Highlights

3D shape estimation in a constraint optimization neural network

Pallavi Mishra,Sébastien Hélie

- A neural network inspired by visual area V4 can minimize the standard deviation of all angles to estimate 3D shape
- 3D shape estimation by this convolutional neural network agrees with human 3D shape perception
- Other visual constraints can be solved within the same network to test the theory of visual constraints in estimating 3D shapes

3D shape estimation in a constraint optimization neural network

Pallavi Mishra[a,*], Sébastien Hélie[a,**]

[a]Department of Psychological Sciences, Purdue University, 703 3rd Street, West Lafayette, IN 47907

ARTICLE INFO

Keywords:
3D Perception
Deep Neural Networks
V4

ABSTRACT

One of the most important aspects of visual perception is the inference of 3D shape from a 2D retinal image of the outside world. The existence of several valid mapping functions from object to data makes this inverse problem ill-posed and therefore computationally difficult. In human vision, the retinal image is a 2D projection of the 3D world. The visual system imposes certain constraints on the family of solutions in order to uniquely and efficiently solve this inverse problem. This work specifically focused on the minimization of standard deviations of 3D angles (MSDA) for 3D perception. Our goal was to use a Deep Convolutional Neural Network based on biological principles derived from visual area V4 to achieve 3D reconstruction using constrained minimization of MSDA. We conducted an experiment with novel shapes with human subjects to collect data and test the model. The performance of the network largely agreed with how humans estimated novel 3D shapes. The results show that the constraint of MSDA in 3D shape can be implemented in a neural network and produce human-like results. Additional visual constraints can be added to the network in the future to fully test the theory of visual constraints as a basis of 3D shape perception.

1. Introduction

The basis of perceptual reconstruction of 3D objects in the human visual system is a long studied problem. The problem of 3D perception from a projected image in 2D by the early visual system has been formulated as an "inverse problem" (Poggio and Koch, 1985; Tikhonov and Arsenin, 1977; Pizlo, 2001). The existence of several valid mapping functions from object to data makes this inverse problem ill-posed and therefore computationally difficult. In human vision, the retinal image is a 2D projection of the outside 3D world. It has been postulated in Pizlo (2001) that the visual system imposes certain constraints on the family of allowable solutions in order to efficiently solve this inverse problem.

The main motivation behind this work is to design and test a biologically-inspired network-based mechanism to study 3D perception of object shape from their 2D projections. In order to understand how human vision perceives the 3D structure of objects from the 2D retinal images, the use of certain constraints is essential. This is because the inverse formulation of 3D percept is insufficient to solve for a unique shape perception. The visual constraints of standard deviation of 3D angles, symmetry, planarity, and compactness of volume in models of 3D shape recovery are derived mathematically from the principles of the traditional Gestalt approach based on the 'Law of Prägnanz' or simplicity principle (e.g., the principles of closure, good continuation, regularity, symmetry, simplicity and so forth). These constraints are chosen specifically because of their demonstrated effectiveness in generating reliable 3D percepts in models of 3D vision (Li, Pizlo and Steinman, 2009).

In this work, the constraint of minimization of standard deviation of 3D angles (MSDA) was used to solve the inverse problem of 3D shape reconstruction in a deep neural network model. The computation of pairs of angles in estimated 3D shape is a good starting point after which other constraints such as symmetry and compactness of 3D shape can be added into the same network.

In this work, areas of the visual cortex were taken into account in order to build a computational model that is based on biological principles of information processing. Specifically, the computational anatomy of the striate cortex and some functional properties of the visual area V4 were used to build the model. The goal was to demonstrate how a computational approach based on biological principles may perform constraint optimization in a network. The reason for restricting the model to computation in a network was simply because the brain itself is a network. The choice of a Deep Neural Network substrate (DNN) for the computational model is based on the recent discovery of interesting properties of DNNs, showing that these models embed general purpose visual computations while displaying extraordinary task-trained accuracy in visual tasks. For example, Dekel (2017) has shown that trained DNNs exhibit general purpose computations that are computationally similar to biological visual systems. They found that perceptual sensitivity to image changes has mid-computational correlates in DNN, while sensitivity to segmentation, crowding, and shape, have DNN end-computation correlates. It has also been shown (Cadieu, Hong, Yamins, Pinto, Ardila, Solomon, Majaj and DiCarlo, 2014; Khaligh-Razavi and Kriegeskorte, 2014; Yamins, Hong, Cadieu, Solomon, Seibert and DiCarlo, 2014) that when the same images are processed by trained DNN, humans, and monkeys, the final DNN computation stages are strong predictors of human fMRI and monkey electrophysiology data collected from visual areas V4 and IT. This is not to say that DNNs are the only computational tools for studying properties of human vision as different learning algorithms and different physical implemen-

*Corresponding author
**Principal corresponding author
palmishr@gmail.com (P. Mishra); shelie@purdue.edu (S. Hélie)
ORCID(s): 0000-0001-7511-2910 (P. Mishra)

tations may converge to the same computation when sufficiently general problems are solved near-optimally (Dekel, 2017). However, DNNs present a wide array of functional architecture and algorithmic choices that serve as a flexible mechanism to simulate certain visual computations due to their generalization capabilities.

