A VISUALLY DRIVEN HIPPOCAMPAL PLACE CELL MODEL

Mark C. Fuhs[1], A. David Redish,[2] and David S. Touretzky[1]

[1]Computer Science Dept. and Center for the Neural Basis of Cognition
Carnegie Mellon University
Pittsburgh, PA 15213
[2]Neural Systems, Memory and Aging
Life Science North Bldg, P.O. Box 24-5115
University of Arizona
Tucson, AZ 85724

INTRODUCTION

The firing of place cells in the rodent hippocampus is partly under the control of visual landmarks in the environment (O'Keefe and Conway, 1978). Most place cell models incorporating "visual" input assume noise-free bearing and/or distance information from idealized point objects (e.g., Burgess, Recce, and O'Keefe, 1994; Touretzky and Redish, 1996), rather than attempting to extract landmark information from real-world scenes. This leaves open the question of what kind of visual information is necessary for the hippocampal system to maintain place fields that are stable across trials yet sensitive to landmark position.

In this paper we first describe an algorithm that operates on real images taken from various viewing locations and returns "blob" descriptions: regions of roughly uniform intensity having a rectangular or ovoid shape. We then construct simulated place cells using radial basis functions tuned to blob parameters, and train them by competitive learning to develop realistic place fields. The result is a model that takes real-world scenes as input and produces a distributed activity pattern over a set of place cells as output, from which the current viewing location can be estimated with good accuracy. The implication of this work is that the visual pathway to the rodent hippocampus could involve a relatively simple representation of the local view; rodents may not require object recognition to navigate visually.

VISUAL PROCESSING

Collecting Images of a Scene

We collected twenty-five greyscale images of a computer laboratory using a digital camera atop a mobile robot. The pictures were taken from multiple viewpoints on one side of the room. The twenty-five viewpoints were organized as a 5x5 grid measuring 102 cm on a side.

Characterizing Images Using Blobs

Growing Blobs. Blobs are detected in images using a region growing algorithm. We define a blob B as a set of pixels which must satisfy a set of conditions. We define the blob's mean intensity I_B and mean intensity variation ΔI_B as:

$$I_B = \frac{1}{|B|} \sum_{(x,y)\in B} I(x,y) \tag{1}$$

$$\Delta I_B = \frac{1}{|B|} \sum_{(x,y)\in B} \|\nabla I(x,y)\| \tag{2}$$

Initially, thousands of pixels are randomly chosen from each image, each of which is the starting pixel for a blob. Neighboring pixels to each blob are then added if their intensity is "reasonably" close to the blob's mean intensity; specifically, the difference $|I(x,y) - I_B|$ must be less than a thresholding function T.

The thresholding function T must be defined carefully to allow for differences in the sharpness of object boundaries, and for the effects of non-uniform lighting in the scene. We start by requiring that all pixels simply be within a constant τ of the mean, in which case $T(\tau, ...) = \tau$. We can refine this constraint by scaling T based on local curvature, so that when there is a sharp edge nearby, the threshold is reduced, tightening the criterion for membership. We approximate the local curvature with a five-point discrete Laplacian:

$$\nabla^2 I(x,y) \approx \frac{1}{4}\left[I(x-1,y) + I(x,y-1) + I(x+1,y) + I(x,y+1)\right] - I(x,y) \tag{3}$$

We modulate T by an inverse sigmoidal function s of the Laplacian, so that for regions of high curvature the sigmoid is nearly zero. μ is the mean curvature value for which s should be 0.5, and γ is a gain factor:

$$\sigma\left(\nabla^2 I\left(x,y\right), \mu, \gamma\right) = \left(1 + \exp\frac{\nabla^2 I\left(x,y\right) - \mu}{\gamma}\right)^{-1} \tag{4}$$

Substituting in the experimentally determined values used in our simulations for μ and γ, the threshold equation becomes:

$$T(\tau, \nabla^2 I\left(x,y\right), ...) = \tau \cdot \sigma\left(\nabla^2 I\left(x,y\right), 15, 0.5\right) \tag{5}$$

Finally, to handle contiguous surfaces with non-uniform lighting we increase T based on the mean intensity variation ΔI_B. This allows individual blobs to grow over shallow intensity gradients in images, typically due to smooth surfaces that are lit at an angle, over which the threshold would otherwise be too small for blobs to be able to grow. Including this final constraint yields the following threshold equation:

$$T(\tau, \nabla^2 I\left(x,y\right), \Delta I_B) = (\tau + \Delta I_B) \cdot \sigma\left(\nabla^2 I\left(x,y\right), 15, 0.5\right) \tag{6}$$

In addition to pixels being added to blobs, a pixel that is a member of a blob B_1 may also switch to a neighboring blob B_2 if it satisfies the constraints for pixel addition to B_2 and its intensity more closely matches that of B_2 than B_1.

