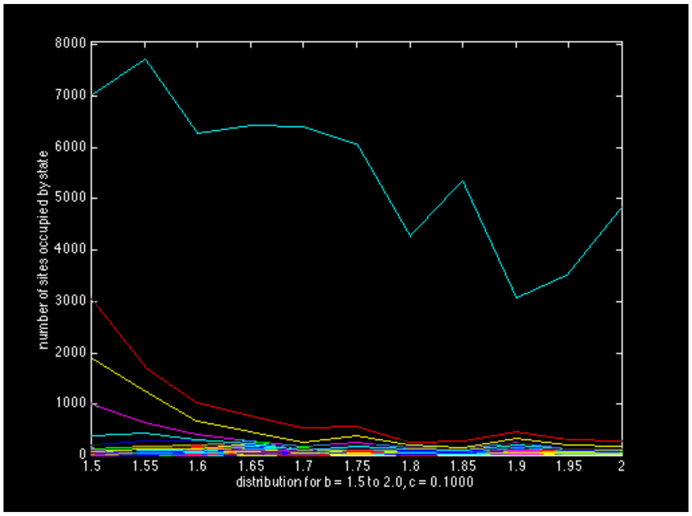


Fig. 33. Distribution of states after 8 iterations for spatially random matrix with site initial conditions chosen from the set {1.000, .0001}.

A few trends are evident. Low coupling with bifurcation values just above the transition to chaos of the uncoupled map (b = 1.54) produce distributions with several major components well separated. The smooth evolution of the modes as bifurcation is varied - for example, with c=.1, b ranging from 1.6 to 1.7 - is somewhat similar to that produced by the response of a system to variations in a spatial form.

This suggests that joint statistics of two interacting networks - one driven by high contrast spatial noise in such a parameter range, coupled to another with a response which linearly tracks the 2-D projection of a 3-D object - might serve to achieve a more nearly invariant response, while retaining metric properties as studied in the next section. Learning to recognize variants might consist of tuning the bifurcation parameters in the randomly driven subnetwork to offset the natural distribution changes associated with different views. This approach is not pursued further in the present work.

Final State Distributions with 45° Line Initial Condition Varying b and c Parameters

A 45 degree line is rendered, with the resulting image matrix mapped to values .0001 and .9999 in the lattice. The coupling between cells is increased between frames.

The x axis shows the bifurcation parameter b ranging from 1.5 to 2.0, while the y axis shows the population of each partition cell (histogram bin) for that value of b.

Each bin is plotted in the same color across the range of b values. Note that there are regions with multiple high population modes (i.e. c=0.1, b = 1.84 or 1.87), while others have a single low population mode with broad dispersion over the remaining bins.

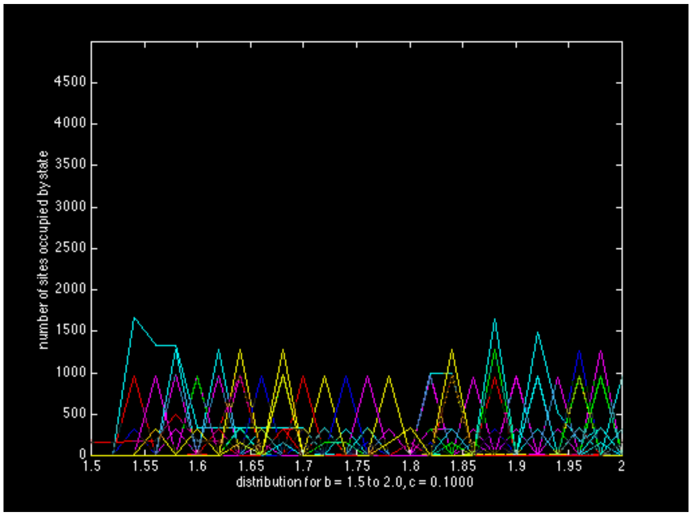


Fig. 34. The state distributions after 8 iterations of a single stage CML with a 45 degree line (line value 0.999 in a background of .0001) used as the initial condition. c=0.1

Final State Distributions with Fixed c varying b and Linear Initial Condition Rotated 0° to 45°

An interesting value of c (0.5 with some strong response modes) is selected from the previous experiment; holding that c value constant, each frame of the animation shows the distribution over the bifurcation parameter b as the slope of a plotted line (mapped to the .0001 and .9999 values) is varied.

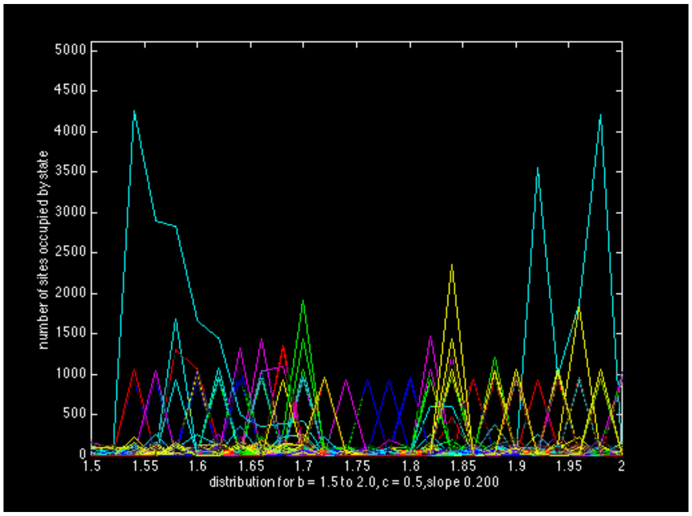


Fig. 35. Distributions after 8 iterations of a single dynamics stage with constant parameter c= 0.5, as the orientation of a diagonal line initial condition is varied from 0-45° between frames. The line shown in this animation frame has slope 0.2.

Discussion

The surfaces formed by sampled transients of logistic maps across the parameter control plane are rather smooth. This has both benefits and drawbacks in the engineering application of sampled transients. It suggests that smooth changes in a curve family or image can be matched by adjusting the CML control parameters to shift some distribution smoothly between bins. (This will be developed further in the next section). On the other hand, the simplicity and smoothness of the sampled surface might suggests that perhaps only simple shapes could be tracked by a network with spatially homogenous bifurcation and coupling parameters.

However, the smoothness in response of a single coupled site does not necessarily translate to simplicity in the network transformation of a form. The experiments with a 45 degree line and a line of varying orientation show abrupt changes in the distributions as dynamical parameters are varied.

EXPERIMENT GROUP 2: ORDERING OF PARAMETRIC CURVES

Data Selection and Methods

There is no universally accepted metric for similarity of shapes (or objects in general), though many have been explored as reviewed in the background material. For this reason, the next step in this study of spatial computations via partial synchronization focused on a restricted class of shapes where a metric is well defined: that of parametric curves. For a set of curves generated from an equation with one free parameter, the distance in the perceptual space of the resulting forms is assumed to vary linearly with distance between the generating parameter. This implies an *ordering* of the images generated for each curve. For each curve, I generated a set of 6 exemplars over a range of the parameters, captured a bitmap of the plot, performed binary to floating point conversion and scaled the values of the resulting array to fit the domain of the map.

The fitness function employed to evolve the parameter sets was simply to match the *order* of the distance between curves to the order expected from the curve's generating parameter. I employ a simple distance metric based on the post-evolution state statistics for the entire shape, with no information about local adjacency statistics (co-occurrence matrices) used in the evaluation of distance between the curves.

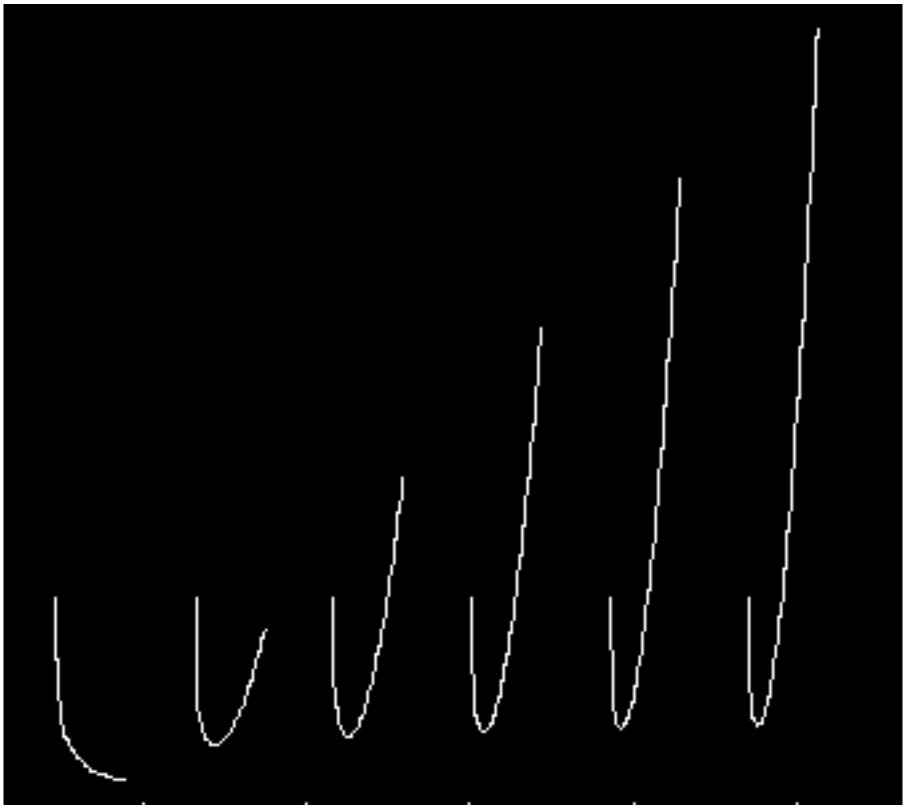


Fig. 36. Exemplar images for one of the parametric curves used in this set of experiments. The images are plots of the equation $y = c * x^2 - \log x$, with parameter c = 0, 1.5, … 7.5.