Visual area V4 is a mid-tier visual cortical area in the ventral visual pathway that has been studied for its role in shape perception among other sensory functions such as properties of surface of objects, motion, visual attention, and depth (Roe, Chelazzi, Connor, Conway, Fujita, Gallant, Lu and Vanduffel, 2012). Studies (Mountcastle, Motter, Steinmetz and Sestokas, 1987; Desimone and Schein, 1987) have shown prominent orientation selectivity in this area suggesting its role in shape perception. In order to encode complex 3D shape representations, this area specializes in encoding the relative coordinates of object features such as edges and curvatures (Pasupathy and Connor, 2001). The V4 cells are found to be extremely sensitive to the relative position of contour fragments within objects rather than absolute coordinates of features. This area is critical to the structural shape coding scheme and also carries sufficient information for reconstruction of moderately complex shape boundaries. The proposed computational model used the stimulus object's relative coordinates to compute the edges in the stimulus. The edges were then used to extract properties about the overall shape of the object using matrix-based operations.

In order to inform the architecture of the model with regard to encoding and processing of 2D spatial coordinates and 3D z-coordinates computed from 2D spatial coordinates, the functional and computational architecture of the striate cortex was taken into account. It is known from several studies (Grill-Spector and Malach, 2004; Fischer, Spotswood and Whitney, 2011; Finlayson, Zhang and Golomb, 2017) that 2D spatial location information is encoded in several visual areas but it's magnitude or sensitivity decreases along the visual hierarchy. However, 3D perceived position in depth can be tracked inversely to 2D spatial position in the sense that magnitude of depth decoding gradually increases from intermediate to higher visual hierarchy. As one goes up the visual hierarchy, visual areas become increasingly tolerant to changes in the 2D location coordinates and become increasingly more sensitive to depth information. Finlayson et al. (2017) have explored the nature of spatial position-in-depth representations and the interactions of the three spatial dimensions. They presented various stimuli spatially in horizontal (X), vertical (Y) and depth (Z) coordinates to explore how 2D and depth information may be organized and how they interact throughout the visual cortex. As per their findings, there was a gradual increase in Z information encoding in later visual areas and Z dimension information was found to highly overlap with XY information in later areas. Such findings confirm that depth information is gradually computed and stored with 2D information as one goes up the visual hierarchy. It makes sense for the model to take the 2D coordinates as inputs and compute depth information in stages across successive layers in the network.

Another important consideration for the network model is the type of computational layers that can best approximate the computation of the depth dimension from lower dimensional inputs (including 2D coordinates) in the visual hierarchy. It was postulated in Schwartz (1980) that one way to encode high-dimensional features such as depth using low-dimensional components such as the 2D spatial coordinates of a scene can be demonstrated by the functional architecture of the striate cortex. The columnar structures in striate cortex can allow for encoding of more complex dimensions such as depth and color using spatial difference-based mappings computed over lower dimensional columnar structures (an algorithm for a possible mapping was also presented in Schwartz (1980)). These types of mapping algorithms present a way in which the computational architecture of striate cortex may allow for multiple different dimensions to be multiplexed using something like a spatial frequency channel for each dimension. Several computational models have since been proposed for encoding schemes and differential mapping algorithms to accomplish such tasks (an extensive review is presented in Fischer (2014)).

Computationally, convolutional layers in DNN provide enough flexibility to create a mapping from lower layer to layers up in the hierarchy and apply filters to carry out computations necessary to extract the visual constraint of MSDA. These layers allow the model to successively compute a differential mapping of the previous layers to extract higher order properties of the stimuli for the next layer. To incorporate the biological principles discussed above, the proposed computational model used the relative coordinates of vertices in the stimulus object to compute every viable edge in the object. These edges are relative to the object coordinates. The edges are then used to extract properties about the overall shape of the object using matrix-based operations.

The model takes as input the 2D coordinates which are mapped to the X and Y axes in the proposed simulation system. The Z dimension information is estimated and refined as a result of convolution operations in each successive hidden layer. The lower layers in the network first randomly guess a Z coordinate and this value is further optimized as the network tries to minimize the standard deviation of all 3D angles in the output layer. Therefore, the layers closer to the output layer have a more accurate estimate of the Z dimension than the first layer in the network. The network stores each of the dimensions in parallel and identical computational structures called 'channels'. These channels are traditionally used for RBG encoding in image processing applications for DNNs. So the x, y, and z dimensions reside in separate channels. Each successive layer computes differences between two given dimensions at a time to compute edges and then 3D angles.

To assess the model's performance, an experiment based on ideas from classic psychophysics is used to measure shape constancy. The experiment measured the consistency in the perception of novel stimuli achieved by human subjects on a set of given shapes. A good test of the model was to be an accurate predictor of the outcome of the psychophysics

experiment based on a metric we defined for measuring the shape consistency of 3D reconstructions of the shapes by the model. The model's prediction was compared against the results from this experiment.

2. Methods

Code and supplementary material are available online at: `https://github.com/palmishr/3dvisnet`.