During the region growing algorithm, the vast majority of pixels were acquired by some blob within the first several iterations. The majority of iterations were therefore dedicated to switching pixels among blobs.

Merging Blobs. Though the region growing algorithm starts with thousands of blobs in order to insure that the entire image can be represented, the images themselves are comprised of far fewer regions of approximately uniform intensity. Without merging blobs, multiple blobs occupying part of the area of a particular region of uniform intensity would compete, and, with few exceptions, none of the blobs would be able to acquire all of the pixels of a particular region. Therefore, neighboring blobs of similar intensity must be merged.

Unfortunately, there is no single threshold of intensity which clearly distinguishes between neighboring blobs in a single region and neighboring blobs in distinct regions with similar intensity; this threshold depends upon the coarseness of features that we wish to detect. Therefore, the threshold was initially set to a low value (15 shades of gray) but was increased as region growing progressed. By intermittently saving the state of each blob at several points during region growing, we were able to create a database of blobs at varying levels of sensitivity to intensity changes. We then compared overlapping blobs collected at different points during region growing; for each set of overlapping blobs, only the blob obtained at the coarsest threshold whose shape was still highly elliptical or rectangular was retained.

Quantitatively Describing Blobs

Describing a blob in terms of the pixels that comprise it is useful for region growing; however, a description of such high dimensionality is far too complex for the purposes of self-localization. Therefore, after blobs were "grown," they went through a series of processing steps which allowed us to describe them using a few simple features. We chose features consistent with the description of an ellipse or rectangle: average intensity, elevation, azimuth, size, eccentricity (the ratio of the lengths of the major and minor axes) and angle of orientation of the major axis.

First, we smoothed the edges and filled in any small holes in the body of the blob, since such high frequency information tended to interfere with categorizing gross features of the blob.

Once blurred, the first moment, or center of mass, of the blob was calculated. The vertical and horizontal positions of the blob center in the image correspond to the elevation and azimuth of the blob, respectively.

A medial-axis transform was then performed on each blob. The medial-axis transform calculates the distance from each pixel to the nearest edge of the blob. The set of pixels whose distances are relative maxima – those pixels whose distance to an edge is greater than all of their neighbors – specifies the "skeleton" of a shape. We fit this skeleton to a line, the slope of which indicated the orientation of the blob's major axis.

Knowing the axis of orientation, we calculated the length of the major and minor axes by calculating the average distances of each pixel, D_x and D_y, from the major and minor axes, respectively. $4D_x$ yields a good approximation of the length of the minor axis; $4D_y$ yields a good approximateion of the length of the major axis. We then calculated the size of the blob as the product of the lengths of the major and minor axes.

Several template superellipses, whose shapes vary between an ellipse and a rectangle depending upon their parameterization, were compared to each blob, and the percentage of overlap of the two shapes was calculated. The highest percentage achieved