The evolutionary programming algorithm, described in the last section, searches for network parameters effective in ordering the parametric shapes according to their generating parameters. A learning trial consisted of 50 generations, with a phenotype pool of 50 individuals per generation, or 2500 individuals. Perfect orderings were found for all curves with this simple termination strategy.

The fitness function employed to evolve the parameter sets was simply to match the expected ordering shown below. Each curve was represented by six variants, thus the expected ordering of distance was represented by the matrix:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 4 & 5 & 6 & 5 \\ 3 & 3 & 2 & 3 & 4 & 4 \\ 4 & 4 & 1 & 6 & 3 & 3 \\ 5 & 5 & 5 & 2 & 2 & 2 \\ 6 & 6 & 6 & 1 & 1 & 1 \end{bmatrix}$$

Pairwise distances from each curve were computed with a weighted Euclidean metric on the space defined by k=256 partition cells. The resulting distance array was sorted, and distance replaced by the corresponding exemplar number, resulting in a matrix suitable for element by element comparison with the order matrix above. A perfect score of zero indicated that the distances obtained between the distribution vectors were identical to the matrix above.

Given that the distances between adjacent neighbors were theoretically equal, mismatch scores were computed according to row sums (1, 2, 3+4, 4+5, 6), with respective weights (10, 3,2, 3). The first and last rows were weighted more highly, given that values in rows 3,4, and 4,5 could be swapped. The high weight on the first row ruled out degenerate highly synchronized solutions, with a single partition cell occupied. This would essentially create a representation space based on size alone, which the genetic algorithm exploited prior to this adjustment.

Mutation probabilities for each network parameter and constraints on ranges are given in the following table. While the mutation probabilities would be high for some reproduction strategies, the simplex method always preserves the best individual of the previous generation and higher mutation rates serve to balance this conservative tendency.

Table 4. Mutation probabilities and ranges for each network parameter.

parameter	mutation probability	Range
b1	.40	1.2-2.0
c1	.25	0.1-0.7
t1	.30	2-6
b2	.40	1.2-2.0
c2	.25	0.1-0.7
t2	. 30	2-6

Experimental Results

Summarizing, I set out to examine the statistics of families of curves after a dynamical evolution in coupled map lattices. I observing that *two independent stages of CML parameters* in the dynamical transients produced distributions which served as a space with metric properties, with the partition cells of the dynamics serving as dimensions of the space. The study set out to answer the questions:

1. What trends are seen in the parameterization of the time varying coupled maps for assessing similarity, given bounds on the number of iterations? I know of no analytical procedure for computing the parameters, so any such optima would be discovered via evolutionary programming. If there were a single optimal value, perhaps the transition to chaos, the network should choose the same parameters for both stages.
2. Is there a single optimum parameter set for all such curve families, or would different curve families result in unique optimum parameters?

The answer to the first question is that the general principle of phase space expansion and contraction proposed in an earlier study appears valid; such solutions were consistently found through adaptive search. For one curve (astroid), a solution with all chaotic-regime local dynamics and high coupling was found. Network dynamics do not cluster around the transition to chaos for uncoupled maps, though this value should be shifted by coupling. This adds support to the earlier refutation by counterexample (Mitchell, Hraber et al. 1993) of optimistic claims for the universal utility of transition–to-chaos dynamics.

The answer to the second and perhaps more interesting question was negative. Unique parameter sets were found for different curves. An explicit attempt at evolutionary search for *a single parameter set* which would produce the proper ordering *for all three curves* was not successful, using the same evolutionary parameters and synchronization opponent stages which successfully ordered each individual curve family.

Table 5. Evolved parameter sets for four curve families

curve	b1	c1	t1	b2	c2	t2
ellipse	1.8671	0.1101	6	1.6392	0.3623	8
ellipse 5 cycles	1.7834	0.5515	5	1.9654	0.3604	2
astroid	1.7084	0.4546	3	0.4937	0.1974	5
pursuit	1.6739	0.2899	2	0.7567	0.3416	5

The table above lists the evolved parameter sets which served to correctly order each family, indicating that the partition cells form a quasi-metric space. Six exemplars were used to train the network. The critical point (i.e. the transition to chaos) for an uncoupled map is b=1.542.

It is reasonable to ask whether any common features characterize the selected parameters and their distributions. I measured the distributions of the resulting images after the CML transformation for parameters with winning and losing fitness, and observed a trend towad higher co-occurence entropies in the selected distributions.

To verify whether the parameter sets generated smoothly evolving distance or perhaps only random points which happened to match the exemplars, I generated a larger family for the curves and plotted the distributions. Typical distributions and co-occurence entropies for some curves are shown in the following plots.

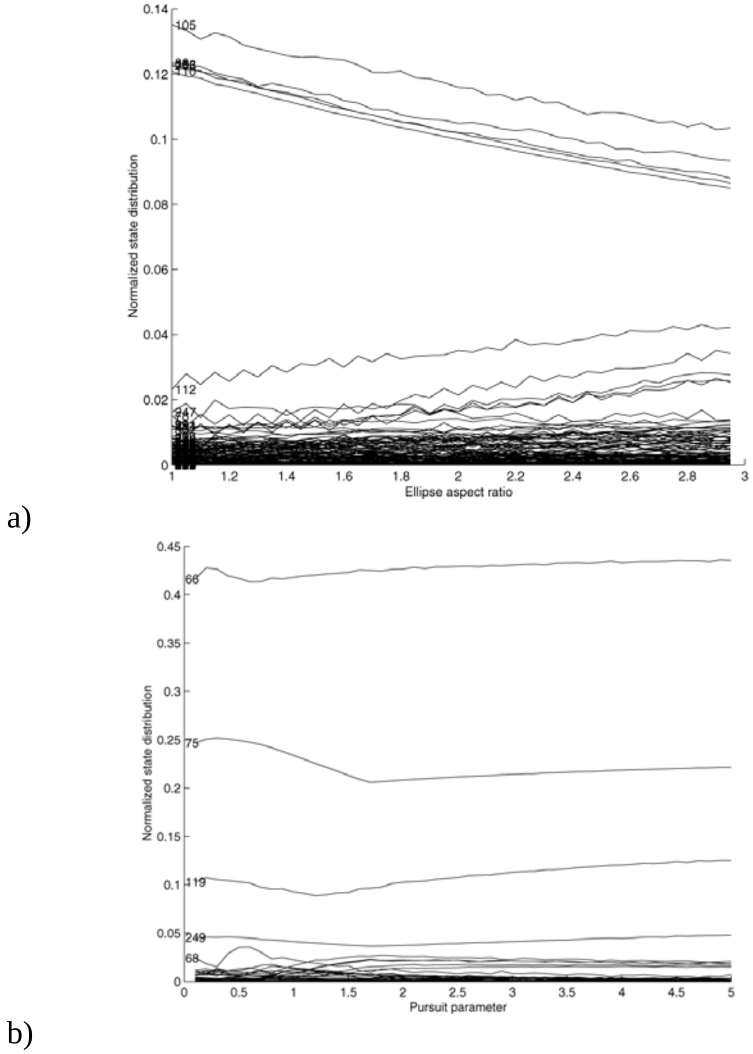
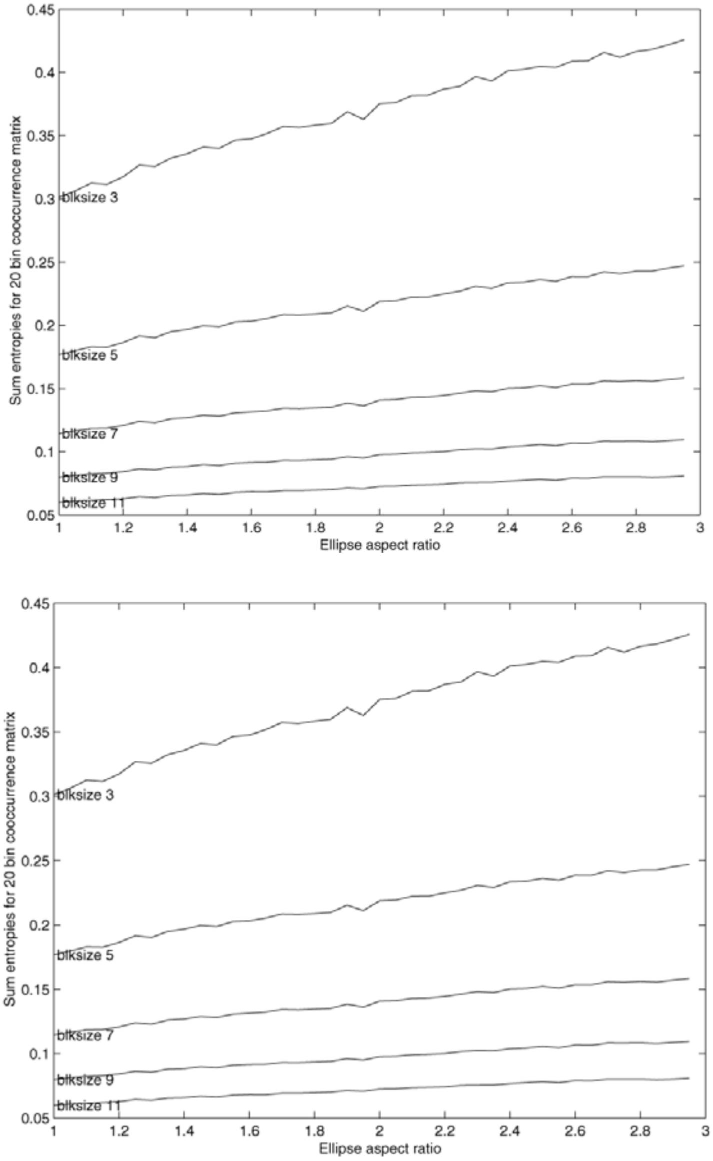