2.1. Model

A DNN (Deep Neural Network)-based model using the Pytorch programming framework was developed based on the ideas discussed above. The model attempted to use the MSDA constraint to estimate the missing depth parameter. The model did not explicitly attempt a full 3D reconstruction of the image but instead estimated the angles for the most plausible 3D structure. The input for the model was a 2D canvas wherein coordinates of visible vertices were presented to the model. The model then computed the 3D angles from these vertices by learning to estimate the depth parameter that minimized the standard deviation of all angles. The model can process a batch of such objects simultaneously with variable number objects in the batch. There is no programmatic limit to the number of objects in a batch. All the input stimuli to the model were created programmatically in the fixed coordinate system, so the 2D coordinate system was consistent across all stimuli. For each stimulus, the following information was extracted: a) (x,y) coordinates of the vertices and b) a list of edges between each pair of vertices.

The coordinate system used to generate stimuli for experimentation and present inputs to the DNN model were the same and consistent. The coordinate system used to display images on the screen during the experiment was different as it was determined using a different programming framework. However, since images exported out of the stimulus generation environment were already processed with regards to rotation, one can assume that the coordinate systems were consistent throughout for all practical purposes.

The model can be configured to process a fixed maximum number of vertices at any given time. This is a limitation imposed by memory constraints in the simulation environment. When the model is presented a stimulus input with fewer than the maximum number of possible vertices, padding is used to fill up the unused matrix cells. This operation allows the model to process a variable number of vertices per input object, even within a single batch of input.

The process of padding unused cells in the computation is straightforward for the convolution operation. However, the edge connections vector in the input needs to be re-structured to comply with the higher dimension of vertices. The model only processes information visible in the 2D projected view of the input stimulus. This implies that the input parameters include only the vertices visible in that projected view of the object. The stimulus generation process takes care of this requirement while generating input files for a given stimulus. The connection matrix only includes vertices visible in the current view of the object. The model estimates

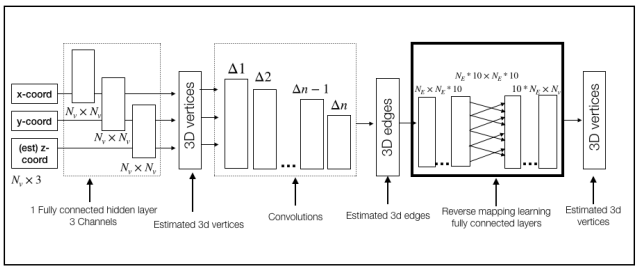

Figure 1: Complete network model to additionally compute missing z parameters using fully connected layers. The reverse mapping layers are highlighted using a darker box frame within the Figure.

the depth parameter for the object by making an assumption about the missing or hidden vertices. For all vertices that are not complete, that is, all three edges are not visible, it is assumed that the number of hidden vertices is equal to the number of incomplete vertices. This assumption is based on the work of Cao, Liu and Tang (2008), where psychophysical constraints were used to extract hidden structure from a partially visible object. The network computes the standard deviation value of all 3D angles for each training object and minimizes this value to learn the best z parameters for given objects. The shape information has to be extracted from the network separately since the model does not directly output the missing z-parameters for all vertices of the given object. The model only outputs SDA measures related to the cost function of how closely the constraints are met by the current z-coordinate estimation.

In order to retrieve z-parameter values from the model, a reverse learning technique is implemented where a set of fully connected layers learn to extract estimated z* information from the layers computing the final estimate of SDA. As shown in Figure 1, a series of hidden layers are added to learn the backward mapping from edges to vertices. The layers encoding the edges have the optimal SDA for the given stimuli. The reverse mapping layers are trained using the actual X and Y values of the 2D vertices from edges values obtained after convolution operation. The architecture of full model using this technique is shown in the same figure (Figure 1).

Extracting the value of the z-parameter from the network amounts to learning a reverse-mapping operation from edges that minimize the SDA constraint back to their corresponding z-coordinate configuration. This means that as the network is trained to minimize the SDA constraint, it needs to simultaneously learn the reverse mapping for each of the training examples. Since our training is based on the stochastic gradient descent method, the model has two parallel and nested learning paths for each set of training examples in each epoch as shown in Figure 2.

After training, the process of reconstructing a novel 3D shape for the network involves searching in the parameter space that the model learned during training. It should be

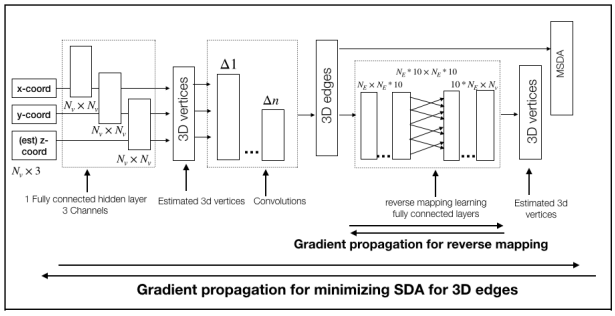

Figure 2: The model included two separate learning mechanisms. Here, MSDA represents the part of the network that computed the Gram matrix and minimized the standard deviation of the matrix.

