

Visual Development and the Acquisition of Motion Velocity Sensitivities

Robert A. Jacobs

Department of Brain and Cognitive Sciences

University of Rochester

Rochester, NY 14627

Melissa Dominguez

Department of Computer Science

University of Rochester

Rochester, NY 14627

September 2002

We thank the anonymous reviewers for their comments and suggestions which resulted in important improvements to this manuscript. This work was supported by NIH research grant R01-EY13149.

Abstract

We consider the hypothesis that systems learning aspects of visual perception may benefit from the use of suitably designed developmental progressions during training. Four models were trained to estimate motion velocities in sequences of visual images. Three of the models were “developmental models” in the sense that the nature of their visual input changed during the course of training. These models received a relatively impoverished visual input early in training, and the quality of this input improved as training progressed. One model used a coarse-to-multiscale developmental progression (meaning that it received coarse-scale motion features early in training and finer-scale features were added to its input as training progressed), another model used a fine-to-multiscale progression, and the third model used a random progression. The final model was non-developmental in the sense that the nature of its input remained the same throughout the training period. The simulation results show that the coarse-to-multiscale model performed best. Hypotheses are offered to account for this model’s superior performance, and simulation results evaluating these hypotheses are reported. We conclude that suitably designed developmental sequences can be useful to systems learning to estimate motion velocities. The idea that visual development can aid visual learning is a viable hypothesis in need of further study.

1 Introduction

With relatively few exceptions, relationships between development and learning have largely been ignored by the neural computation community. This is surprising because development may be nature’s way of biasing biological learning systems so that they achieve better performance, and may represent an effective means for engineers to bias machine learning systems. It is well known in the machine learning literature that learning systems are inherently faced with the bias-variance dilemma (Geman, Bienenstock, and Doursat, 1995). Systems with little or no bias tend to interpolate in unpredictable ways and, thus, have highly variable generalization performance. Systems with larger bias, in contrast, tend to show better generalization performance when exposed to those training sets that they can adequately learn. We speculate that development may be an effective means of adding suitable bias to a system thereby enhancing the generalization performance of that system.

In previous work, we studied the effects of different types of developmental sequences on the performances of systems trained to estimate the binocular disparities present in pairs of visual images (Dominguez and Jacobs, 2002a, 2002b). The systems consisted of three components. The first component was a pair of right-eye and left-eye images. For example, the images may have depicted a light or dark object against a gray background. The second component was a set of binocular energy filters. These filters are widely used to model the binocular sensitivities of simple and complex cells in primary visual cortex of primates (Ohzawa, DeAngelis, and Freeman, 1990). Based on local patches of the right-eye and left-eye images, each filter acted as a disparity feature detector at a coarse, medium, or fine scale depending on whether the filter was tuned to a low, medium, or high spatial frequency, respectively. The third component was an artificial neural network. The outputs of the binocular energy filters were the inputs to this network. The network was trained to

estimate the disparity of the object which was defined as the amount that the object was shifted between the right-eye and left-eye images.

A non-developmental system was compared to three developmental systems. The network of the non-developmental system received the outputs of all binocular energy filters throughout the entire training period. The networks of the developmental systems, in contrast, were trained in three stages. The network of the coarse-to-multiscale system received the outputs of binocular energy filters tuned to a low spatial frequency during the first training stage. It received the outputs of filters tuned to low and medium spatial frequencies during the second training stage, and it received the outputs of all filters during the third training stage. The network of the fine-to-multiscale system was trained in an analogous way, though its filters were added in the opposite order. This network received the outputs of filters tuned to a high frequency during the first training stage, and the outputs of lower-frequency filters were added during subsequent stages. The network of the random developmental model was also trained in stages, though its inputs were chosen at random at each stage and, thus, were not organized by spatial frequency content.

The results show that the coarse-to-multiscale and fine-to-multiscale systems consistently outperformed the non-developmental and random developmental systems. The fact that they outperformed the non-developmental system is important because this demonstrates that models that undergo a developmental maturation can acquire a more advanced perceptual ability than one that does not. The fact that they outperformed the random developmental system is important because this demonstrates that not all developmental sequences can be expected to provide performance benefits. To the contrary, only sequences whose characteristics are matched to the task should lead to superior performance. In conjunction with other results not described here, these findings suggest that the most successful systems at learning to detect binocular disparities are systems that are exposed to visual inputs at a

single scale early in training, and for which the resolution of their inputs progresses in an orderly fashion from one scale to a neighboring scale during the course of training.

At a more general level, these results suggest that the idea that visual development aids visual learning is a viable hypothesis in need of further study. This paper studies this hypothesis in the context of visual motion velocity estimation. There are enough similarities between the tasks of binocular disparity estimation and motion velocity estimation that it is reasonable to believe that a developmental approach that is useful for the former task may also be advantageous for the latter task. However, the differences between the two tasks mean that this belief needs to be checked. Indeed, as discussed below, our simulations show that the two tasks do not yield the same pattern of results. Although a developmental approach to the velocity estimation task is shown to be beneficial, it is not the case that all developmental progressions that lead to performance advantages on the disparity estimation task also lead to advantages on the velocity estimation task. In particular, a coarse-to-multiscale developmental system outperformed non-developmental and random developmental systems on the velocity estimation task, but a fine-to-multiscale system did not. We hypothesize that the performance advantage of the coarse-to-multiscale system relative to the fine-to-multiscale system is due to the fact that the coarse-to-multiscale system learned to make greater use of motion energy filters tuned to a low spatial frequency. Analyses suggest that coarse-scale motion features are more informative for the velocity estimation task than fine-scale features.

2 Developmental and Non-developmental Systems

The structure of the developmental and non-developmental systems is illustrated in Figure 1. The input to each system was a sequence of 88 retinal images where each image was a one-dimensional array 40 pixels in length. As described below, this sequence depicted an object

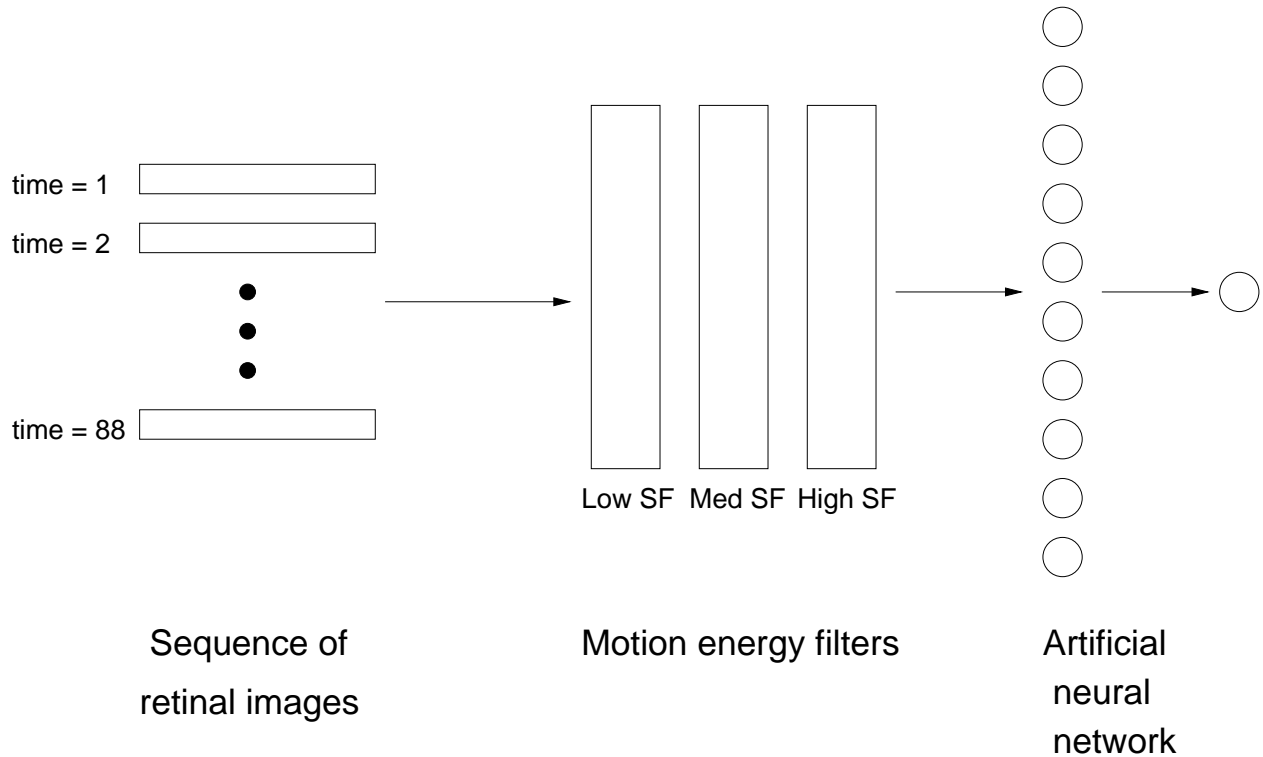


Figure 1: The developmental and non-developmental systems shared a common structure. The input to the systems was a sequence of retinal images. This sequence was filtered by motion energy filters. The set of filters can be partitioned into subsets tuned to low, medium, and high spatial frequencies. The outputs of the filters were the inputs to an artificial neural network that was trained to estimate the object velocity depicted in the image sequence.

moving at a constant velocity in front of a stationary background. The retinal array was treated as if it were shaped like a circle in the sense that the leftmost and rightmost pixels were regarded as neighbors. This wraparound of the left and right edges was done to avoid edge artifacts. Although a one-dimensional retina is a simplification, its use is justified by the need to keep the simulation times within reason. The sequence of retinal images was filtered using motion energy filters.