Fig. 37. Normalized state distributions from successful parameter sets on rank order task are plotted against the free parameter for a) the ellipse b) pursuit curve. Each plotted line is the fraction of states in a particular bin (partition cell); numbers are bin labels 1-256. Note that different bins are found for each curve, corresponding to different clusterings in the representation space. As the shapes evolve, the distribution changes might be described as a mode transfer.



a)

b)

Fig. 38. Sum of entropies plots. a) Sum of entropies of 20-bin co-occurrence matrix for increasing block size on ellipse with winning parameters; the entropy computation is as in (Del Bimbo 1999), but scaled by 1/(pixels/block). b) Sum of entropies for the *worst* score on the ellipse ordering problem in the first generation.

As noted earlier, the two processing stages could be viewed as a prototype of a cyclic change in bifurcation parameters based on slow wave rhythms seen in biological systems. Accordingly, I also examined whether an extended cycle (five “outer loops” of two stages) would still allow learning of parameters producing an ordering state distribution for the ellipse shape family. It was able to do so, but learning produced a different parameter set for this scenario. However, it is unclear whether cyclic changes are required or more realistic; readout could occur after the first (slow) cycle, and biological performance constraints indicate that this must be true if recurrence is used in the computation.

The *sensitivity* of the solution to changes in the bifurcation and coupling parameters was investigated for the ellipse curve. The fitness function was evaluated after perturbing each of the base parameters by the increments shown, while holding the others constant. The b1 parameter was very sensitive, with a change of +/- .0001 degrading the solution so that exemplars were incorrectly ordered when using the distributions as a distance vector. Changes in b2, c1, and c2 of +/ .0005 produced little degradation, preserving the order. The b1 parameter, typically operating in the chaotic regime, is presumably responsible for most of the entropy production, while the b2 parameter and high coupling accounts for the smoothness.

Table 6. Sensitivity analysis for the ellipse ordering.

parameter	delta	fitness (+ / -)
b1	+/- .00005	0 / 0
	+/- .0001	30 / 30
	+/- .0005	15 / 50
	+/- .0010	43 / 61
c1	+/- .00005	0 / 0
	+/- .0001	0 / 0
	+/- .0005	0 / 0
	+/- .0010	5 / 0
	+/- .0100	54 / 15
t1	+/- 1	55 / 68
b2	+/- .00005	0 / 0
	+/- .0001	0 / 0
	+/-.0005	15 / 0
	+/- .0010	5 / 0
	+/- .0100	30 / 54
c2	+/- .00005	0 / 0
	+/- .0001	0 / 0
	+/- .0005	0 / 0
	+/- .0010	0 / 40
	+/- .0100	20 / 54
t2	+/- 1	64 / 42

Discussion

Network parameters are found produce high entropy but smoothly evolving state distributions which serve to order curves in a metric space defined over partition cells of the network phase space. I interpret the *corresponding parameters* as the memory trace for the category corresponding to each curve, and the distribution produced as an intermediate computational state to be subject to some comparision to memory. The transformation of arbitrarily detailed images to partition cell bin statistics constitutes feature selection and dimension reduction. I have demonstrated the basic ability of the chaotic-periodic transient scenario to produce a state distribution with metric properties, allowing it to be used as a similarity function for curves.

I have developed procedures to compute multi-scale co-occurrence matrices, and examined the entropy statistics at different scales. State distributions found to be effective in rank ordering the members of curve families had higher entropies than the failed parameter sets.

Entropy measures are commonly used in statistical pattern recognition as a measure of the effectiveness (in terms of error minimization) of feature subsets (Chen 1973). This may be related to the discovery of high entropy solutions are found in this representation space, which also performs a dimension reduction. This insight was useful in the formation of the objective function used in the next section, where the maximization of a Shannon entropy measure is one of several components used to evaluate the representation for multiple views of an object. However, the maximization of co-occurrence entropies, as measured in this section, was deemed too costly a computation to embed in evolutionary search.

EXPERIMENT GROUP 3: RECOGNITION OF PAPERCLIP OBJECTS ROTATED IN DEPTH

The following set of experiments examines the abiliity of the network to solve the stimulus identity problem, in which different views of an object must be recognized as "similar" to form a category, even though they may differ rather radically. It would not be clear that the ability of the network to make a representation for smoothly evolving objects demonstrated in the last section could be extended to transformations of very different outlines or curves. With an appropriately chosen fitness reasonable performance is obtained on this task, even when a subset of views is shown during training. In addition to recognizing the identity of an object, I will require that this be done against a background of similar distractors.

Selection and Preprocessing of Data

The images used in this set of experiments were designed in the visual psychology lab of M. Tarr. The image set consists of 39[29] *paper clip* objects, with seven views provided for each object rotated in depth. Each object is a chain of 5 cylinders, with a variable joint angle connecting each pair. The views are separated by 30°, ranging from –90° to 90°. The objects were originally developed to answer questions regarding the *recognition by components* or *geon* theory of Biederman (Biedermann 1987). The set consists of four *complexity groups* with 0, 1, 3 or 5 unique geons substituted at some position in the chain.

Similar paperclip objects (corresponding to the low complexity set) have been used in human psychophysics experiments (Bulthoff and Edelman 1992). In these experiments, subjects were trained with motion sequences of 2-D views, giving an impression of a 3-D object through kinetic depth effects. In a two-alternative forced choice task on their object set, with single static views of a target or distractor, the miss rate (failure to indicate a match when the target was shown) averaged 30%, indicating that the task is rather difficult.

---

[29] One object in the last group was duplicated in the original set, hence 39 rather than an even 40.

The specific object set used here was also used in a study by Tarr and colleagues attempting to discriminate between view-based and structural theories (Tarr, Bulthoff et al. 1997). In this study no training period was provided; subjects simply had to judge whether two views shown briefly (200 and 100 ms, separated by a mask stimulus) were the same or different. Under these conditions, the baseline set of shapes (all tubes with no geons inserted) were essentially not recognizable by subjects when presented in other than the training views.

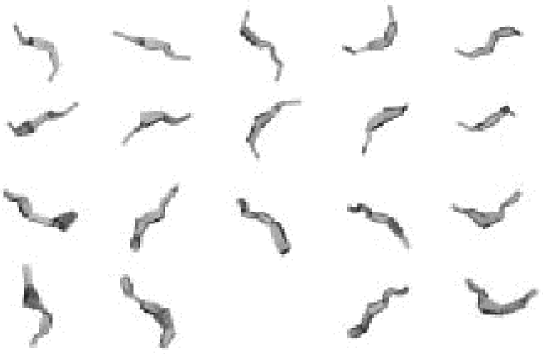


Fig. 39. Single views of a subset of the paperclip + geon objects used for recognition. The base set (not shown) are 5 tubes connected at arbitrary angles. The top two rows are 10 unique objects with 3 geons substituted for arbitrary tubes, while the bottom 2 rows have 5 geons (no tubes). The array of Soca classifiers is required to identify which object is presented, given a single view ranging from +90 to –90 from the 0 orientation shown. In a less demanding task, the network must judge whether two successive images are the same or different objects. From Tarr, M. J., Bulthoff, H. H., Zabinski, M. and Blanz, V. (1997). “To what extent do unique parts influence recognition across changes in viewpoint?” Psychological Science 8(4): 282-289. Reproduced with permission of Blackwell Publishers.

The image set was pre-processed for learning and recognition by the Soca system as follows. The images were provided as 8 bit gray scale, at 300 x 300 pixels. Each image was subsampled two times (i.e. to 75 x 75, thresholded to a binary image,then transformed into floating point as required for the CML iteration, with 0 mapped to .0001 and 1 mapped to .9999. Finally, additional padding sufficient to account for the maximum allowed iterations was added to the image border to prevent boundary effects from the Matlab CML implementation[30] from intruding on the sampled distribution of the object itself.

---

[30] A convolution routine with toroidal boundary conditions would eliminate this step.

Experiments were performed in a staged fashion. Initially, performance was assessed on a small set of objects to experiment with weights in different terms of the objective function described below.

Given a baseline learning algorithm, an experiment to assess the maximum recogntiion performance on a small set (20 objects) was performed by training with all views. The effects of several variations in the learning process were studied with this set. The parameters varied include both weights on terms of the objective function described below, and changes in the evolutionary programming parameters.

A subsequent set of experiments assessed generalization or *view interpolation* performance: what effect does providing fewer training views have on recognition performance? Finally, the same experiments were applied to the full set to get some indication of recognition performance scaling with an increasing number of objects in the visual world.