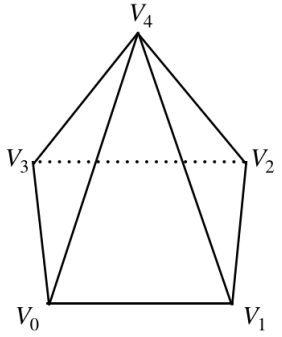

Figure 3: A sample stimulus with 5 vertices.

noted that the reconstruction output (i.e., the value of the missing z-dimension) may change based on how long the model has been trained. Depending on the size of the network (e.g., based on the number of hidden layers), the time it takes to retrieve the z-parameter can vary. For example, a network that can process up to 20 vertices with 4 hidden layers takes a fraction of a second to output the z-parameter while running on a GPU with 1920 cores and 8 GB memory. However, for a larger network with more hidden layers, this time may increase by several orders of magnitude to seconds depending on the computational resources for the network on a particular machine.

2.2. Model demonstration using an example stimulus

The model receives an input containing the 2D coordinates of the visible vertices along with the connection matrix. The connection matrix represents the pairs of edges visibly connected in the 2D object view. It has to estimate an initial depth z_i^* , $i \in 0 \ldots (N_v - 1)$ for each of the N_v vertices. It is to be noted that the edge vectors can be obtained using x, y, z coordinates of the vertices $V_0 \ldots V_{N_v-1}$ by taking the difference in coordinates as shown in Equation 1.

Name	x	y
V_0	x_0	y_0
V_1	x_1	y_1
V_2	x_2	y_2
V_3	x_3	y_3
V_4	x_4	y_4

Edge	Value
$E_{0,1}$	1
$E_{1,2}$	1
$E_{2,3}$	1
$E_{3,4}$	1
$E_{0,2}$*	0
$E_{1,3}$*	0
$E_{2,4}$	1
$E_{0,3}$	1
$E_{1,4}$	1
$E_{0,4}$	1

From	To
V_0	V_1
V_1	V_2
V_2	V_3
V_3	V_4
V_0	V_2
V_1	V_3
V_2	V_4
V_0	V_3
V_1	V_4
V_0	V_4

Figure 4: The sample stimulus (Figure 3) encoded into model inputs. The connection matrix is a list of vectors encoding visible vertices and visible edges.

$$E_{i,j} = V_j - V_i \tag{1}$$

Since the edge computations in each of the three dimensions are identical operations, the model can work on the three dimensions in parallel as shown in Equation 2. Here, z^* represents the estimated z coordinates for vertices V_i and V_j.

$$E_{x_{i,j}} = x_j - x_i, \quad E_{y_{i,j}} = y_j - y_i, \quad E_{z^*_{i,j}} = z^*_j - z^*_i \tag{2}$$

Figure 3 shows a simple stimulus for illustration. The vertices in the sample stimulus and their coordinates in three dimensions are shown in Figure 4. Only the X and Y dimension is input into the model. The edge vector shown in Figure 4 encodes the information about visible connections in the stimulus. The edges that visibly exist correspond to the value 1 and the ones that do not visibly exist have the value 0. Figure 4 also shows the connection between the vertices that each edge represents. This relationship was established in input formatting Algorithm 1. Figure 5 illustrates how the computation is distributed in three separate and identical channels as the input is processed in the model.

The edges are computed using a series of convolutional layers with differing dilation values as illustrated in Figure 6. As depicted in the illustration, the first convolution operation computes the edge between vertices that are adjacent,

	Edge	From	To	x	y	z
Convolutional Layer 1	$E_{0,1}$	V_0	V_1	$x_1 - x_0$	$y_1 - y_0$	$z_1 - z_0$
	$E_{1,2}$	V_1	V_2	$x_2 - x_1$	$y_2 - y_1$	$z_2 - z_1$
	$E_{2,3}$	V_2	V_3	$x_3 - x_2$	$y_3 - y_2$	$z_3 - z_2$
	$E_{3,4}$	V_3	V_4	$x_4 - x_3$	$y_4 - y_3$	$z_4 - z_3$
Convolutional Layer 2	$E_{0,2}$	V_0	V_2	$x_2 - x_0$	$y_2 - y_0$	$z_2 - z_0$
	$E_{1,3}$	V_1	V_3	$x_3 - x_1$	$y_3 - y_1$	$z_3 - z_1$
	$E_{2,4}$	V_2	V_4	$x_4 - x_2$	$y_4 - y_2$	$z_4 - z_2$
Convolutional Layer 3	$E_{0,3}$	V_0	V_3	$x_3 - x_0$	$y_3 - y_0$	$z_3 - z_0$
	$E_{1,4}$	V_1	V_4	$x_4 - x_1$	$y_4 - y_1$	$z_4 - z_1$
Convolutional Layer 4	$E_{0,4}$	V_0	V_4	$x_4 - x_0$	$y_4 - y_0$	$z_4 - z_0$

Channel 1 Channel 2 Channel 3

Figure 5: Identically divided computations across three separate channels of the network using the sample stimulus shown in Figure 3