Based on neurophysiological results, Adelson and Bergen (1985) proposed motion energy filters as a way of modeling the motion sensitivities of simple and complex cells in

primary visual cortex. A sequence of one-dimensional images can be represented using a two-dimensional array where one dimension encodes space and the other dimension encodes time. In this case, motion energy filters are two-dimensional filters which extract motion information in local patches of the spatiotemporal space.

The receptive field profile of a simple cell can be described mathematically as a Gabor function which is a sinusoid multiplied by a Gaussian envelope. A quadrature pair of such functions with even and odd phases tuned to leftward (-) and rightward (+) directions of motion is given by

$$g_e^\pm = \frac{1}{2\pi\sigma_x\sigma_t} \exp\left\{-\frac{x^2}{2\sigma_x^2} - \frac{t^2}{2\sigma_t^2}\right\} \cos(\omega_x x \pm \omega_t t) \quad (1)$$

$$g_o^\pm = \frac{1}{2\pi\sigma_x\sigma_t} \exp\left\{-\frac{x^2}{2\sigma_x^2} - \frac{t^2}{2\sigma_t^2}\right\} \sin(\omega_x x \pm \omega_t t) \quad (2)$$

where x and t are the spatial and temporal distances to the center of the Gaussian, σ_x^2 and σ_t^2 are the spatial and temporal variances of the Gaussian, and ω_x and ω_t are the spatial and temporal frequencies of the sinusoids. The ratio ω_t/ω_x determines the orientation of a Gabor function in the spatiotemporal space which, in turn, determines the velocity sensitivity of the function.

The activity of a simple cell is given by the square of the convolution of the cell's receptive field profile with the spatiotemporal pattern, denoted f . For example, $(f * g_e^+)^2$ is the activity of a simple cell with even phase that is sensitive to rightward motion. The activities of simple cells with even and odd phases are summed in order to form the activity of a complex cell. This sum is known as a motion energy. Thus leftward and rightward motion energies are given by

$$E^- = (f * g_e^-)^2 + (f * g_o^-)^2 \quad (3)$$

$$E^+ = (f * g_e^+)^2 + (f * g_o^+)^2. \quad (4)$$

Because a complex cell’s activity is based on the combined properties of simple cells with even and odd phases, this activity is phase-insensitive meaning that this value is relatively insensitive to the exact position of a motion within the complex cell’s receptive field.

In our simulations, we used a subset of the possible receptive-field locations in the two-dimensional (40 pixels \times 88 time frames) spatiotemporal space. This subset formed a 20×4 uniform grid located in the middle of the space such that receptive-fields were centered on odd-numbered pixels and odd-numbered time frames. An advantage of this choice of locations was that edge artifacts were avoided because all receptive-fields fell entirely within the spatiotemporal space.

Fifteen complex cells corresponding to three spatial frequencies and five temporal frequencies were centered at each receptive-field location. All cells were tuned to rightward motion because we restricted our data sets to only include objects that were moving to the right. The parameter values of these cells are shown in Table 1. The spatial and temporal frequencies were each separated by an octave. Importantly, temporal frequencies were chosen so that the set of cells at each spatial frequency had the same pattern of velocity tunings. Specifically, the sets tuned to low, medium, and high spatial frequencies had velocity tunings of 0.25, 0.5, 1.0, 2.0, and 4.0 pixels per time frame. This was achieved by correlating the spatial and temporal frequency tunings of the cells; i.e. low spatial frequency cells were tuned to a comparatively low range of temporal frequencies, and high spatial frequency cells were tuned to a high range of temporal frequencies.¹ A cell’s spatial and temporal stan-

¹Recent neuroscientific studies indicate that the preferred velocity of velocity-tuned neurons in area MT of primates is largely independent of spatial frequency (Perrone and Thiele, 2001). In our simulated models, the filters at different spatial frequency bands are sensitive to the same set of velocities and, thus, it is relatively easy for our models to learn to have this same property; i.e. a network could easily learn to represent the

dard deviations were set to be inversely proportional to its spatial and temporal frequencies, respectively.

The outputs of the complex cells within each spatial frequency band were normalized using a softmax nonlinearity:

$$\hat{E}(\omega_{x_i}, \omega_{t_j}) = \frac{e^{E(\omega_{x_i}, \omega_{t_j})/\tau}}{\sum_k e^{E(\omega_{x_i}, \omega_{t_k})/\tau}} \quad (5)$$

where ω_{x_i} and ω_{t_j} are the spatial and temporal frequencies to which a complex cell was tuned, $E(\omega_{x_i}, \omega_{t_j})$ was the initial output of the complex cell, $\hat{E}(\omega_{x_i}, \omega_{t_j})$ was the normalized output, τ is a scaling parameter (its value was set to 0.001), and ω_{t_k} , $k = 1, \dots, 5$, indexed the 5 temporal frequencies corresponding to spatial frequency ω_{x_i} . As a result of this normalization, complex cells tended to respond to relative contrast in the image sequence rather than absolute contrast (Heeger, 1992; Nowlan and Sejnowski, 1994).

The normalized outputs of the complex cells were the inputs to an artificial neural network. The network had 1200 input units (the complex cells had 80 receptive-field locations and there were 15 cells at each location). The network’s hidden layer contained 18 hidden units which were organized into 3 groups of 6 units each. The connectivity of the hidden units was set so that each group had a limited receptive field, and so that neighboring groups had overlapping receptive fields. A group of hidden units received inputs from thirty-two receptive field locations at the complex cell level, and the receptive fields of neighboring groups overlapped by eight receptive-field locations. The hidden units used a logistic activation function of an object independent of the spatial frequency content of the object’s luminance values. The pairing of spatial and temporal frequencies in the models resembles a suggestion by Simoncelli and Heeger (2001) that a velocity-tuned MT neuron pools the outputs of a set of V1 cells whose spatial and temporal frequency tunings are positively correlated. It may be appropriate, therefore, to think of our motion energy filters as being more analogous to MT neurons than to complex neurons in area V1.

Table 1: These tables provide the parameter values for the complex cells at each receptive-field location. The top table shows the three spatial frequencies and their corresponding temporal frequencies. The temporal frequencies were chosen so that the same velocity tunings existed at each spatial frequency. The middle and bottom tables show the spatial standard deviation corresponding to each spatial frequency, and the temporal standard deviation corresponding to each temporal frequency, respectively.

<i>spatial freq (cycles/pixel)</i>	<i>temporal freq (cycles/time frame)</i>	<i>velocity (pixels/ time frame)</i>
0.0625	0.015625	0.25
	0.03125	0.5
	0.0625	1.0
	0.125	2.0
	0.25	4.0
0.125	0.03125	0.25
	0.0625	0.5
	0.125	1.0
	0.25	2.0
	0.5	4.0
0.25	0.0625	0.25
	0.125	0.5
	0.25	1.0
	0.5	2.0
	1.0	4.0

<i>spatial freq (cycles/pixel)</i>	<i>spatial standard deviation (σ_x)</i>
0.0625	4.0
0.125	2.0
0.25	1.0

<i>temporal freq (cycles/time frame)</i>	<i>temporal standard deviation (σ_t)</i>
0.015625	16.0
0.03125	8.0
0.0625	4.0
0.125	2.0
0.25	1.0
0.5	0.5
1.0	0.25

vation function. The output layer consisted of a single linear unit; this unit's output was an estimate of the object velocity depicted in the sequence of retinal images.