Learning: Balancing Normalization and Separation

Any effective representation must negotiate the classic clustering vs. separation dillema. In the present case, class members to be clustered are different views of the same object. To associate various views to the same object, their presentation is clustered in time in learning epochs. A representation is formed chiefly on a measure of the CML computation on a single object's views, but with some influence from the representations of objects already learned. The relative influence of these factors is controlled by empirically determined weights in an objective function.

The view normalization concept, in which a classifier network strives to transform all the presented views to a common output is here modified to apply to recurrent networks. (Since the Soca network is a single layer recurrent system, the output is identical with the state variable). Unlike the Chorus feed-forward implementation of view normalization, no particular view is chosen as the canonical one. Instead, the sum of Euclidean distances across *all* view pairs *i, ,j* is taken as the objective function to be minimized:

$$D = \sum_{ij} \sqrt{\left(\sum_{p=1}^{k} v_{i,p} - v_{j,p} \right)^{2}}$$

where v_p are the occupancies of k partition cells for each of the j views.

It is easy for the network to discover *highly synchronized* dynamics which map all views to the same representation; in fact to search more efficiently such solutions are detected, and result in termination of the fitness evaluation prior to the more costly procedure described next. The minimum bifurcation parameters were increased in an attempt to search more efficiently.

To separate categories in pattern recognition applications, it is necessary to tightly cluster the category members (different views of the same object) while separating the means of each category (Duda and Hart 1973). To accomplish this task *without* *explicitly performing the search task as part of the fitness evaluation*, a **cross-entropy** measure was maximized as part of the objective function.

Two different strategies for computing cross entropy were examined. In the first, a reference view (for the genotype under evaluation) was chosen arbitrarily for these computations and used throughout the experiments described here; during the generalization experiments when only two views were used, the previous reference –30 was changed to the –60 view.

After noting that the recognition rate for this entropy reference view often exceeded the overall rate, the strategy was changed to use a mean distribution over all views for both the genotype under evaluation and the stored signature distributions. This appeared to have little impact, with slight gains for small object world but a loss for the full set. More experiments would be required to address this question to a level of statistical significance.

Before describing the objective function in detail, it may be useful to have a look at the desired end result. The following figures illustrate the view normalization process and the effect of the cross entropy term in forcing diverse distributions.

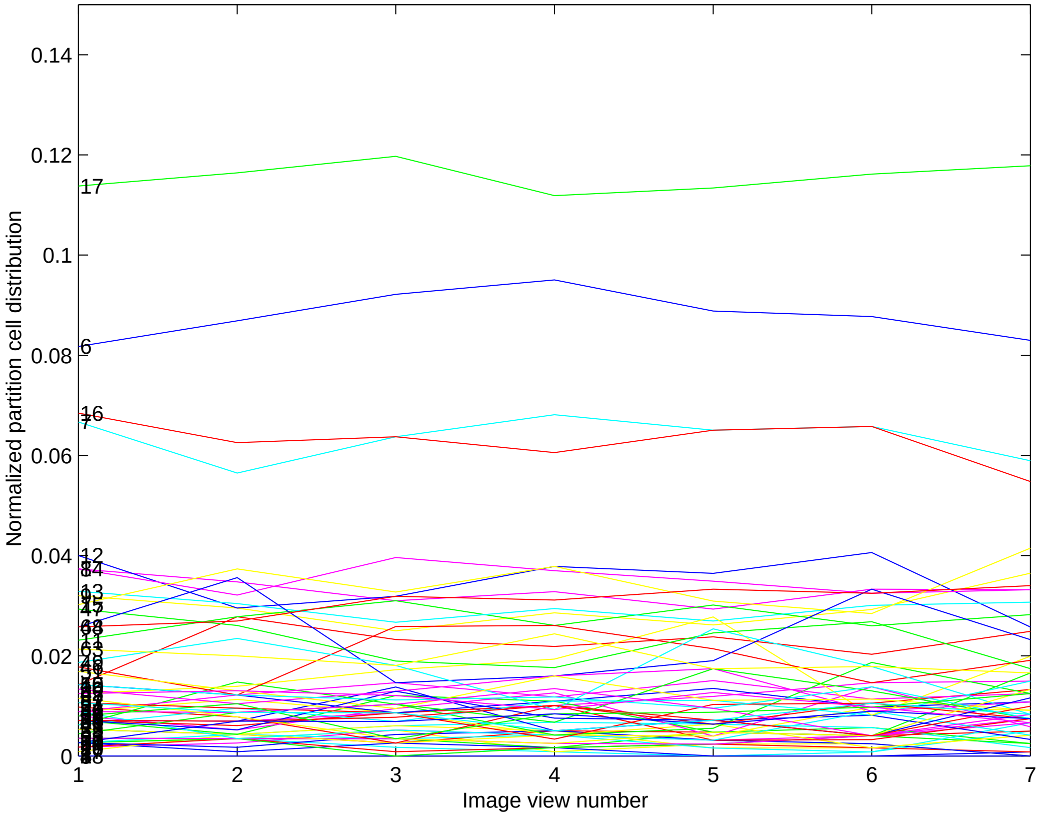


Fig. 40. Distribution of states generated by classifier across six views of an object 5.5 in the Tarr paperclip+ object set. The relatively small variance of each bin's occupancy illustrates the success of the normalization process. The labels on each line are the bin number. The mean value of each bin across all views are used as the signature for comparison during search. This distribution was generated with standard parameters W_d = 20, W_e =2, SyncThresh = .15.

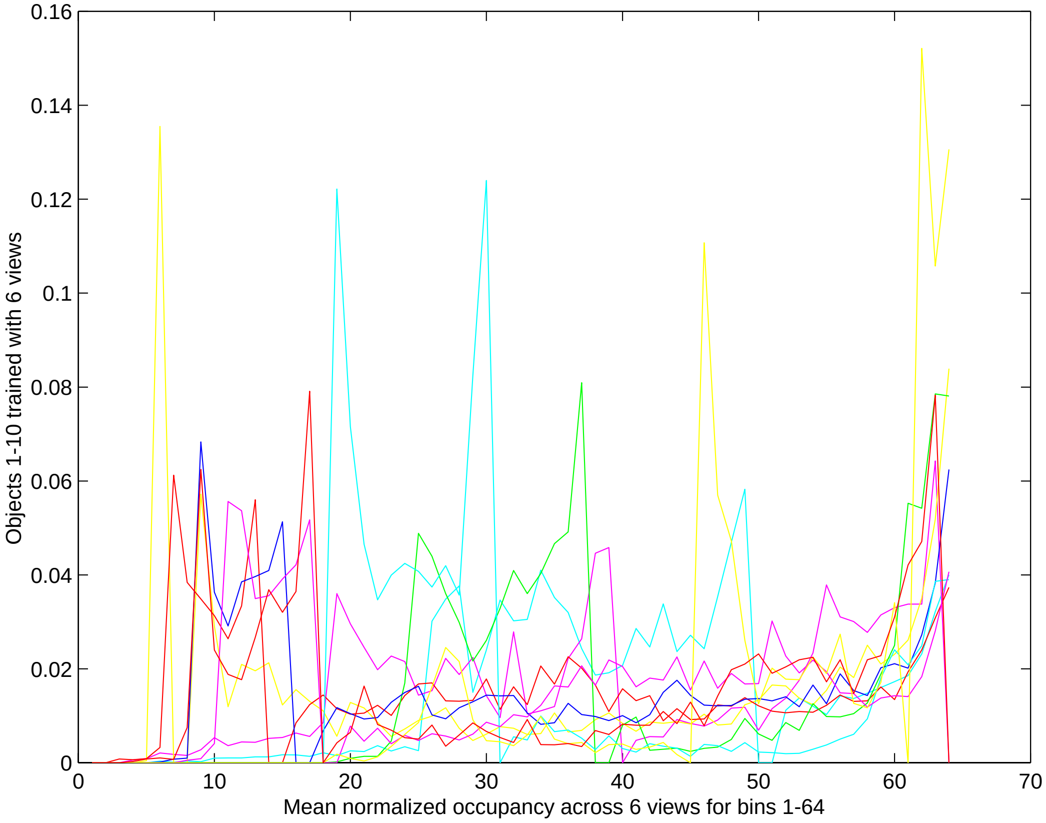


Fig. 41. The phase space dispersion effect of the cross entropy term. Each colored line is the normalized distribution across k=64 partition cells for a single object. The 10 objects comprise the low complexity (tube geons only) group, trained with the standard fitness function weights. Concentration of the population in the upper bins is evident, suggesting that many units may be approaching synchronization or a limiting distribution even in the low number of iterations allowed.

Many errors in earlier recognition experiments resulted from objects whose representation entailed relatively high synchronization, indicated by concentration of the population in one or two bins. Other objects would then tend to map closer to this distribution than to the mean mapping of the “correct” classifiers. It is easy to see that such a distribution is effective on the normalization task – it indicates that the system is approaching a limiting distribution under high coupling, a distribution which may even be uniform *for any input*. This was recognized early in the exploration of the system; to prevent this behavior, a *Shannon entropy* term rewarding broad distributions was added to the objective function.

However, adjusting the relative weights of entropy and normalization terms seemed unable to compensate for the problem. To correct for this tendency, a third term was added which penalized solutions where the maximum bin occupancy exceeded some threshold. After a few trials, summarized in the table below, a value of .15 was chosen. Any distribution whose largest bin population exceeded this threshold had a large penalty (1000) added, ensuring its rejection.