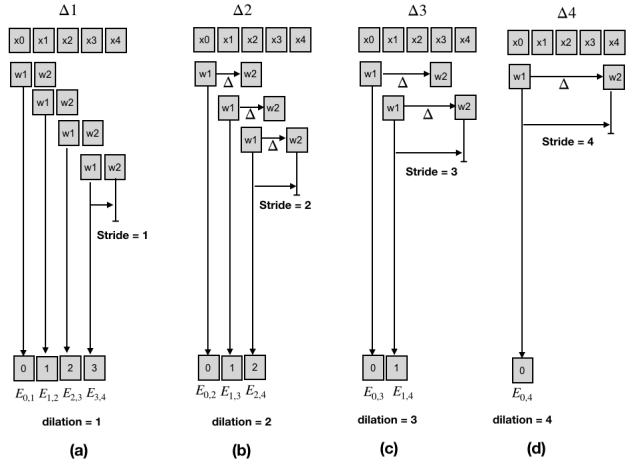

Figure 6: Edge computation operations in the convolutional layers for a particular channel (X in this case) (a) Dilation of 1 (b) Dilation of 2 (c) Dilation of 3 (d) Dilation of 4. The same operation is repeated for the Y and Z dimensions on the second and third channel respectively.

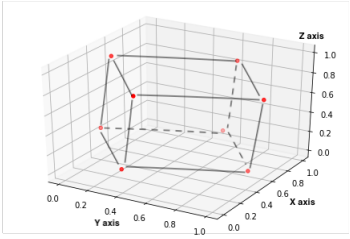

Figure 7: 3D voxel output of the vertices extracted from the network for a simple symmetrical cuboid object.

the second convolution computes edges that have 1 vertex between them, and the last convolution computes edges for vertices that are the furthest apart. All possible combinations of vertices are covered by this process. This process computes edges in the same order as they are computed in Algorithm 1. After all edges are computed by the series of convolution layers, the connection matrix denoting vertices connected by an edge is used to drop out the edges that do not exist in the object. Given our sample input stimulus in Figure 3 and the corresponding edge vector table in Figure 4, the edges that do not exist in the stimulus have been marked with a * in the edge vector computed by the network in Figure 5. These edges will be dropped from all further computations. All the edges are then normalized to facilitate keeping track of the computations and intermediate results generated. Further details regarding the specific values used to configure the convolutional layers along with details of the mathematical operations involved in a typical convolution are presented in Appendix Section A.2.

In the next step, the model forms a Gram[1] matrix for the edge vectors. This is done by taking the outer product of the entire set of computed edges with itself. Each cell in this matrix corresponds to a combination of any two edges.

The angle between a pair of edges is defined as:

$$\theta_{i,j} = cos^{-1}\frac{E_i^T E_j}{||E_i||_2||E_j||_2} = cos^{-1}(G_{i,j}/(l_i l_j)) = cos^{-1}G_{i,j}$$

$$given : l_i = l_j = 1 (normalized\ edge\ vectors)$$

[1]Given a set V of m vectors, the Gram matrix G is the matrix of all possible inner products of V

where $G_{i,j}$ is the gram matrix cell for normalized edges E_i, E_j. Since cos^{-1} is a monotonic function, minimizing the angle between the edges E_i, E_j corresponds to minimizing $G_{i,j}$. Minimizing SDA amounts to minimizing the variance of the Gram matrix itself.

2.2.1. Sample Output and Training Data

Figure 7 shows the output of the estimated z parameters by the model for an input with a symmetrical cuboid shape. The model was trained on a set of random cuboid objects created using a symmetrical cuboid shape (as shown in Figure 7). These training examples were generated by transforming the original shape by adding random displacement to the vertices of the original shape in all 3 axes. The original shape was also randomly re-scaled in a different size and orientation to create training data for the model.

2.3. Experiment

A shape constancy experiment was designed to estimate consistent shape perception from a group of human subjects in order to: (a) Isolate and identify cases where human sub-

jects can perceive the shapes of novel stimuli consistently. (b) Isolate and identify cases where our model succeeds in achieving a consistent 3D shape estimation as measured by angular estimation using the same stimuli with different rotations. (c) Compare the outcomes of (a) and (b) in order to test the effectiveness of the model.

2.3.1. Stimulus

Stimulus generation was a critical step for designing the experiment as well as for testing the model. The shape perception experiment required subjects to consistently identify objects presented from more than one viewing angle. For a reliable test of consistent 3D perception from different viewing angles, it was necessary that subjects used no previous knowledge about the shape but only the information presented to them in the experiment. Therefore, a set of novel and unfamiliar stimuli was constructed for the purpose of testing reliable shape perception in the experiment.