The weights of an artificial neural network were initialized to small random values, and were adjusted during the course of training to minimize a sum of squared error cost function using a conjugate gradient optimization procedure (Press, Teukolsky, Vetterling, and Flannery, 1992). Advantages of this procedure are that it tends to converge quickly and it has no free parameters (e.g., no learning rate or momentum parameters). Weight sharing was implemented at the hidden unit level so that corresponding units within each group of hidden units had the same incoming and outgoing weight values, and so that a hidden unit had the same set of weight values from each receptive field location at the complex unit level. This provided the network with a degree of translation invariance, and also dramatically decreased the number of modifiable weight values in the network. It therefore decreased the number of data items needed to train the network, and the amount of time needed to train the network.

Models were trained and tested using separate sets of training and test data items. Each set contained 250 randomly generated items. Training was terminated after 100 iterations through the training set. The results reported below are based on the data items from the test set.

Three developmental systems and one non-developmental system were simulated. The *coarse-to-multiscale* system, or model C2M, was trained using a coarse-to-multiscale developmental sequence which was implemented as follows. The training period was divided into three stages. During the first stage, the neural network portion of the model only received the outputs of complex cells tuned to the low spatial frequency (the outputs of other complex cells were set to zero). During the second stage, the network received the outputs of complex cells tuned to low and medium spatial frequencies; it received the outputs of all

complex cells during the third stage. The training of the *fine-to-multiscale* system, or model F2M, was identical to that of model C2M except that its training used a fine-to-multiscale developmental sequence. During the first stage of training, its network received the outputs of complex cells tuned to the high spatial frequency. This network received the outputs of complex cells tuned to high and medium spatial frequencies during the second stage, and received the outputs of all complex cells during the third stage. The training of the *random developmental* system, or model RD, also used a developmental sequence, though this sequence was generated randomly and, thus, was not based on the spatial frequency tunings of the complex cells. The collection of complex cells was randomly partitioned into three equal-sized subsets with the constraint that each subset included one-third of the cells at each receptive-field location. During the first stage of training, the neural network portion of the model only received the outputs of the complex cells in the first subset. It received the outputs of the cells in the first and second subsets during the second stage of training, and received the outputs of all complex cells during the third stage. In contrast, the training period of the *non-developmental* system, or model ND, was not divided into separate stages; its neural network received the outputs of all complex cells throughout the entire training period.

3 Data Sets and Simulation Results

The performances of the four models were evaluated on two data sets. In all cases the images were gray scale with luminance values between 0 and 1, and motion velocities were rightward with magnitudes between 0 and 4 pixels per time frame. Fifteen simulations of each model on each data set were conducted.

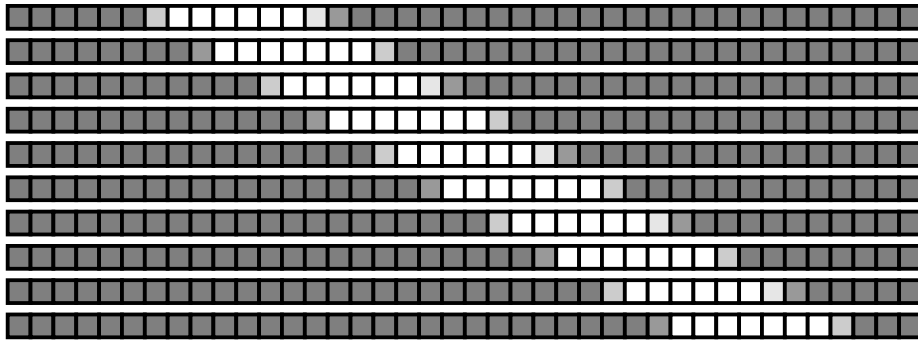
In the *solid object data set*, images consisted of a moving light or dark object in front of a stationary gray background. The object's gray-scale values were randomly chosen to

either be in the range from 0.0 to 0.1 or from 0.9 and 1.0, whereas the gray-scale value of the background was always 0.5. The size of the object was randomly chosen to be an integer between 6 and 12 pixels, its initial location was a randomly chosen pixel on the retina, and its velocity was randomly chosen to be a real value between 0 and 4 pixels per time frame. Because the velocity was real-valued, the boundaries between an object’s internal gray-scale values, or the boundaries between the ends of an object and the background, could fall at a real-valued location within a retinal pixel. In these (common) cases, the luminance value of a pixel was appropriately linearly interpolated. Given a sequence of images, the task of a model was to estimate the object’s velocity. The top portion of Figure 2 gives an example of ten frames of an image sequence from the solid object data set.

The bar graph in Figure 3 illustrates the results. The horizontal axis of each graph gives the model, and the vertical axis gives the root mean squared error (RMSE) on the data items from the test set at the end of training (the error bars give the standard error of the mean). The labels for the developmental models C2M, F2M, and RD include a number. Recall that the training of these models was divided into three training stages (or developmental stages). The number in the label gives the length of developmental stages 1 and 2 (the length of developmental stage 3 can be calculated using the fact that the entire training period lasted 100 iterations). For example, the label ‘C2M-5’ corresponds to a version of model C2M in which the first stage was 5 iterations, the second stage was 5 iterations, and the third stage was 90 iterations. In regard to model RD, we simulated four versions of this model (RD-5, RD-10, RD-20, and RD-30). For the sake of brevity, only the version that performed best is included in the graph.

Model C2M significantly outperformed all other models. The version of this model which performed best was version C2M-20 which had an 11.5% smaller generalization error than model ND (the difference between the mean performances of these models is statistically

Solid object data item



Noisy object data item

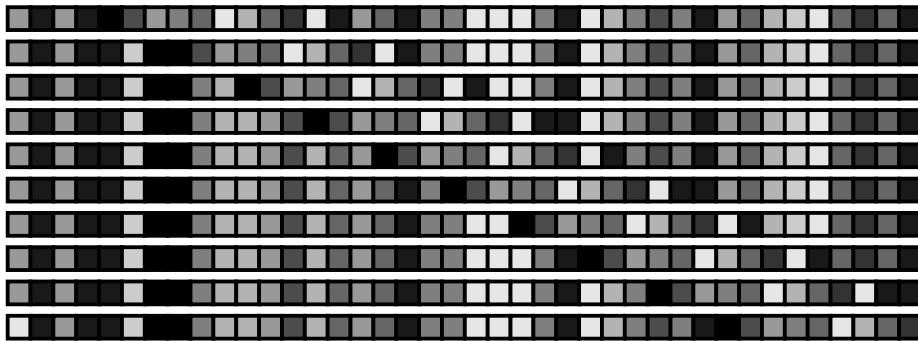


Figure 2: Ten frames of an image sequence from the solid object data set (top) and ten frames of an image sequence from the noisy object data set (bottom).