The Objective Function for Balancing Clustering and Separation of Classes

The objective function *f*, with low values indicating higher fitness, takes the form

$$f = W_d D - W_e(H_c + H_s) + P_{synch}$$

where W_d is the inter-view distance or normalization weight and W_e is the entropy weight; *D*, the inter-view distance sum for *j* views is

$$D = \sum_{ij} \sqrt{\left(\sum_{p=1}^{k} v_{i,p} - v_{j,p} \right)}^{2}$$

where v_p are the occupancies of *k* partition cells for each of the *j* views.

H_c is the cross entropy or Kullback-Lieber information measure between the current reference view distribution *C* and the database of N object distributions (signatures) S with k bins:

$$H_c = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{k} C_j \log_2 \frac{C_j}{S_{i,j}}$$

H_s , the Shannon entropy of the current signature with *k* bins is

$$H_s = \sum_{i=1}^{k} S_i \log_2 S_i$$

and P_{synch} = 1000 if max(S_I) > synchPenThresh, 0 otherwise. The parameter synchPenThresh was empirically determined as .15.

The following table indicates results of varying the synchronization penalty threshold on the nearest neighbor match of the 0° view.

Table 7. Error rates for alternative weights on objective function terms

W_d	We	SynchPenThresh	Error Rate %
20	2	.4	45
20	2	,25	24
20	**2**	**.15**	**15**
min	Max	.15	40

The table above is a comparison of effects of different sets of objective function parameter weight on performance of nearest neighbor match with 20 objects and training with all views. The last row indicates that rather than two terms with weights, the objective was formulated as a minimizing the *ratio* distance / entropy. The set W_d=20,W_e=2, SynchPenThresh =.15 was selected as the *standard parameter set* for training in the subsequent set of experiments.

In another investigation of learning parameters, two different mutation styles were evaluated. All results reported here apply a new random value uniformly distributed within the bounds presented earlier. However, standard practice for mutation of real valued parameters is to update values by drawing from a Gaussian distribution centered on the current value (Wright 1991). Using Gaussian mutation (variance = .02) on the CML bifurcation and coupling parameters resulted in an error rate of 29%, significantly worse than the 15% best rate obtained with uniformly distributed mutation. Presumably this is due to the highly nonlinear effect of the parameter values on network performance; however without a more extensive set of trials, possibly exploring a broader range of variances no firm conclusion can be drawn on the relative merits of the two mutation strategies.

Mutation parameters used during learning trials for this task are shown in the following table. Note that mutation is applied only to copies of the best individual from the last generation, which comprises 50% of each new generation, so the rates are effectively half of the quoted number.

Table 8. Mutation rates used for each parameter

parameter	mutation probability	Range
b1	.50	1.4-2.0
c1	.35	0.1-0.7
t1	.30	2-6
b2	.50	1.4-2.0
c2	.35	0.1-0.7
t2	. 30	2-10

Nearest Neighbor Match with Training on All Views

Once a reasonable baseline objective function was arrived at, some variations in learning parameters were examined on a set of 20 objects, consisting of 5 from each complexity group. In this set, all views of the object were used to form the representation. The representation consists of *both* the winning parameter set (i.e. the synchronization opponent genotype) for the objective function described above and the resulting mean distribution over 64 partition cells in the interval <-1,1>. The mean is taken for each cell from the 7 separate distributions computed for each view. The full set of representations, consisting of two arrays *genotype* and *binMeans*, constitute the object recognition database for a particular object world.

In a recognition test suite, each {object, view} pair is used as a target. A set of "hypothesis" distributions is computed by applying all classifiers (synchronization-opponent genotypes) to the target. The Euclidean distance of each hypothesis distribution to each corresponding database mean is computed; the object classifier whose genotype produced the minimum of all such hypothesis distances is selected as the match. Since all classifiers evaluate, performance of the best match process for N objects and corresponding classifiers is of **O**(N) on a serial machine.

Training Order

There was some concern that the best recognition performance achieved in the parameter tuning scenarios could be an artifact of training order, as all members of each 5 member group were trained in sequence, with groups listed in increasing order of complexity. An experiment was performed after scrambling the order of training; this produced an error rate of 18% vs. 15%, suggesting that training order within objects does not have a dramatic influence on the outcome. Once again, a large set of experiments would be needed to draw significant conclusions.

Single Dynamical Stage vs. Synchronization Opponent Stages

A set of learning trials was conducted with iteration time for the second stage fixed at zero, limiting the computation to a *single* dynamical stage (in contrast to the synchronization opponent system with two stages). To allow a fair comparison the single stage was allowed an equivalent maximum number of iteration as the two stage network (16). All seven views were provided for training, on the set of 20 objects. It turns out that the single stage dynamics performs surprisingly well, but with a limited number of trials (10 each for single and two stages) the average recognition rate and the average recognition time over all the classifiers were better for the two stage system. The following table summarizes this preliminary result:

Table 9. Single Stage vs. Two Stage (Soca) Dynamics (Average of 10 trials)

Trial type	Average Recognition Rate	Average Iterations / Classifier
single stage	74.7	15.3
two stages	77.9	12.7

This preliminary result is interesting in at least two ways. Based on the original intuition underlying the network, I expected worse performance from the single stage trials. The fact that the trend in performance is that the difference in scores is not so significant suggests that perhaps the simple probabilistic finite state automata recognizer explanation is adequate and the best possible explanation, without the additional gloss of the subspace synchronization argument.

The difference in average iterations per classifier is interesting, particularly since this is *not* part of the objective function. It is suggestive of a biological evolutionary scenario in which significant incremental peformance and reaction time benefits accrue from the use of non-stationary parameters.

Further study with a large number of trials is required to settle these questions. Ideally, this should be performed after experiments to assess the most effective mutation rates. Early work in progress on mutation rates suggests that the mutation rates used in the thesis produce a larger variance and overall lower mean for the two stage trial.

Random Algorithm vs. Evolutionary Learning

A learning trial was performed with random replacement of all but the best of generation during the reproduction stage, essentially a random algorithm with relaxation to a local minimum. This training procedure resulted an error rate of 29%. This validates the effectiveness of the genetic algorithm with simplex reproduction procedure, which consistently produced results in the range of 15-20% error rates.

Recognition is Not Scale Invariant

A learning trial was performed with the objects sub-sampled to 50% rather than 25% during the search process. The network parameters learned at 25% scale applied to this larger scale object resulted in an error rate of 95%, essentially chance performance.

Recognition of Untrained Views

Having settled on a canonical set of learning parameters, a series of learning trials were run with the number of training views ranging from two to seven, to assess the networks ability to generalize to unseen views. Generalization performance for worlds consisting of 20 and 39 objects, is shown for a number of views ranging from 2-7. Training view subsets started with the pair {-90, -60}, adding additional views in order {-30, 0, 30, 60, 90}. The results of this exhaustive nearest-neighbor match test suite are reported in the tables below and plotted in the figure following the tables. The third column indicates the rate of matching the view designated as the *reference view* in cross-entropy calculations during training, which was expected to be higher. This is indicated as n/a for mean-mean entropy trials.

Table 10. Nearest neighbor match recognition rates (20 objects, ref.-mean cross entropy)

training views, % correct for views from –90 to +90	mean % correct all views	% correct entropy ref.
2 c=[100 100 15 20 10 10 45]	42	100
3 c=[100 100 100 30 10 10 15]	52	100
4 c=[100 100 100 25 15 15 20]	53	100
5 c=[90 95 85 100 80 10 30]	70	85
6 c=[75 75 80 80 80 75 20]	69	80
7 c=[85 80 85 80 85 85 90]	84	85

Table 11. Nearest neighbor match recognition rates (20 objects, mean-mean cross entropy)

training views	mean % correct all views	% correct entropy ref.
2 c=[100 100 35 10 15 15 40]	45	n/a
3 c=[95 95 100 20 35 15 40]	57	n/a
4 c=[90 90 95 85 20 10 15]	58	n/a
5 c=[90 100 100 95 85 5 30]	72	n/a
6 c=[90 95 90 70 95 85 30]	79	n/a
7 c=[85 75 85 75 80 85 80]	81	n/a

Table 12. Nearest neighbor match recognition rates (39 objects, ref.-mean cross entropy).

training views	mean % correct all views	% correct entropy ref. (2)
2 c=[100 100 13 18 8 5 10]	36	100
3 c=[95 95 100 13 8 10 13]	48	95
4 c=[82 92 79 87 13 5 8]	52	92
5 c=[92 87 87 88 85 10 5]	64	87
6 c=[79 77 79 77 74 69 18]	60	77
7 c=[62 69 62 69 61 56 56]	62	69

Table 13. Nearest neighbor match recognition rates (39 objects, mean-mean cross entropy).

training views	mean % correct all views	% correct entropy ref. (2)
2 c=[100 100 15 7.7 7.7 0 13]	34	N/a
3 c=[92 92 92 13 5.1 13 13]	46	N/a
4 c=[90 97 95 90 10 13 5.1]	57	N/a
5 c=[79 82 82 84 64 13 2.6]	58	N/a
6 c=[65 67 72 72 59 56 31]	60	N/a
7 c=[62 60 59 59 69 56 51]	59	N/a