It has been hypothesized in Chan, Stevenson, Li and Pizlo (2006) that 3D perceptual representation is reliable in cases of structured 3D objects but not in cases of unstructured objects. Pizlo and Stevenson (1999) showed that shape constancy from novel views can only be achieved if structured novel objects obey some regularity constraints (such as symmetry). Therefore all novel shapes were constructed so that they had a pronounced regular structure for unique shape perception. These stimulus objects displayed mirror symmetry along one axis only. The entire set of these objects is presented in the Supplement Section 1. The selection of these specific shapes was based on the results from several iterations of pilot versions of the experiment. It was observed that without any regularity in the stimuli, there was no consistent shape recovery as measured by our pilot experiments. Some examples objects from the pilot experiment that failed to be recovered above chance level are shown in Supplement Section 2. It was observed from the pilot experiments that objects with fewer vertices were more difficult to recover.

Based on these findings, sufficiently complex but regular and novel sets of shapes were created. The algorithm to generate these new shapes (as documented in Appendix Section A.3) was able to create a limited number of unique shapes that were clearly distinct from one another. Several other shapes created using this algorithm were too similar to other shapes in the set (except for only a slight difference). The set of distinct shapes was then divided into blocks based on the level of complexity of the shape for the final version of the experiment. The motivation in dividing the shapes in blocks was to group shapes of similar complexity (same number of vertices) together to reduce the variance in measuring shape constancy by aggregating individual shapes in blocks.

Each block was designed so that it contained objects with similar complexity. The numbers associated with blocks had no meaning. Two of these blocks contained three objects and four blocks contained two objects. There are different numbers of objects in blocks to rule out the possibility that subjects could only discriminate between an object and itself but did not perceive the object uniquely. This hypothesis can be tested if there were any significant differences in the performance of blocks with two objects versus blocks with three objects.

Another block was created that contained objects with dissimilar shapes from other blocks. This block was used to test whether the subjects are only discriminating between objects or were perceiving them individually. If they were only discriminating between objects, then this block would have a higher performance because of higher discriminability between the objects compared to other blocks.

The set of stimuli used for training and testing the model and the set of stimuli used to test human subjects were the same. This requirement was imposed in order to make a direct comparison between the performance of the model and the experiment outcome. An open source 3D graphic rendering tool called 'Blender' was used to create 3D models of the novel structured objects. Since this software allows for Python-based programmatic creation, manipulation, and extraction of data, the object parameters could be extracted in the form of a text file along with images from a variety of rotation viewpoints and projections. The stimulus parameters exported from the software were used as input to the model and the corresponding images were used for the experiment.

To create the stimuli, the 3D graphical modeling software was set to perspective projection which is its default setting. The camera operator in the software (used to generate object views) was positioned at the origin relative to the object coordinates. The camera position was constant and therefore it was equally distant for all the generated stimuli. The average angular size of the objects created in the simulator was 5.3 degrees. The distance of the camera from the object was fixed at 11 units. The average length of an edge in each object was 1 unit.

2.3.2. Experiment Details

Stimuli: All stimuli were symmetrical on the X axis with a pronounced structure for shape perception so that subjects can achieve shape constancy for these unfamiliar but structured objects. There were four rotations per stimulus on the Y and Z axes each (for a total of eight rotated versions per stimulus). Each object was shown a total of sixteen times - eight times against its own rotated version and eight times with another object's rotated version. Depending on the shape of the original object, a rotation on the Y or Z axis may change the perception of the object shape to a certain extent. It should be noted that the rotation operation is relative to the stimulus and not in regard to the absolute coordinate system used to display stimuli on the monitor. Therefore any rotation on the Y and Z axis has an effect on the experiment parameters.

The image length was set to 11 cm and the image width was set to 20 cm for the experiment. The average viewing distance was 80 cm. The length and width of the image was fixed. However, based on the rotation angle and differences in lengths across the diagonal, the length and width of shapes could vary. The range of this variation did not exceed 1.5 cm

for the width and height of each object. The average angular size for object width was 4.3 degrees and for height it was 4.9 degrees. The range of angular width and height size did not exceed 1.07 in either dimension.

Number of subjects: Twenty-five subjects, all students at Purdue University, were recruited for the experiment. All subjects were students in the Department of Psychological Sciences. The study protocol was approved by the Purdue University Human Research Protection Program. Written informed consent was obtained from each subject before beginning the experiment. Twenty subjects participated in the experiment for course credit while the remainder were volunteers. The experiment lasted for about an hour.

Number of trials per subject: The total number of trials for each subject in the experiment was 272. The trials were distributed across seven experimental blocks. Each stimulus object in an experiment block was presented to the subject from eight different projection angles.

Design of Experiment Blocks: There were seven blocks in the experiment. Each block was designed so that it contained similarly shaped objects with similar complexity. The numbers associated with blocks had no ordinal meaning. Blocks 1, 2 and 7 contained three objects while the rest contained two objects. All blocks except Block 7 contained similar but distinct shapes. Block 7 contained objects from other blocks (2, 3 and 4). Since Block 7 had objects with dissimilar shapes, it was used to test whether subjects only discriminated between objects instead of perceiving them individually. In the former case, the performance in this block should be better than all the other blocks.