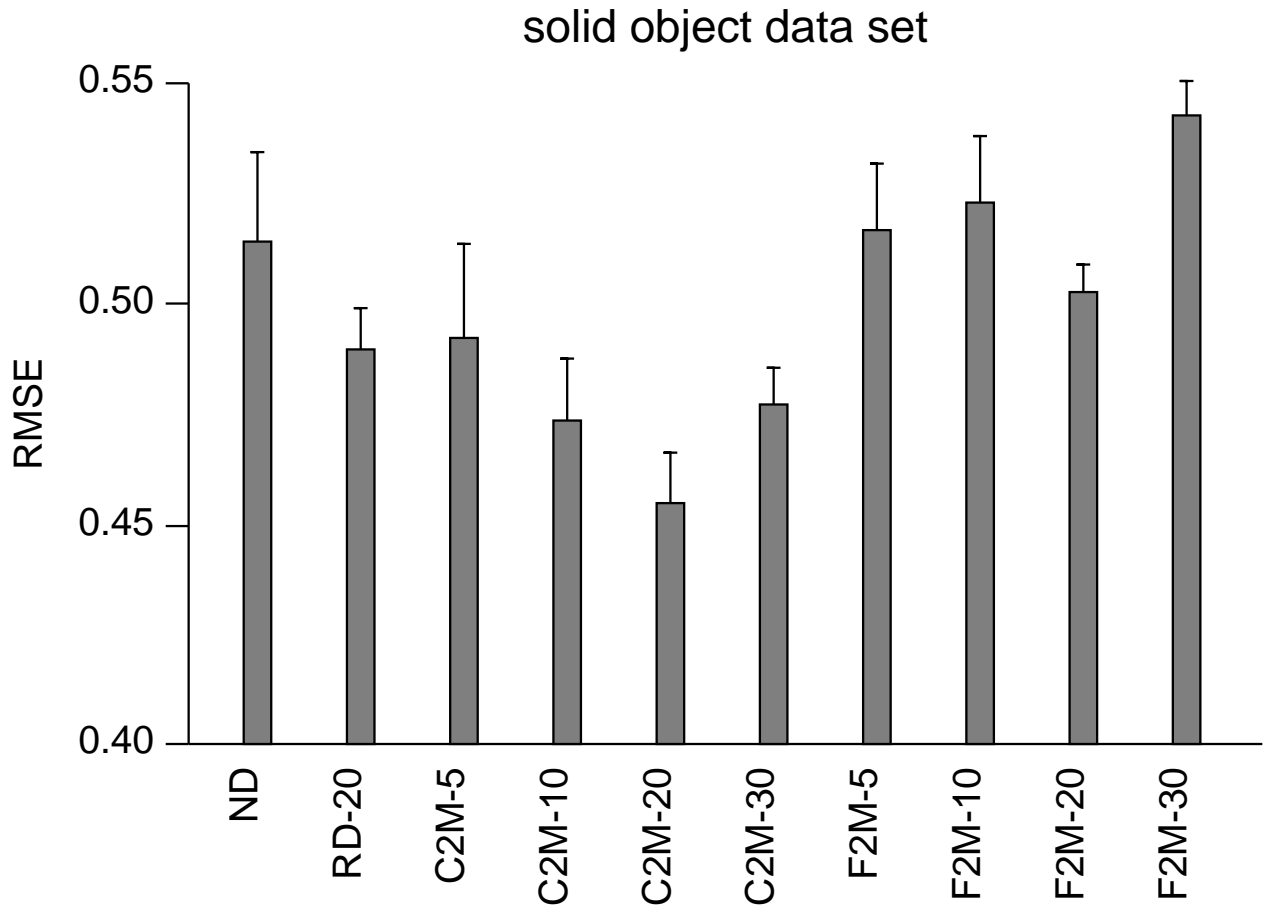


Figure 3: The root mean squared errors (RMSE) on the test set data items for model ND, the best performing version of model RD, and different versions of models C2M and F2M after training on the solid object data set (the error bars give the standard error of the mean).

significant; $t = 2.50$, $p < 0.02$ using a two-tailed t-test with 28 degrees of freedom). In addition, C2M-20 had a 9.6% smaller error than the best version of model F2M ($t = 3.57$, $p < 0.01$), and a 7.2% smaller error than the best version of model RD ($t = 2.30$, $p < 0.05$).

The images in the second data set, referred to as the *noisy object data set*, were meant to resemble random-dot kinematograms frequently used in behavioral experiments. Images contained a noisy object which was moving to the right and a noisy background which was stationary. The gray-scale values of the object pixels and the background pixels were set to random numbers between 0 and 1. The size of the object was randomly chosen to be an integer between 6 and 12 pixels, its initial location was a randomly chosen pixel on the retina, and its velocity was randomly chosen to be an integer between 0 and 4 pixels per time frame. As before, the task was to map an image sequence to an estimate of an object velocity. The bottom portion of Figure 2 gives an example of ten frames of an image sequence from the noisy object data set.

The results are shown in Figure 4. Model C2M, once again, outperformed the other models. Relative to model ND, all versions of model C2M showed superior performance (ND vs. C2M-5: $t = 2.69$, $p < 0.02$; ND vs. C2M-10: $t = 2.78$, $p < 0.01$; ND vs. C2M-20: $t = 3.03$, $p < 0.01$; ND vs. C2M-30: $t = 4.14$, $p < 0.001$). The version of model C2M which performed best was version C2M-30. On average, this version had an 8.9% smaller generalization error than model ND ($t = 4.14$, $p < 0.001$), a 6.1% smaller error than the best version of model F2M ($t = 2.95$, $p < 0.01$), and a 4.3% smaller error than the best version of model RD ($t = 2.10$, $p < 0.05$).

Taken as a whole, the simulation results are interesting for a number of reasons. Most importantly for our purposes, the fact that model C2M outperformed model ND is important because this demonstrates that a model that undergoes a developmental maturation

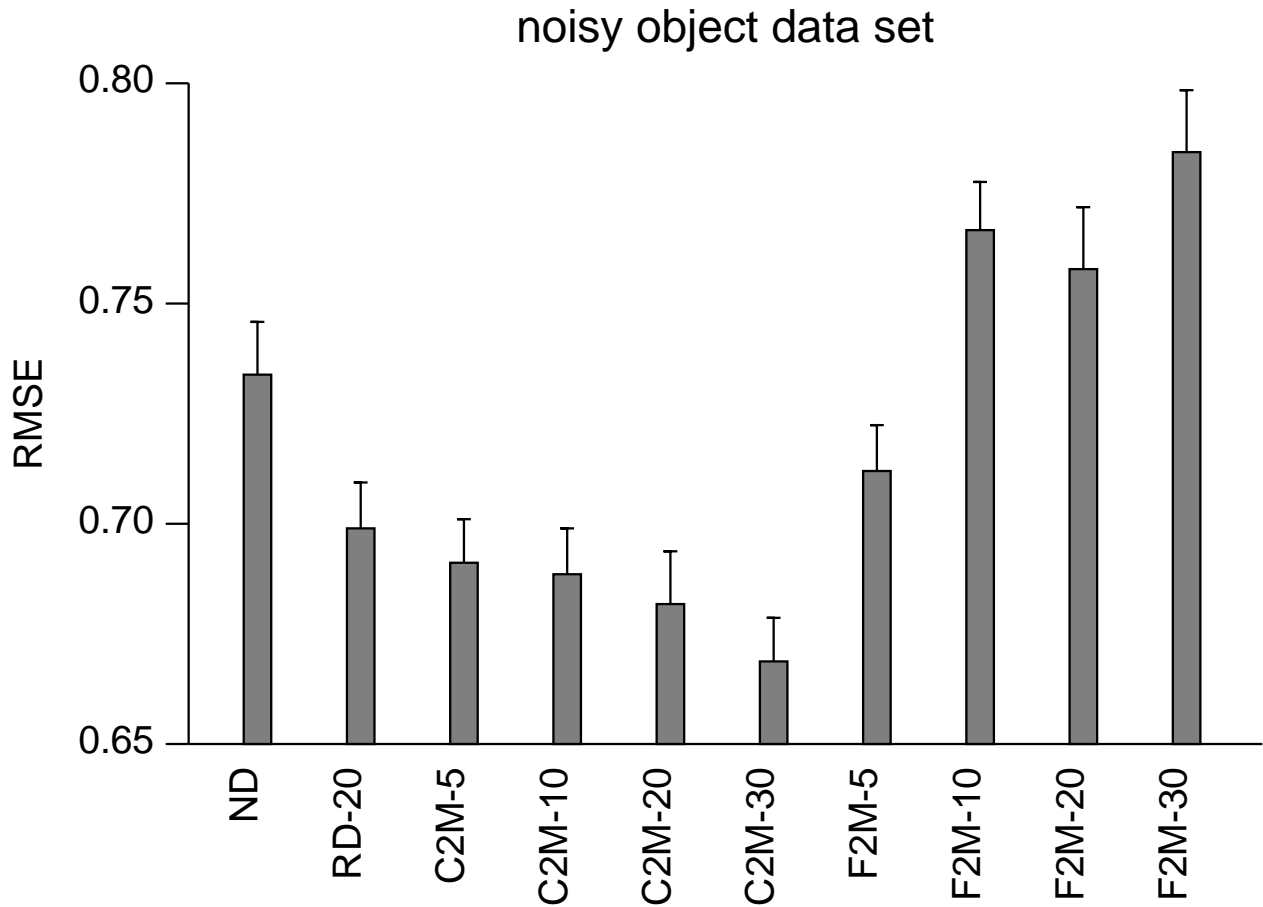


Figure 4: The root mean squared errors (RMSE) on the test set data items for model ND, the best performing version of model RD, and different versions of models C2M and F2M after training on the noisy object data set (the error bars give the standard error of the mean).

can acquire a more advanced perceptual ability than one that does not. The fact that model F2M performed similarly to or worse than model ND, and worse than model C2M, is important because this demonstrates that not all developmental sequences provide performance benefits. It is tempting to hypothesize that only those sequences whose characteristics are matched to the task should lead to superior performance. However, the finding that the best version of model RD outperformed model ND is inconsistent with this hypothesis because it is difficult to understand why a random developmental progression would be well-matched to a velocity estimation task. Future work will need to examine this unexpected finding.