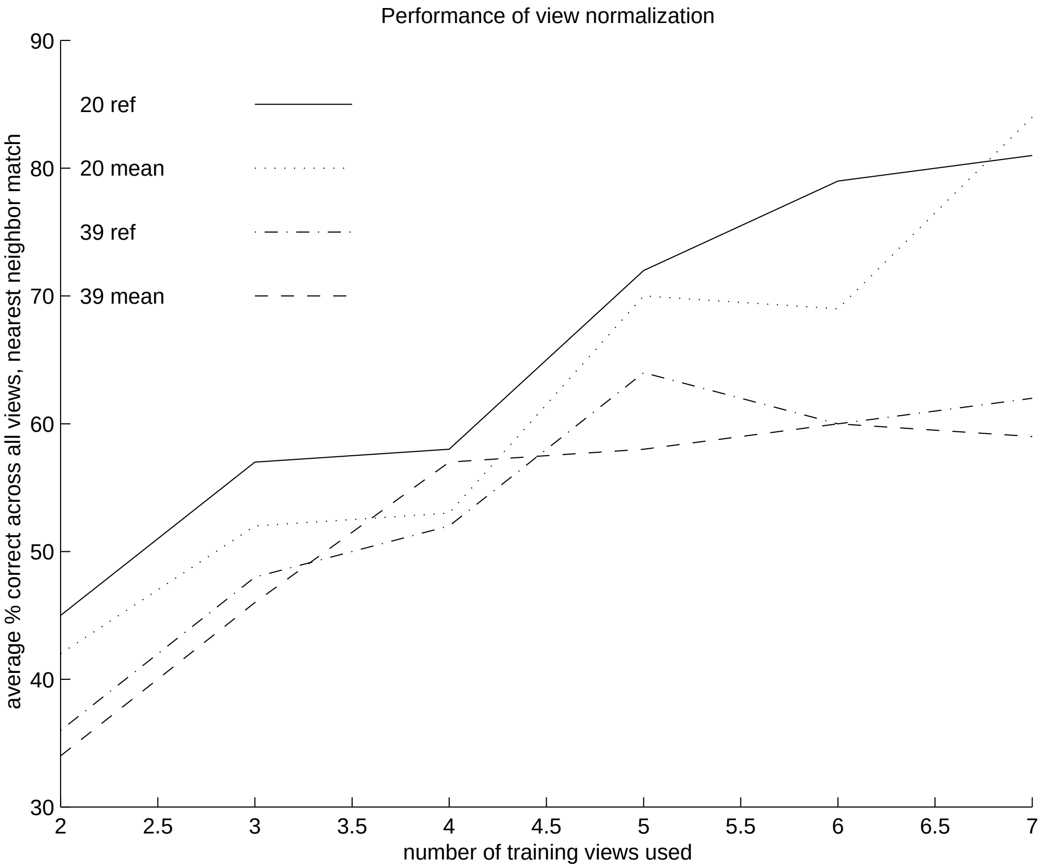


Fig. 42. Summary of nearest neighbor recognition rates with training on 2-7 views, for 20 vs. 39 object worlds, and for reference and mean cross-entropy computations .

For the 39 object world, performance improvement is minimal beyond 4 training views. The curves labeled *ref* computed the cross entropy term of all other object distributions relative to a particularly arbitrary view (-60) during training, while those labeled *mean* computed cross entropy relative to the *mean of all views the current object.* No clear advantage in seen for the mean vs. mean computation style, which I expected after noting that the view used as a reference was rarely misidentified. The mean-mean procedure improved performance slightly for the small set, but degraded it on the full set.

Category Generalization: Training with A Subset Of Objects

During the trials with all views provided for learning, I noted that erroneous matches were predominantly to objects within the same geon complexity class (i.e. tubes + 1, 3 or 5 geons), suggesting that some clustering was taking place in the representation space.

Two learning trials were performed using the standard learning parameters as above, but only using the 5 of the 10 available objects in each group (thus 20 training objects), then applying the nearest neighbor match to the remaining set of 20 objects never seen in training. For the two partitions (learning using first or second 5 of each set), 50% and 55% of the target objects returned matches from the same category. Chance expectation that an individual guess is in the correct category is .25; applying the binomial formula gives a 1% chance of achieving 50% correct assignment by random guesses. In accordance with the previous experiment, fewer errors were made for more complex objects; for the 5 geon group, all test objects were assigned to the correct class.

Given these results, the system appears to form clustering and subordinate level categorization (e.g. which “breed” of species paperclip+) on the basis of complexity. Whether humans would form such categories is less clear and is probably task dependent. Certainly alternatives which cut across complexity based categories are easy to envision, such as the categories ‘Large center with small ends’ and ‘Large ends but small center’. A more explicit examination of clustering is covered below.

Simulating Match / No Match Trials with Random Pairs

In this set of experiments, an alternative paradigm for assessing performance was used to support easier comparison with neuropsychology and psychophysics experiments. The *two-alternative forced choice paradigm* involves a human or animal subject undergoing a period of training on some set of stimuli, then undergoing a brief (sub-second) exposure to a stimulus, followed by a second stimulus. Some distractor mask signal may be used in between the two targets. The subject responds with a match/no-match choice as rapidly as possible. Error rates, in terms of false positives and negatives are computed; reaction times may also be recorded, and used as a measure of confusability between objects.

This paradigm more closely matches the testing strategies undertaken in psychological studies reviewed earlier. Since (to my knowledge) no experiments with the nearest neighbor match paradigm of the previous section have been performed, it is unclear how well monkeys or humans could perform an arbitrary nearest-neighbor match with a large number of paperclip objects, as in the previous sections simulation paradigm. Certainly for special cases of subordinate categories like faces, good performance is possible for a large number of objects.

In the simulated version of the forced choice test paradigm (with the Soca classifier system as subject), 500 trials of two views are drawn from the set of all objects and views. The nearest-neighbor function is computed as above for each view of a pair to determine whether the nearest object is judged as same or different for the pair. An error is counted for false positive or false negative matches, with the total error rate as shown in the table below.

This test scenario was performed for 20 (5 of each 4 groups) and 39 object worlds. In addition, a variant in which the *second object of each pair* is drawn only from *untrained views* is shown.

Table 14. Pair match error rates (20 objects, both drawn from trained or untrained views, 500 trials).

Training views	Error Rate %
7	2.6
6	3.8
5	4.0
4	7.6
3	9.2
2	9.2

Table 15. Pair match error rates (39 objects, both drawn from trained or untrained views, 500 trials).

Training views	Error Rate %
7	4.4
6	6.4
5	6.0
4	6.4
3	7.6
2	10.4

Table 16. Pair match error rates (20 objects, second drawn from untrained views, 500 trials).

Training views	Error Rate %
7	no untrained views
6	8.2
5	10.2
4	7.6
3	8.6
2	8.6

Table 17. Pair match error rates (39 objects, second drawn from untrained views, 500 trials).

Training views	Error Rate %
7	no untrained views
6	4.8
5	4.6
4	3.4
3	3.6
2	4.0

Discussion

The biggest surprise in the pair matching errors is the relatively low error rates in the last set (Table 13), with the first object drawn from trained views and the second object drawn from untrained views. This *increased* the errors when only 20 objects were used made performance nearly independent of the number of training views. Closer examination of the experimental conditions and combinatorics suggest that the explanation lies in the fact that for the alternative condition (i.e. first object drawn from trained or untrained views), the probability of choosing similar viewing angles of different object is higher; in the apparently anomalous condition, the same viewing angles are never compared. The number of “false positives” (judgements that different objects are the same) accounts for most of the higher error rate seen in table 17, when few objects but greater likeliood of similar viewing angles are given. Thus I believe the combination of more (and more complex) objects in the world and non-intersection of viewing angles explains the low error rates in table 18; views which map to the wrong object are simply rare under this condition.

Some degradation of performance on the larger test set was expected, but the amount seen here suggests that some action should be taken. Several modifications which might help in scaling up to a larger object "world" are apparent. Increasing the number of bins (the number of dimensions in the representation space) might help increase the separation between object categories. Increasing the weight of the entropy term relative to normalization might also improve the scaling; the current values were chosen after limited experimentation with the 20 object set. It is possible that the ratio strategy, combined with a constraint on allowable normalization error would be superior. Alternatively, local density in the representation space (Krumhansl 1978) could be used to adjust weights during learning, recognition, or both to reduce false matches.

It may be that the biggest gain in performance from scaling up would be to simply improve the learning process, in particular the normalization or inter-view distance term. Analysis of the errors indicate that the views which are misjudged are typically outliers in a family of similar distributions for the rest of the views, but the system is unable to find a better solution. It is unclear whether this is dynamically not feasible, or due to inefficient learning. Relatively little effort has been spent to date on in the evolutionary search component of the system; some possible improvements are discussed in the last chapter.

More elaborate (and computationally intensive) measures could include

1. using a multi-scale lattice. To implement this would necessitate a higher performance implementation than the current Matlab system, which essentially precludes the use of wider convolution kernels; even with the nearest neighbor kernels, 85% of the computation time in training or search is spent in the diffusion step.
2. using some dynamical process (apart from the localized diffusion currently implemented) to distinguish the location of parts in space.
3. training multiple Soca style classifiers per object and changing the matching process to a majority vote of the top n responses, where n = the number of classifiers per object. This assumes that parameter sets which result in good normalization performance are dense in the parameter space. While there is currently no proof that this is so, some of the experimental results (multiple solutions to the ellipse ordering problem, relative independence of the quality of matching results to the order of objects during learning) suggest that a multiplicity of solutions to particular representation problems exist.

Explaining the Results

The functioning of the Soca network on the view-based recognition by normalization task can be explained on several levels. Clearly if a representation space over dynamical partition cells can be created with perfect normalization and the category boundaries well separated, the system is accounted for in the sense of its input / output performance. In the previous chapter on representation, I proposed a deeper explanation in terms of statistical language recognizers with a state flow resembling a probabilistic finite state automata, but in reality more precise in its discrimination ability.