Task: Within each block, each object (A) was shown either paired with itself (A) at a different rotation angle or with another object (B) with a different rotation angle. The subject was to decide if the two objects were the same or different by answering a 'YES/NO' question at the end of the display. The 'YES' response was mapped to the 'f' key and 'NO' response was mapped to 'j' key on the keyboard. There was no feedback given to the subjects on their responses. The sequence of display was:

1. Blank Screen (1 sec)
2. Object (A) (4 sec)
3. Blank Screen (1 sec)
4. Object (A) or Object (B) (4 sec)
5. Are the objects shown same? YES/NO

Based on the outcome of pilot studies and the design of shape constancy experiment, it was predicted that blocks with higher object complexity would outperform blocks with lower complexity. Also, since the experiment tests subjects' ability to perceive a shape consistently and individually, it was also predicted that performance in blocks with dissimilar objects would not differ from from performance in other blocks. The other prediction was that performance in blocks with two objects and blocks with three objects should not differ.

The predicted block performance based on object complexity are as follows: Block 4 has the least complex objects so the lowest performance is expected for this block. Block 3 has more complexity than block 4 but lower than the rest of the blocks. Performance in this block should thus be better than Block 4 but not as good as other blocks. Blocks 1 and 2 have high complexity objects and similar shapes so their performance is not expected to differ and should be higher than blocks 3 and 4. Blocks 5 and 6 have the highest complexity so their performance should be higher than blocks 1 and 2. Block 7 was exploratory. It should have the highest performance if this task is being performed as a discrimination task rather than a consistency of perception task. Since this block contains various objects complexity, we could not predict how its performance would compare to the rest of the blocks based on the principle of object complexity.

In general, if the model and the experiment outcomes agree on the shapes that are more consistently perceived than others, then the model achieved the objective of using the constraint of MSDA to recover shapes. But since the model outputs an estimate of z-coordinates of the 2D shapes for each given rotation of the object shape and no information about exact object reconstruction can be extracted from the experiment, we proposed a metric to measure object perception consistency from the model. If the model is indeed applying MSDA to consistently perceive the stimulus, then lower variance in model output would corresponds to higher block accuracy in the human experiment. Because the measuring scales for the human and model are different, agreement of the performance order of the blocks from the experiment and model was used to measure success.

In other words, the experiment is based on the classic psychophysics experiment structure to measure shape constancy achieved by human subjects on a set of given shapes. A good test of the model is to be an accurate predictor of the block ordering outcome of the experiment based on a metric we defined for measuring the shape consistency of 3D reconstructions of the shapes by the model.

3. Results

3.1. Experiment Results

For each subject, whether or not a given stimulus is correctly categorized was recorded. If the two stimuli shown back to back were the same object and the participant responded 'yes' then a correct response was recorded. If the two stimuli were different objects and the subject responded 'no', a correct response was also recorded. In all other cases, an incorrect response was recorded.

The first step in data analysis was to identify the subjects who were able to perform the task above chance level. This chance level cutoff was computed based on the number of successes in 272 independent trials with the probability of success equal to 50% per trial with a confidence interval of 95% (using a binomial distribution). The accuracy cutoff was found to be 55%. The response accuracy for a given object was obtained by counting all the correct responses against the total number of times the object was shown to the subject. The overall performance of a subject in the ex-

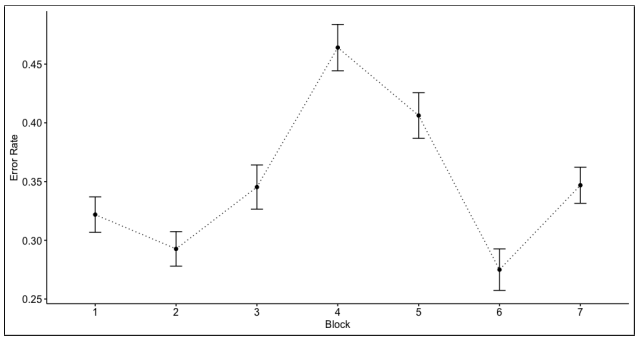

Figure 8: Average error rates for each block in the human experiment.

periment was their accuracy on all the objects combined. In previous pilot studies, it was noted that engaged subjects could perform considerably above chance (up to 80% accuracy overall). Out of a total twenty-five subjects, five were removed from further analysis based on this cutoff.

A suitable method to compare the experiment outcome against the model was to compare performances at the block level. The model's output across several iterations could be aggregated at the block level. In this way, the individual variations for object consistency from the experiment and the variation of the model's output across different simulations were both aggregated at the same level. Since blocks contained similar objects with similar complexity, blockwise comparison was more appropriate than comparing individual stimuli one by one.

The overall error rate per block is shown in Figure 8. In order to test for the effects of block and rotation angle in the Y and Z axes on the binary response outcome (correct or incorrect), a generalized linear model based on maximum likelihood estimation was fit to the data. The generalized linear mixed effects model used a logit link function for the binomially distributed dependent variable.