To understand these results better, we conducted additional simulations. We compared the performances of several different types of neural networks on the solid object and noisy object data sets. These networks were not developmental systems; each received the same set of inputs throughout the entire training period. Some networks received only the outputs of the low spatial frequency motion energy filters (labeled ‘coarse’ networks), only the outputs of medium frequency filters (labeled ‘medium’ networks), or only the outputs of high frequency filters (labeled ‘fine’ networks). Other networks received the outputs of multiple subsets of filters; for example, ‘C,M’ networks received the outputs of low and medium frequency filters and ‘C,M,F’ networks received the outputs of all filters (identical to model ND). Lastly, the inputs to ‘random’ networks were the outputs of a randomly selected one-third of the motion filters. Furthermore, we examined the performances of these networks at a relatively early point in their training periods (after 10 epochs) and also at the end of training (after 100 epochs). The results at a relatively early point in training are shown in Figure 5. The results at the end of training on the solid object and noisy object data sets are shown in Figures 6 and 7 respectively.

For our purposes, the most important result to emerge from the data in these figures is that motion features at different scales are not equally informative for the task of velocity

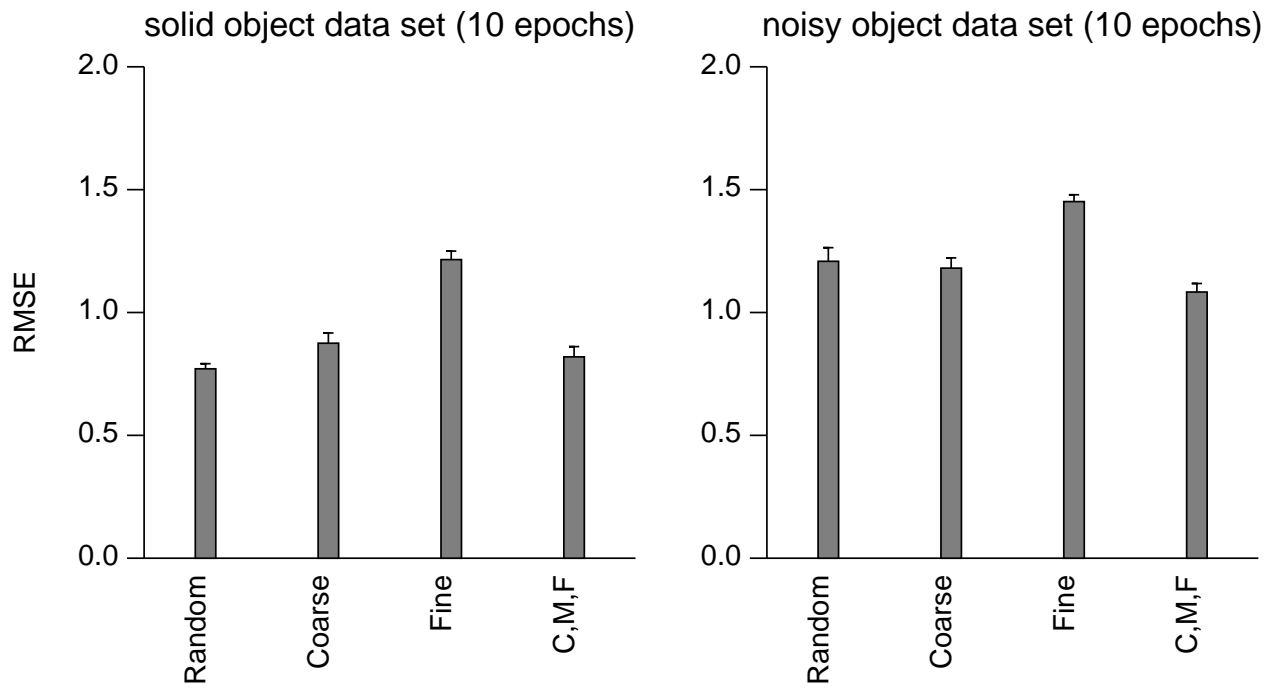


Figure 5: The performances of several networks after 10 epochs of training. Their root mean squared errors (RMSE) on the test set data items are shown for the solid object data set (left graph) and for the noisy object data set (right graph).

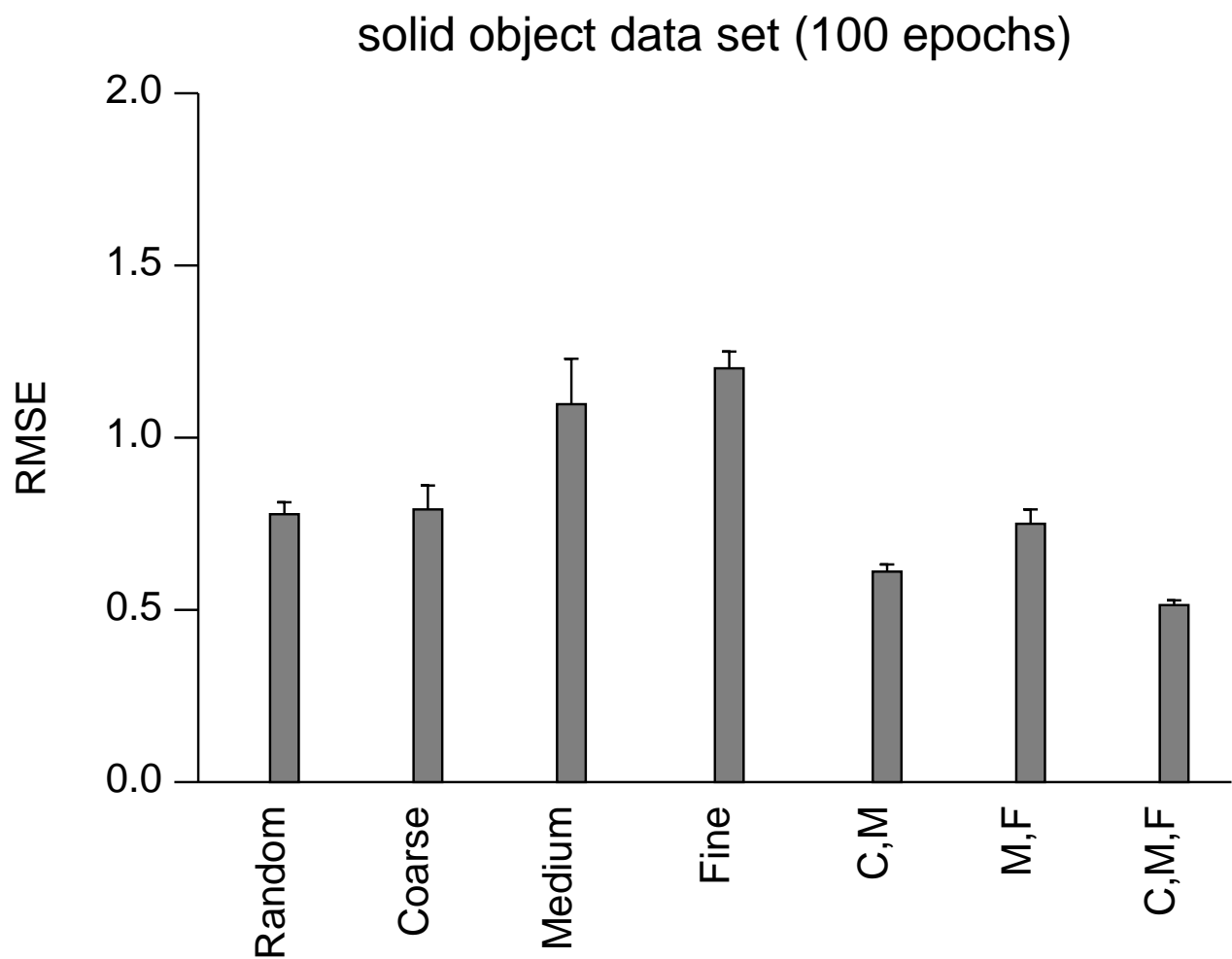


Figure 6: The root mean squared errors (RMSE) of several networks after 100 epochs of training on the solid object data set.