Partition Cells are Not Spatially Consistent Across Views

Another level of explanation seemed required to assess what the system is "paying attention to" across views and across objects. .To address this question, plots were made of the spatial location of the highest population bins (normalized fraction > .03) across all seven views of an object (paperclip + 5.5). From this and similar plots, it appears that in the course of performing normalization (i.e. maintaining roughly the same distribution across views), there is little correspondence or registration of the partition cell components with particular "parts" or features across views. Instead, the network is able to find dynamics whose underlying state-flow graph reaches the same distribution for the particular starting distribution of initial "words", as they are arranged around the contour in overlapping windows.

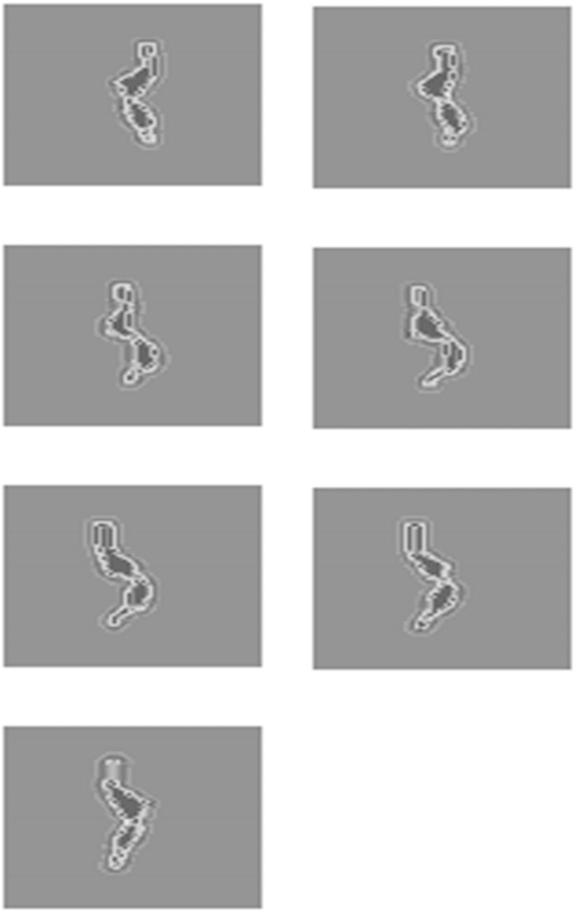


Fig. 43. The spatial locations of high population bins (> .03% of population) of paperclip object 5.5 are shown as a white overlay on the seven views of object 5.5 . Since the maximum occupancy of any partition cell is limited to .15 by the objective function, the light areas must be significant in mapping the object. Note that the entire contour seems to be represented, rather than any particular feature; based on a more detailed examination of locations for different thresholds (which can pick out the highest bin, etc) , there appears to be little consistency of the location of individual bins with particular features across views.

Visualizing Clusters by Multidimensional Scaling of the Representation Space

A second approach to understanding the dynamics was to examine the clustering of the objects in representation space. The following set of images shows the projection in two dimensions, obtained via multi-dimensional scaling (MDS), applied to a distance matrix obtained from the 64 bin representation vectors[31]. The constraints used in the formation of the representation space – minimizing differences across views and maximizing sum of entropies between classifiers – appear to result in a psychologically reasonable clustering and separation, with a few exceptions. No psychophysical testing has been performed, so at this point the reader must judge this by inspection. The requirement for normalization essentially forces the emergence of subordinate level clusters, as the dynamics is simply incapable of equally spacing representations for individual objects in the space under that constraint.

The cluster labeled 1 in the first consists of objects with few extra geons, minimal protrusions; those in cluster 2 have distinctive protruding geons, with several in the center of the chain. Within these clusters (and to some extent across clusters) the pattern appears to be something like the turning angle measure, or oriented run length. There are certainly instances in which I would cluster things differently given free choice to group the most similar object to any target, the paradigm used in the study of (Scasseleti, Alexopoulos et al. 1994). Performing a match – no match task stressing a pre-attentive response may give different results.

---

[31] A measure "stress" indicates the goodness of fit to a particular dimension in the MDS procedure; the stress for mapping to 2 dimensions was 3.4, which is higher than the value 2 which is often taken as a rule of thumb for a good fit. Stress under two was obtained for 3 dimensions.

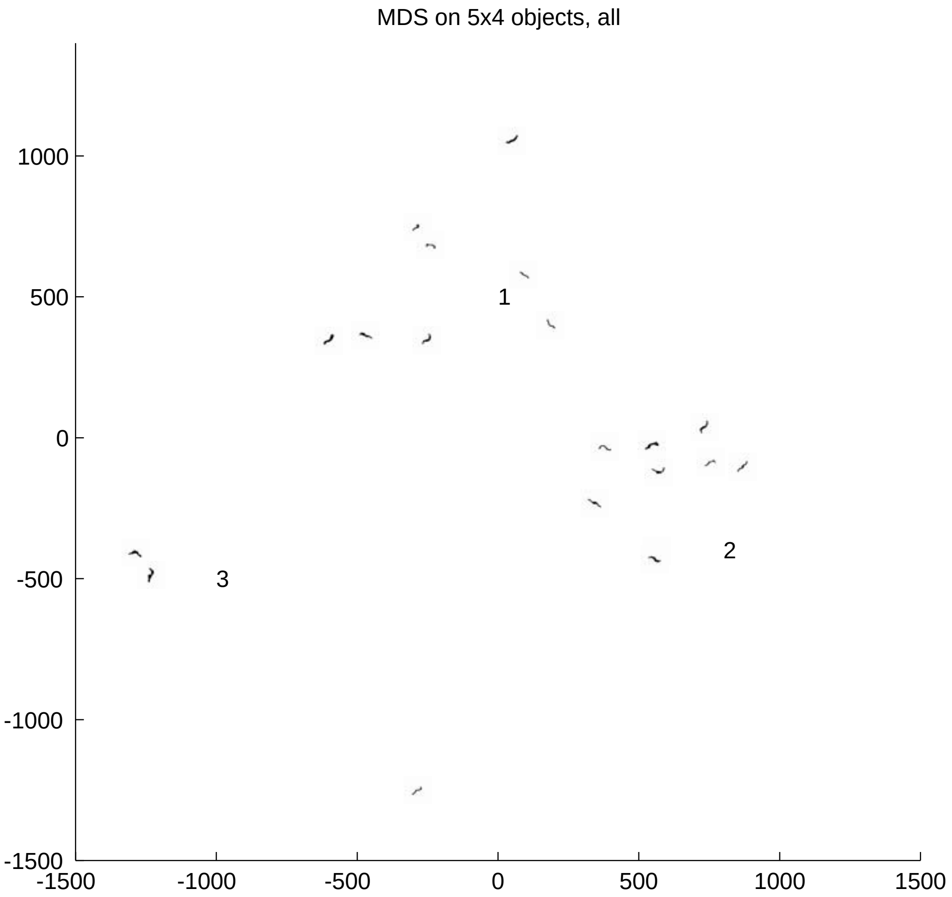
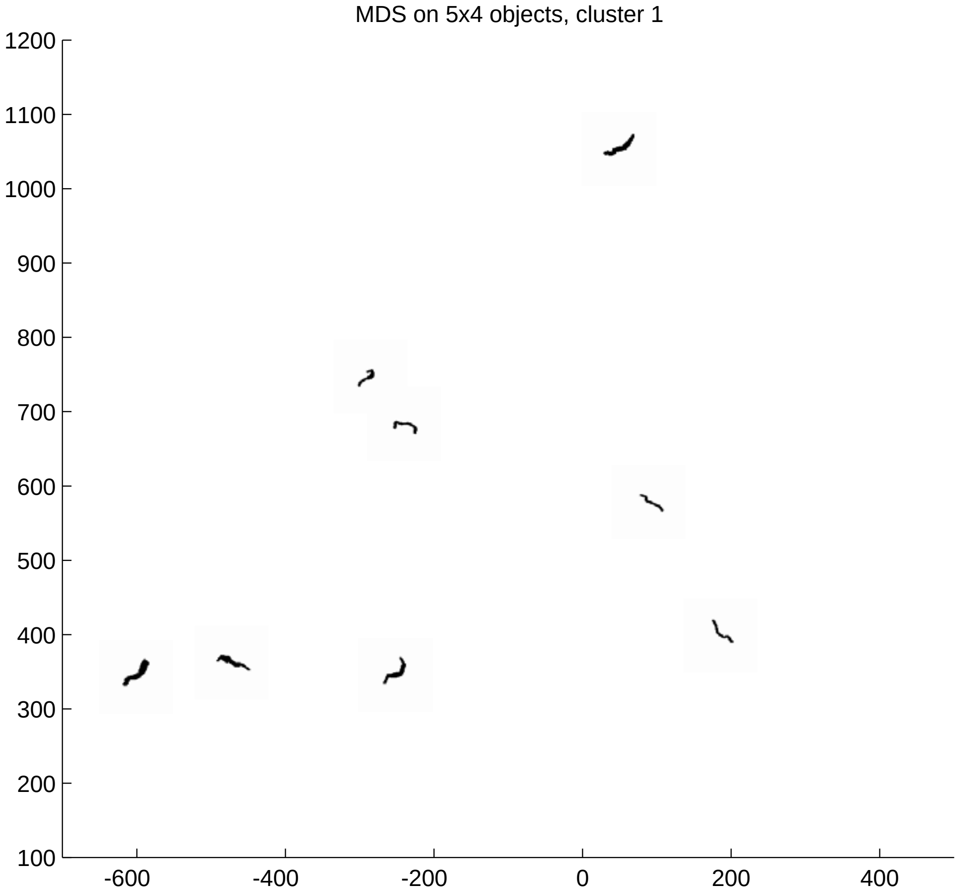


Fig. 44. Projection of distances into two dimensions via multi-dimensional scaling on inter-object distance matrix, with distributions computed for view invariance. Three clusters are seen; clusters one and two are enlarged below.