Results of Blob Finder on a Grid Image

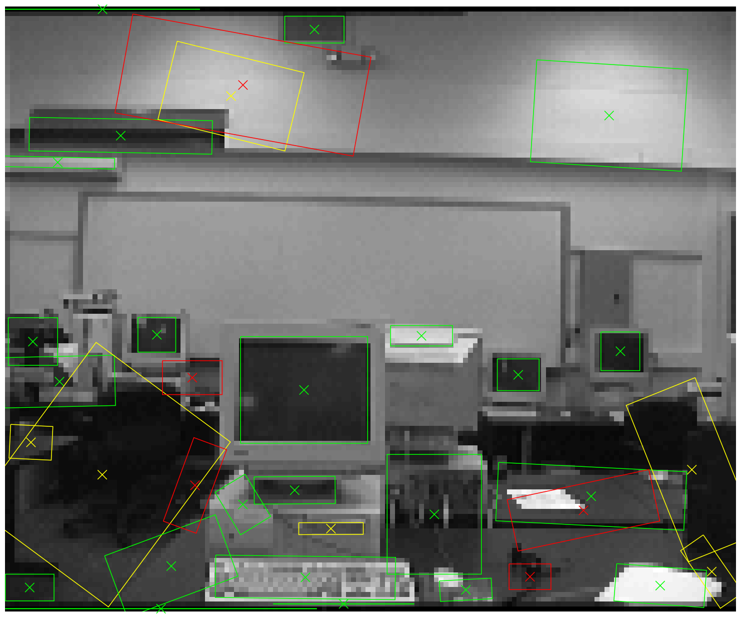

Figure 1. Rectangular blobs extracted from a real-world scene.

among the possible templates was then used to select out only those blobs with greater than 90% overlap. (See Figure 1.) This enforces our criterion that blobs be of approximately elliptical or rectangular shape, which ensures that the features extracted from the blob do, in fact, represent its shape accurately.

Once these features had been calculated, the 5x5 grid of blob features was interpolated into a 17x17 grid. This interpolation provided a more continuous view of the environment than the original 5x5 grid.

TRAINING PLACE CELLS

The Place Cell Model

We modeled place cells as radial basis functions. Each place cell was tuned to two blobs that were each described using four features: average intensity, elevation, azimuth, and size. Each place cell was therefore represented by two sets of feature values, M_1 and M_2; this pair of feature values described the pair of blobs that would maximize the place cell's activation. The similarity of a place cell's ideal feature value μ_f to the corresponding feature value of a blob x_f is measured using a Gaussian function:

$$G(x_f, \mu_f, \sigma_f) = \exp - \left(\frac{x_f - \mu_f}{\sigma_f} \right)^2 \tag{7}$$

When x_f and μ_f are close, this function will be close to 1; as their difference increases, the function approaches zero. The magnitude of σ_f determines the sensitivity of the function to the difference between x_f and μ_f.

The response of a place cell to a particular blob B_i is determined by the product of feature similarities over the set of features F:

$$A_{B_i} = \prod_{f \in F} G(x_f, \mu_f, \sigma_f) \tag{8}$$

All features of B_i must be similar to the place cell's maximal response value in order for the activation to be high; thus, the place cell responds to conjunctions of features.

There is no predetermination of which blobs are to be associated with which place cells. Those blobs B_i and B_j that generated the strongest response to the two sets of place cell feature values M_1 and M_2 determined the activation of the place cell:

$$A = \max_{i \in B} A_{B_i} \cdot \max_{j \in B, j \neq i} A_{B_j} \tag{9}$$

Adapting Place Field Centers

Place cells were initially tuned to random feature values, which were unlikely to correspond to any particular blob in the environment. Therefore, place fields were initially quite broad and weak, conveying little spatial information. In order to increase the spatial information content, a competitive learning algorithm was used to adapt each place cell feature value μ_f to the corresponding blob feature value x_f. For each iteration t of the competitive algorithm, the place cell whose activation was the highest at a particular grid position was determined to be the "winner" of that grid position. For each position in the grid, the pair of blobs that maximized the activation of the winning place cell were used to train the feature values of the winning place cell. Each feature value of the winning place cell was adapted to the correspoonding blob feature value according to the following equation:

$$\mu_f^{(t+1)} = \alpha x_f + (1 - \alpha)\, \mu_f^{(t)} \tag{10}$$

The coefficient α controls the rate at which feature values are adapted; for our simulations the α value of 0.5 was used.

Adapting Place Field Sizes

Overly large place fields lack spatial information content useful for tasks such as navigation; conversely, overly small place fields require too many place cells to represent the entire area of an environment. Therefore, it was important to control the size of the sensitivity parameters σ_f to normalize the sizes of place fields. Unfortunately, there is no single value for σ_f that is appropriate in all conditions, as features such as position and size vary nonlinearly through space.

Therefore, in parallel with the competitive learning algorithm, we adapted the sizes of the place fields by modifying the sensitivity parameters σ_f of the place cells. We defined an "ideal" place field size – in this case, the area around a 5x5 grid position –and used a simple gradient descent technique to adjust the σ_f values to produce the desired place field size.