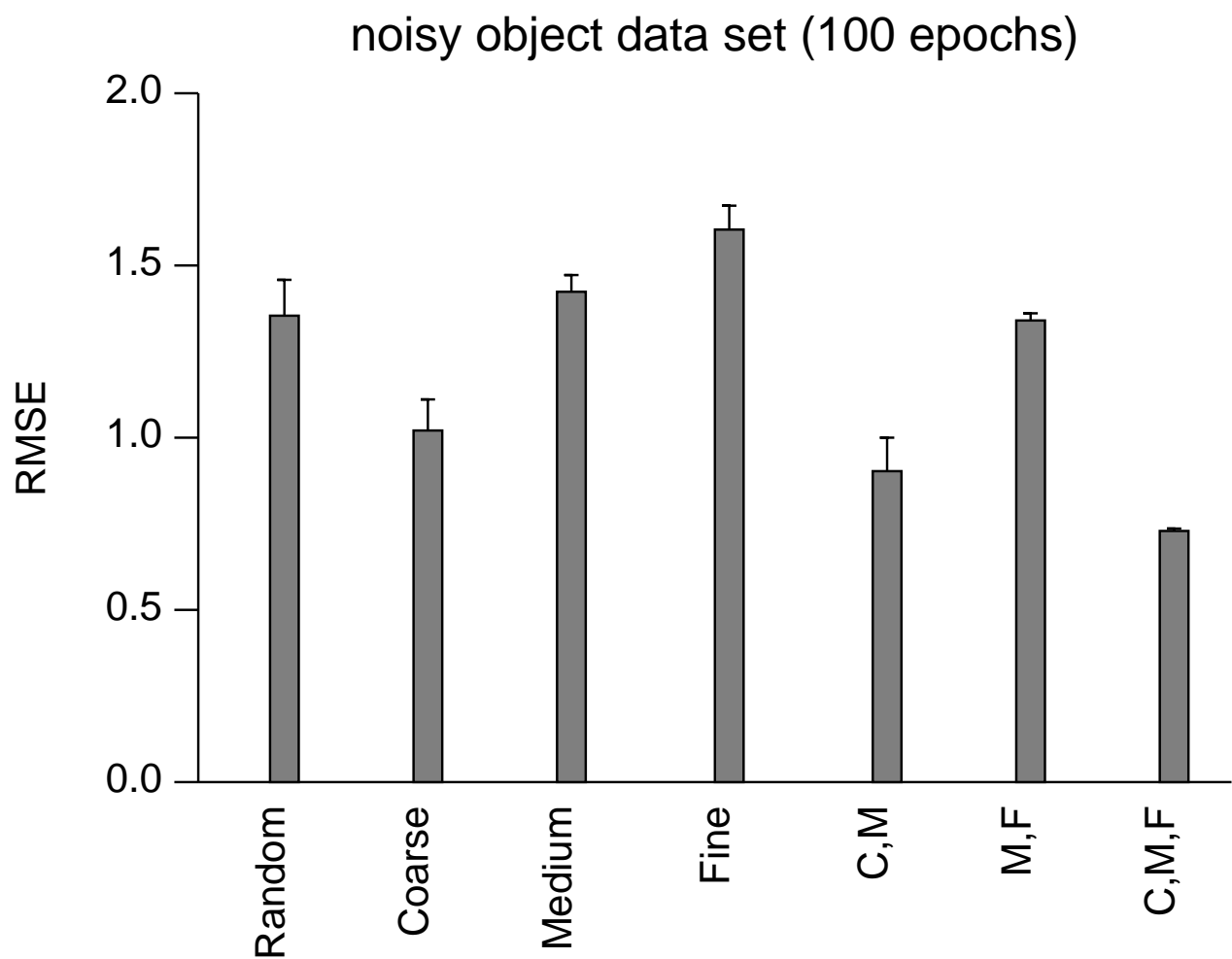


Figure 7: The root mean squared errors (RMSE) of several networks after 100 epochs of training on the noisy object data set.

estimation. To the contrary, coarse motion features (i.e. the outputs of low spatial frequency motion energy filters) are more informative than medium-scale motion features which, in turn, are more informative than fine motion features. Evidence for this includes the facts that the coarse network outperformed the medium network, and the medium network outperformed the fine network. In addition, the C,M network performed better than the M,F network. These performance rankings were found using both solid object and noisy object data sets, and were found at an early point in training and at the end of training.

Based on these results, we can speculate as to why model C2M showed the best performance and why model F2M showed comparatively poor performance. Because the only inputs to model C2M at the start of training were the outputs of the coarse motion energy filters, and because these outputs were the only set that it received at all stages of training, it is likely that this model made extensive use of these signals. The analyses above suggest that the coarse motion signals are the most informative for the velocity estimation task. Model C2M's extensive use of these highly informative signals presumably allowed it to achieve a high level of performance. In contrast, model F2M is likely to have made greater use of the outputs of fine motion energy filters because these outputs were its only inputs at the start of training, and because these outputs were the only set that it received at all stages of training. The analyses above suggest that the fine motion signals are the least informative for the velocity estimation task. Model F2M's extensive use of these least informative signals presumably is responsible for this model's comparatively poor performance.

As described above, in earlier work we found that the most successful systems at learning a binocular disparity estimation task were those that: (1) received inputs at a single frequency scale early in training, and (2) for which the resolution of their inputs progressed in an orderly fashion from one scale to a neighboring scale during the course of training (Dominguez and Jacobs, 2002a, 2002b). Condition (1) allowed a system to combine and compare input

features at an early training stage without the need to compensate for the fact that these features could be at different spatial scales. If condition (2) was satisfied, when a system received inputs at a new spatial scale, it was close to a scale with which the system was already familiar.

To test the importance on the velocity estimation task for the resolution of a system’s inputs to progress in an orderly fashion from one scale to a neighboring scale, we compared the performances of five systems. Three of the systems, ND, C2M-20, and F2M-20, were described above; the other two systems are new. The neural network of model C-CF-CMF-20 received coarse motion features during the first developmental stage. It received coarse and fine motion features during the second developmental stage, and coarse, medium, and fine features during the third stage. The neural network of model F-CF-CMF-20 received fine motion features in stage one, coarse and fine features in stage two, and coarse, medium, and fine features during the final developmental stage. Because the inputs to systems C-CF-CMF-20 and F-CF-CMF-20 did not proceed from one scale to a neighboring scale, we predicted that these systems would perform poorly.

The results on the solid object and noisy object data sets are shown in Figure 8. On the solid object data set, C2M-20 outperformed C-CF-CMF-20, and F2M-20 outperformed F-CF-CMF-20. On the basis of this data, we conclude that it is important for a system’s inputs to proceed from one scale to a neighboring scale. On the noisy object data set, the results are different. The performances of C2M-20 and C-CF-CMF-20 are similar, and F2M-20 performed worse than F-CF-CMF-20. We do not believe that these results are necessarily inconsistent with the hypothesis that we are currently considering. Instead the results may indicate that on this task it is more important for a system to receive the outputs of the low spatial frequency motion filters as early in training as possible than it is for a system to receive inputs whose resolution changes in an orderly manner. Overall, we believe that our

results imply that it is moderately important, but not highly important, for a developmental system learning to estimate motion velocities to receive inputs whose resolution progresses in an orderly fashion from one scale to a neighboring scale during training.²

We have presented analyses suggesting that coarser-scale motion features are more informative than finer-scale features on the velocity estimation task. Several factors may help explain this finding. As discussed by Weiss and Adelson (1998), motion signals tend to be less ambiguous when the stimulus is viewed for a long duration and more ambiguous when the stimulus is viewed for a short duration. Their reasoning applies to the activities of motion energy filters with receptive fields in the spatiotemporal domain. Coarse-scale filters tend to have larger receptive fields than those of fine-scale filters. Consequently, there is less ambiguity in the activities of coarse-scale filters relative to the activities of fine-scale filters. In addition, coarse-scale filters have large, overlapping receptive fields and, thus, form a high resolution coarse-code of the spatiotemporal space (Milner, 1974; Hinton, 1981; Ballard, 1986). This code could provide a network with accurate information as to the location of the moving object at each moment in time. For example, the activities of these filters may have coded with high accuracy the fact that the moving object was at location A at time

²Dominguez and Jacobs (2002a, 2002b) found that this was a highly important factor for a developmental system learning to estimate binocular disparities, whereas we find on a motion velocity estimation task that it is only a moderately important factor. Unfortunately, it is difficult to compare in a detailed way our current and prior findings for a number of reasons. First, whereas the disparity estimation task involved two images (left-eye and right-eye images), the motion estimation task involved a sequence of eighty-eight images. Second, differences between binocular energy filters and motion energy filters make the two sets of simulations difficult to compare. For instance, binocular energy filters make extensive use of the relative phases of filters, whereas motion energy filters do not. As a second example, the binocular energy filters that we used earlier were one-dimensional (we were only interested in horizontal disparities, not vertical disparities), whereas the motion energy filters used here were two-dimensional.