MDS on 5x4 objects, cluster 1
1200
1100
1000
900
800
700
600
500
400
300
200
100
-600
-400
-200
0
200
400

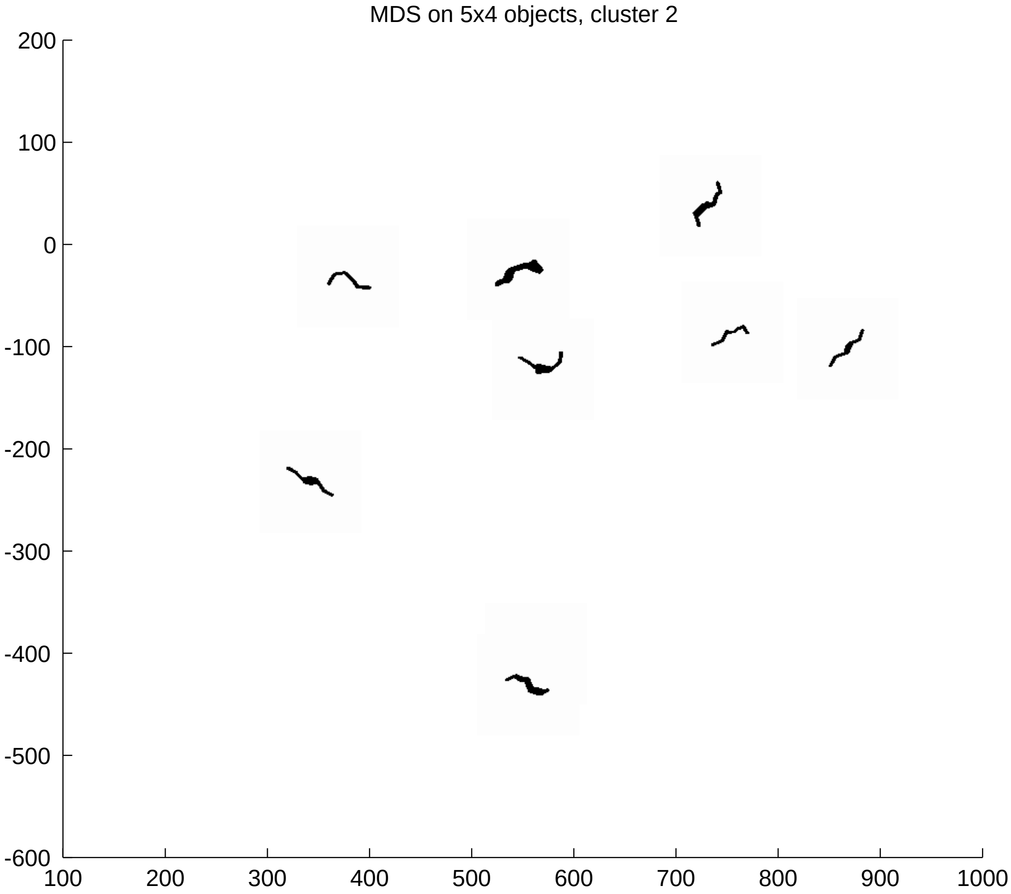


Fig. 45. Zooming in on two clusters in the representation space. The axes are arbitrary distance units determined from inter pair distances in a unit cube.

Comparing The Soca Classifier Approach With Other Systems

Exact comparison with the other systems reviewed earlier is difficult, as the training circumstances reported in each varied slightly. On the whole, the raw information presented for training to the Soca system is less than that provided for Chorus and SEEMORE, consisting of fewer, widely separated views and no shading or color data. The other systems have "front ends" consisting of more than one hundred local feature detectors. Soca, in contrast has no feature front end, only the asymmetric, diagonal-suppressed coupling. We might consider the simple threshold based conversion from shaded color to binary as a primitive front end.

I assume, of course that the non-trivial task of segmentation to produce a primal-sketch style image has been done by previous computational stages. All three systems have only been tested with isolated objects.

Error Rate Measures and Testing Conditions

The Chorus system (Duvdevani-Bar and Edelman 1999); (Edelman 1999), an ensemble of radial basis function (RBF) classifiers, was trained on a visual world consisting of 10 shaded objects readily assigned to categories such as quadrupeds, airplanes and cars. Test views of the objects, separated by 10° increments, were presented in a range of +/- 60° in azimuth and elevation, for a total of 16-17 views for each objects. Of several tests reported, the closest comparison to the present work is the recognition of untrained views of the 10 objects using a winner-take-all algorithm over individual RBF classifiers. In this case, the error rate for Chorus is 10%. Use of an additional competitive layer reduces the error to 7%.

This performance slightly exceeds that of the Soca system, but differences in complexity and information content of the raw data, view separation, and look ahead over the entire object set prior training make judgements about the superiority of either as a computer vision method difficult to assess without further study. Soca consistently exhibited its best performance on more complex objects in its training set, thus might perform better overall in a more complex object set; the effect on performance of providing more closely spaced views is unknown. Testing of Chorus on a set of objects with limited range like the paperclip set has not been performed.

The views selected for training the Chorus classifiers were chosen by an algorithm with access to all views of all objects in order to separate the clusters. No such global view was made available to Soca, which formed representations based on cross-entropy functions of objects representations already formed in an attempt to solve the same problem.

Error rates for a nearest neighbor match performance assessment of Mel's SEEMORE system using only the 79 shape channels alone is reported at 21.3% This rate was based on 12 rotation in depth views separated by around 60°; additional views for scaling were provided. Test target views *excluded* highly foreshortened views. The visual world in SEEMORE consisted of 100 images, with three scales provided for each training view, resulting in 3600 views total[32] if all objects were rigid. It is difficult to project how the performance of Soca or Chorus approaches would scale to that size.

Considering that Soca used more widely separated views, foreshortened views, no shading data, and no selection of preferred training views based on overall world statistics, the performance is nevertheless comparable to the other algorithms. The training procedure is rather arbitrarily truncated, and explores at most 3000 genotypes per classifier; more thorough exploration of the parameter space might improve the process of normalization across views. One could argue that the paperclip discrimination test is

---

[32] 3600 would be the number if all objects were rigid; additional views were provided for nonrigid objects.

the most difficult, given that human error rates range as high as chance for low complexity (tube only) geons with exposure to only single views (Tarr, Bulthoff et al. 1997). Clearly humans are already trained on the everyday objects used in Soca and SEEMORE, and it would be surprising to see substantial error rates for humans in match/no match tasks.

Memory Utilization

For each object represented in the Soca system, a vector of six parameters are stored specifying the dynamics for the two synchronization opponent stages. In addition, a signature corresponding to the means across all views of the sampled distribution after the specified number of iterations is stored. In the Matlab implementation, all floating point values are double precision; thus 48 bytes are used for the parameter set (genotype) and 512 bytes (64 bins x 8 bytes) for the signature. It seems unlikely that performance would suffer greatly by storing single or even lower precision values, since the signatures are not used in iterative calculations. As noted in the learning and representation selection, no precision exceeding $\log_2 \frac{1}{N}$, where N is the number of lattice units, is useful, since the occupancy of a phase space partition cell cannot be less than this unless it is zero.

A table below summarizes the relative memory utilization of these shape recognition systems. Note that the *minimum* error rate for Chorus and SEEMORE is not reported as more training views were available for Chorus and non-shape data was available to SEEMORE. The Chorus rate reflects fewer views (and less memory) to give an error rate comparable to Soca for memory comparison, while the SEEMORE error rate reflects performance and corresponding the memory used for the shape channels only. Both reports featured sufficient information on perfomance with limits on training to partially compensate for these differences in the following comparison.

Memory representations for Chorus involve specification of radial basis function centers, widths, and weights for each node. One center per view is typically used. Error rates depend on the number of views provided; to achieve an error rate of 15%, the data reported for Chorus suggest 5 views would be required, resulting in storage requirements of 2400 / bytes object[33] with double floating point representation of network parameters. For SEEMORE, the appropriate comparison (using only shape channels, representing only rigid objects, and not providing extra scaled versions during training) is 1264 bytes / object.

---

[33] 5 views x 3 (center,width,weight) x 8 bytes/double x 200 receptive field activations

Table 18. Memory utilization and error rates for Chorus, SEEMORE, Soca

System	Memory Bytes/Object	Error Rate %	Objects	Views per object
Chorus	2400	.15*	10	5
SEEMORE	632	.21	100	36
Soca	560	.15	20	7

A NOTE ON RECOGNITION PERFORMANCE

Performance of the recognition algorithm is a O(N), where N is the number of objects. Specifically, it is N*(M+D), where M is the computation time for the CML on the given view for t1+t2 iterations, and D is the time for the distance computation. This is dominated by the CML computation, and profiling shows that 85% of the time is spent in the averaging convolution function *filt2d*. Using 82x82 image arrays for the experiment reported in the next chapters , the Matlab implementation on a 400 MHz PPC 750 (1M cache, 100MHz system bus) resulted in a maximum CML computation time of 0.18 sec.