This algorithm, in conjunction with the aforementioned competitive learning algorithm, was able to quintuple the information content of the place code, using the measure of Skaggs et al. (1993). The total information of the place code was sufficient to distinguish which of the twenty-five images were being presented to the place cells. (See Figure 2.)

DISCUSSION

This work demonstrates that a simple paradigm of feature extraction can provide sufficient information to derive place fields. Classically, one associates landmarks with objects; however, this paradigm makes the task of self-localization quite complex, as

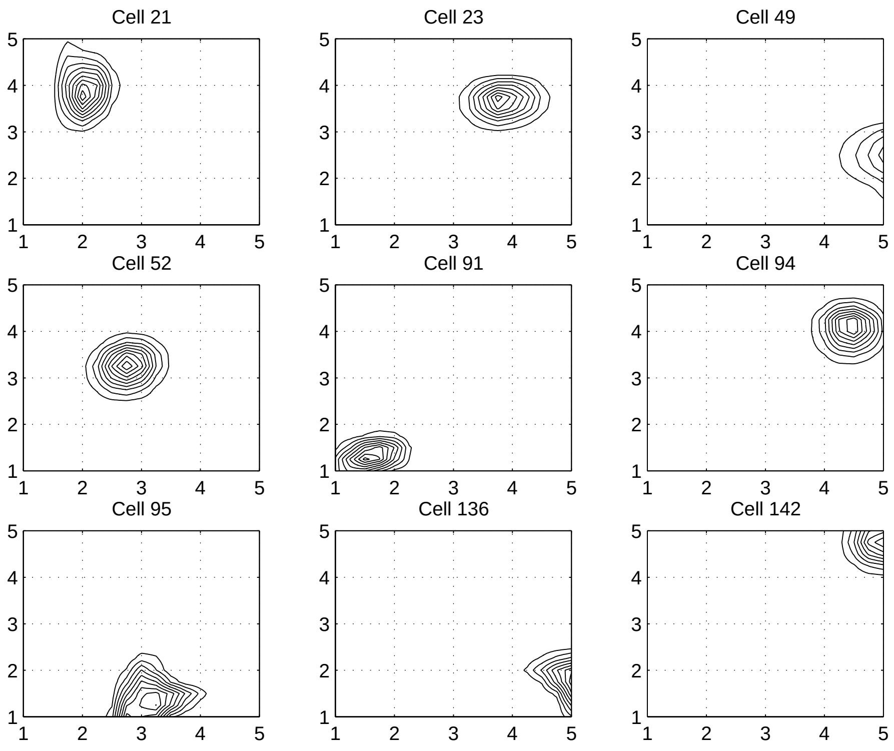

Figure 2. Sample place fields constructed by tuning a product-of-Gaussians function to blob parameters extracted from local views.

it requires that the abstract object be recognized through complex visual processing and that one's location relative to that object be calculated based on one's previous knowledge about the object's shape, size, etc. However, the wealth of information available in a single image, even at low resolution, provides enough information that such complicated cognitive processing is unnecessary.

In the theoretical sense, a landmark is any abstraction that may be characterized by features that possess both position sensitive and position invariant qualities. In order to recognize the same landmark from different viewpoints, there must be some function of the feature space that changes among landmarks, but does not change with different viewpoints of the same landmark. In order to determine one's location relative to the landmark, there must be another function of the feature space that changes as a function of one's relative position to the landmark. So long as these conditions are satisfied by the features extracted from sensory information, any visual processing system would likely be as successful as the system presented here.

Acknowledgments. Supported by National Science Foundation grant IBN-9631336.

REFERENCES

Burgess, N., Reccce, M., and O'Keefe, J., 1994, A model of hippocampal function, *Neural Networks*, **7**(6/7):1065-1081.

O'Keefe, J. and Conway, D. H., 1978, Hippocampal place units in the freely moving rat: Why the fire where they fire. *Experimental Brain Research*, **31**:573-590.

Skaggs, W. E., McNaughton, B. L., Gothard, K. M., and Markus, E. J., 1993, An information-theoretic approach to deciphering the hippocampal code, in: *Advances in Neural Information Processing Systems 5*, pp. 1030-1037, S. J. Hanson, J. D. Cowan, and C. L. Giles, eds., Morgan Kaufmann, San Mateo, CA.

Touretzky, D. S. and Redish, A. D., 1996, Theory of rodent navigation based on interacting representations of space, *Hippocampus*, **6**(3):247-270.