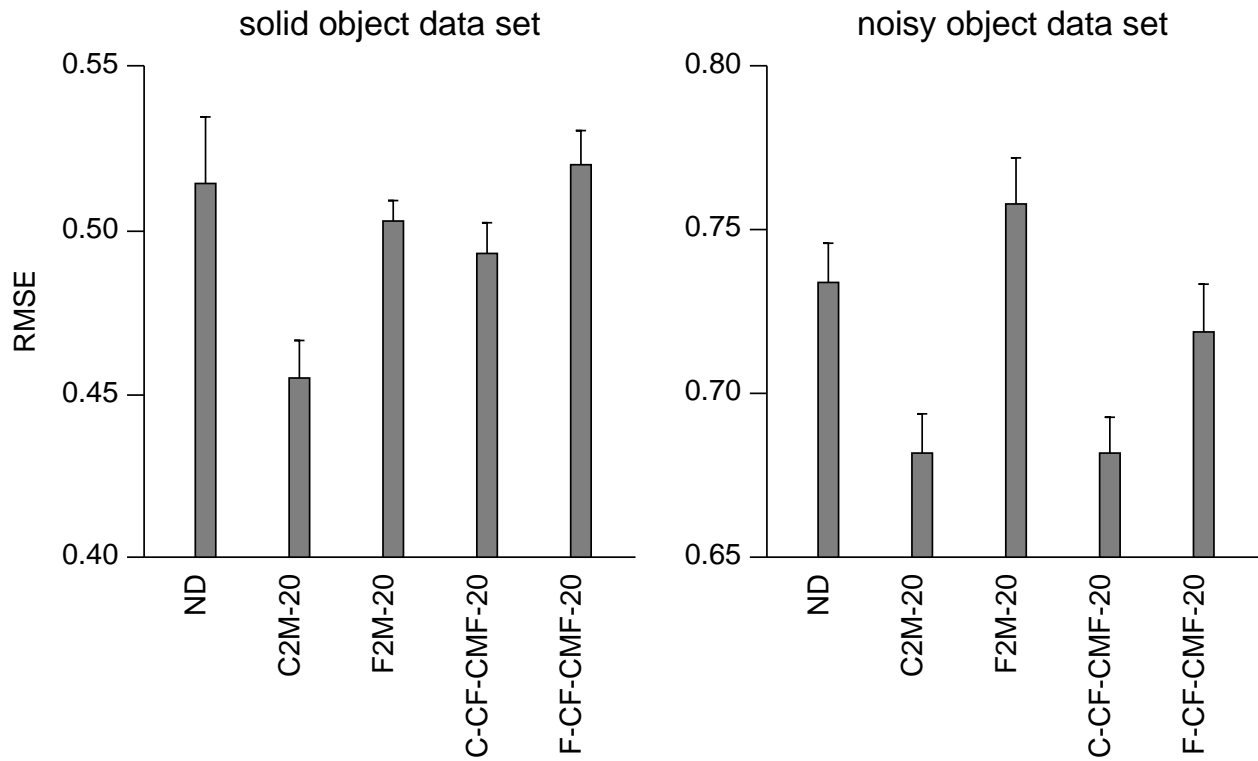


Figure 8: The root mean squared errors (RMSE) of several systems on the test set data items for the solid object data set (left graph) and the noisy object data set (right graph).

t_A and at location B at time t_B . If so, a network could have easily learned to accurately estimate the object velocity by calculating $(B - A)/(t_B - t_A)$. In contrast, fine-scale filters have smaller, less-overlapping receptive fields which form a lower resolution coarse-code.³

An interesting prediction follows from this line of reasoning about the advantages of filters with large, overlapping receptive fields. In general, filters with larger receptive fields tend to be tuned to slow velocities, whereas filters with smaller receptive fields tend to be tuned to fast velocities. Consequently, we predict that all models should be better at estimating slow velocities than fast velocities. However, we expect that this tendency will be comparatively strong for model C2M, because the receptive fields of low spatial frequency filters tuned to slow velocities are much bigger than those of low frequency filters tuned to fast velocities. In contrast, we expect that this tendency will be weak for model F2M, because the receptive fields of high frequency filters tuned to slow velocities are only mildly bigger than those of high frequency filters tuned to fast velocities.

To test this prediction, we evaluated the accuracy of seven systems at estimating slow (0.0 to 1.333 pixels per frame) and fast (2.667 to 4.0 pixels per frame) object velocities after training on the solid object data set. The results are shown in Figure 9. The horizontal axis gives both the system and the object velocity (s = slow, f = fast); the vertical axis gives the root mean squared error. All systems are generally good at estimating both slow and

³Analyses by Hinton (1981) and Ballard (1986) provide an understanding of the relationship between receptive field size and resolution in the case of binary units that each become active when a stimulus falls within its receptive field. If D is the diameter of a unit's receptive field, k is the dimensionality of the space to be represented, and N is the desired number of just-noticeable differences in each dimension (i.e. the desired resolution), then the required number of units is N^k/D^{k-1} . For a fixed number of units, a high resolution code (that is, one with a large N) requires units with large receptive fields (fields with a large D), whereas a low resolution code can be achieved with units with small receptive fields.

fast object velocities in the sense that they all show sub-pixel accuracy. Most importantly for our current purposes, models ND, RD, and C2M were more accurate at estimating slow velocities than at estimating fast velocities. This trend was strongest for model C2M, in agreement with the predicted outcome. Also in agreement with our prediction is the fact that model F2M showed roughly equal accuracy at estimating slow and fast velocities.

4 Concluding Remarks

In summary, we have compared four models on a visual motion velocity estimation task. Three of the models were “developmental models” in the sense that the nature of their visual input changed during the course of training. Model C2M used a coarse-to-multiscale developmental progression, meaning that it received coarse-scale motion features early in training and finer-scale features were added to its input as training progressed, model F2M used a fine-to-multiscale progression, and model RD used a random progression. The final model, model ND, was non-developmental in the sense that the nature of its input remained the same throughout the training period. The simulation results show that model C2M performed best, and model F2M often performed worst.

The fact that model C2M outperformed model ND is important because this demonstrates that a model that undergoes a developmental maturation can acquire a more advanced perceptual ability than one that does not. The fact that model F2M performed similarly to or worse than model ND, and worse than model C2M, is important because this demonstrates that not all developmental sequences provide performance benefits. It is tempting to hypothesize that only those sequences whose characteristics are matched to the task should lead to superior performance. However, the finding that the best version of model RD outperformed model ND is inconsistent with this hypothesis because it is difficult

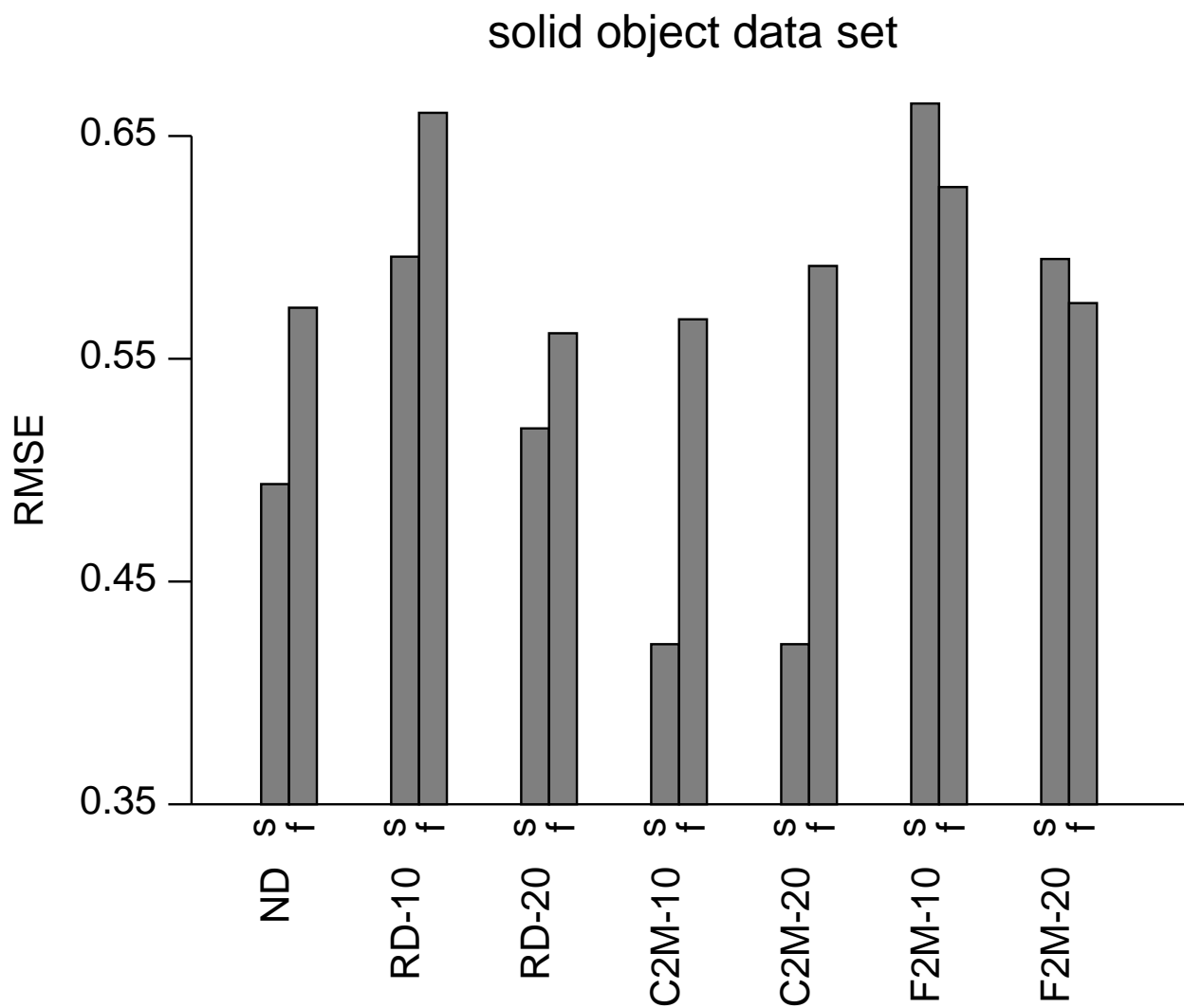


Figure 9: The root mean squared errors (RMSE) of several systems on the test set data items from the solid object data set with slow (s) or fast (f) object velocities.

to understand why a random developmental progression would be well-matched to a velocity estimation task. In general, we conclude that the idea that visual development can aid visual learning is a viable hypothesis in need of further study.

Model C2M’s superior performance is interesting because its developmental progression resembles that of human infants. The spatial acuity of newborns is roughly 1/15 to 1/30 that of adults with normal eyesight. In other words, newborns are sensitive only to low spatial frequencies; they cannot see fine details. Acuity improves approximately linearly from these low levels at birth to near adult levels by about eight months of age (Norcia and Tyler, 1985). Our simulation results suggest that this developmental sequence may provide important functional benefits for the acquisition of motion velocity estimation.

Model C2M’s superior performance is also interesting because this finding is broadly consistent with a theory of child development known as the “less is more” hypothesis (Newport, 1990). According to this view, cognitive, perceptual, and motor limitations early in infancy (such as being able to visually perceive only low spatial frequencies) are helpful, perhaps necessary, stages in development. Limited mental abilities reflect simple neural representations which are useful “stepping stones” or “building blocks” for the subsequent development of more complex representations (Turkewitz and Kenney, 1982).

Consistent with the “less is more” hypothesis, our view of perceptual development is that early developmental periods “set the stage” for later periods in the sense that early periods bias the performance of a biological system during these later periods. Although machine learning researchers rarely use the term ‘development’ when referring to their learning algorithms, many researchers have pursued a “stage setting” strategy. Some examples include:

- Systems that perform clustering frequently are trained in two stages. During the first stage, the K-means algorithm is used to approximately locate the cluster centers. These

centers are then used to initialize the mean vectors of a mixture of Normal distributions which is trained using an EM algorithm during a second training period (Bishop, 1995);

- Bayesian systems are frequently trained in two stages. In the first stage, point estimates of a system's parameters are obtained using a maximum likelihood estimation algorithm such as the EM algorithm. These point estimates are then used to initialize a Markov chain Monte Carlo (MCMC) sampler such as a Gibbs sampler. The initialization of MCMC samplers in this manner can lead to dramatic speed-ups in their convergence (Peng, Jacobs, and Tanner, 1996);
- Jordan (1994) proposed that probabilistic decision trees (also known as hierarchical mixtures-of-experts; Jordan and Jacobs, 1994) be trained in two stages. In the first stage, a deterministic decision tree is trained via, for example, the CART or C4.5 algorithms. The resulting tree is then used to initialize both the shape and the parameters of a probabilistic decision tree which is trained via the EM algorithm in a second training stage;
- Neural networks are sometimes trained in two stages. During stage one, a genetic algorithm is used to identify a good network structure and a good set of initial weights. The backpropagation algorithm is then used to modify these initial weights in a second stage of training (Belew, McInerney, and Schraudolph, 1991).

These examples highlight the fact that machine learning researchers are exploring multi-staged strategies for biasing their learning systems so as to enhance their performances. We believe that a developmental approach based on biological principles as presented here represents a promising, but under-studied, method for suitably biasing learning systems.

References

- Adelson, E.H. and Bergen, J.R. (1985) Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, 2, 284-299.
- Ballard, D.H. (1986) Cortical connections and parallel processing: Structure and function. *Behavioral and Brain Sciences*, 9, 67-120.
- Belew, R.K., McInerney, J., and Schraudolph, N.N. (1991) Evolving networks: Using the genetic algorithm with connectionist learning. *Proceedings of the Second Artificial Life Conference*. New York: Addison-Wesley.
- Bishop, C.M. (1995) *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press.
- Dominguez, M. and Jacobs, R.A. (2002a) Does visual development aid visual learning? In P. Quinlan (Ed.), *Connectionist Models of Development*. East Sussex, UK: Psychology Press.
- Dominguez, M. and Jacobs, R.A. (2002b) Developmental constraints aid the acquisition of binocular disparity sensitivities. *Neural Computation*, in press.
- Geman, S., Bienenstock, E., and Doursat, R. (1995) Neural networks and the bias/variance dilemma. *Neural Computation*, 4, 1-58.
- Gray, M.S., Pouget, A., Zemel, R.S., Nowlan, S.J., and Sejnowski, T.J. (1998) Reliable disparity estimation through selective integration. *Visual Neuroscience*, 15, 511-528.
- Heeger, D.J. (1992) Normalization of cell responses in cat striate cortex. *Visual Neuroscience*, 9, 181-197.

- Hinton, G.E. (1981) Shape representation in parallel systems. In A. Drina (Ed.), *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*.
- Jordan, M.I. (1994) A statistical approach to decision tree modeling. In M. Warmuth (Ed.), *Proceedings of the Seventh Annual ACM Conference on Learning Theory*. New York: ACM Press.
- Jordan, M.I. and Jacobs, R.A. (1994) Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6, 181-214.
- Milner, P.M. (1974) A model for visual shape recognition. *Psychological Review*, 81, 521-535.
- Newport, E.L. (1990) Maturational constraints on language learning. *Cognitive Science*, 14, 11-28.
- Norcia, A. and Tyler, C. (1985) Spatial frequency sweep VEP: Visual acuity during the first year of life. *Vision Research*, 25, 1399-1408.
- Nowlan, S.J. and Sejnowski, T.J. (1994) Filter selection model for motion segmentation and velocity integration. *Journal of the Optical Society of America A*, 11, 3177-3200.
- Ohzawa, I., DeAngelis, G.C., and Freeman, R.D. (1990) Stereoscopic depth discrimination in the visual cortex: Neurons ideally suited as disparity detectors. *Science*, 249, 1037-1041.
- Peng, F., Jacobs, R.A., and Tanner, M.A. (1996) Bayesian inference in mixtures-of-experts and hierarchical mixtures-of-experts models with an application to speech recognition. *Journal of the American Statistical Association*, 91, 953-960.
- Perrone, J.A. and Thiele, A. (2001) Speed skills: Measuring the visual speed analyzing properties of primate MT neurons. *Nature Neuroscience*, 4, 526-532.

- Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P. (1992) *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge, UK: Cambridge University Press.
- Simoncelli, E.P. and Heeger, D.J. (2001) Representing retinal image speed in visual cortex. *Nature Neuroscience*, 4, 461-462.
- Turkewitz, G. and Kenney, P.A. (1982) Limitations on input as a basis for neural organization and perceptual development: A preliminary statement. *Developmental Psychobiology*, 15, 357-368.
- Weiss, Y. and Adelson, E.H. (1998) Slow and smooth: A Bayesian theory for the combination of local motion signals in human vision. Center for Biological and Computational Learning Paper Number 158, Massachusetts Institute of Technology, Cambridge, MA.