There were three generalized linear models fit to the data. The first model contained both the blocks and the rotation angles as individual predictors. The other two models were fitted to the data by dropping one of these two predictors at a time. Table 1 summarizes the coefficients, their significance level, and standard errors for the particular blocks and rotation angles from the first model. A test of significance of block and the rotation angle on response accuracy was carried out by comparing the fit of the full model with the fit of the other models without each predictor. The results of the significance test on blocks is presented in Table 2. It was observed that block had a significant effect on the outcome (correct or incorrect) using the deviance statistic ($Chisq(6) = 79.49, p < 0.001$). The effect of rotation on either the Y or Z axis on the outcome was also significant ($Chisq(7) = 50.4, p < 0.001$) as seen in Table 3. A simple linear regression was done on the parameter estimates (beta) for Y and Z versus the angle of rotation. The plots showing the relationship between the betas for Y and Z and the angle of rotation for each directions are shown in Figure 9. As

Table 1
Results from the Generalized Linear Models. Each independent variable is displayed with its coefficient and standard error along with significance.

	Dependent variable:
	Correct
Block1	1.095*** (0.123)
Block2	1.238*** (0.124)
Block3	0.988*** (0.131)
Block4	0.479*** (0.128)
Block5	0.721*** (0.129)
Block6	1.325*** (0.135)
Block7	0.980*** (0.122)
rotY36	−0.163 (0.120)
rotY54	−0.232* (0.121)
rotY72	−0.392*** (0.119)
rotZ18	0.178 (0.117)
rotZ36	−0.042 (0.115)
rotZ54	−0.150 (0.115)
rotZ72	−0.281** (0.113)
Observations	5,440
Log Likelihood	−3,417.652
Akaike Inf. Crit.	6,865.304
Bayesian Inf. Crit.	6,964.327
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 2
ANOVA table for the block significance test obtained by comparing two nested models, one with block as predictor and another without block as predictor of response accuracy.

	Df	AIC	BIC	logLik
	deviance	Chisq	Chi Df	
bmod_no_blocks	9	6932.79	6992.20	-3457.39
	6914.79			
bmod	15	6865.30	6964.33	-3417.65
	6835.30	79.48	6	
Pr(>Chisq)				
4.567e-15***				

can be seen, an increase in the rotation on the Z axis reduced subject accuracy, but no effect of rotation on the Y axis was observed.

The discrimination sensitivity measure (d') related to the performance within each block is shown in Figure 10. Since higher discriminability should lead to higher accuracy in the task, the plot for discrimination sensitivity for blocks should match the one for accuracy. This is indeed the case as the order of blocks based on both these measures is the same.

Due to the nature of the task involving a forced choice (yes/no response), it makes sense to test if the responses were biased in one way or the other. That means, if subjects were more likely to say 'yes' when the stimuli presented were different objects rather than 'no' when they were same objects. The criterion location of 0 means that the responses are unbiased. Criterion location was obtained using the formula:

Table 3
ANOVA table for the rotation angle significance test by comparing two nested models, one with rotation (4 rotations on Y axis and 4 rotations on Z axis) as predictor of response accuracy and another without rotation as predictor of response accuracy.

	Df deviance	AIC Chisq	BIC Chi Df	logLik
bmod_no_rot	8 6885.70	6901.70	6954.51	-3442.85
bmod	15 6835.30	6865.30 50.40	6964.33 7	-3417.65
Pr(>Chisq)				
1.208e-08***				

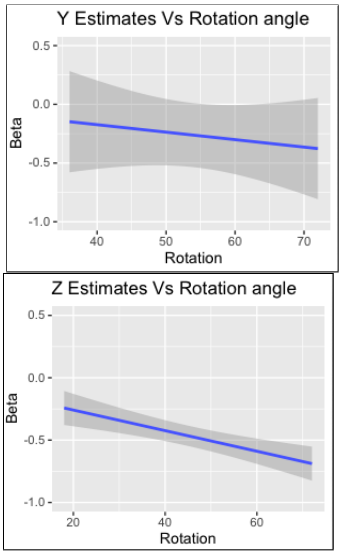

Figure 9: The relationship between the Y and Z rotation angles on GLMz model estimates for Y and Z respectively ($R^2 = 0.95$ for Y and $R^2 = 0.97$ for Z).

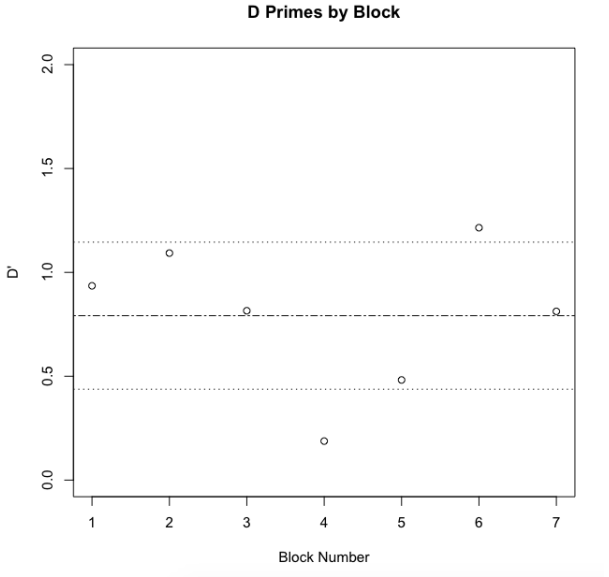

Figure 10: Average discriminability for all objects within each block.

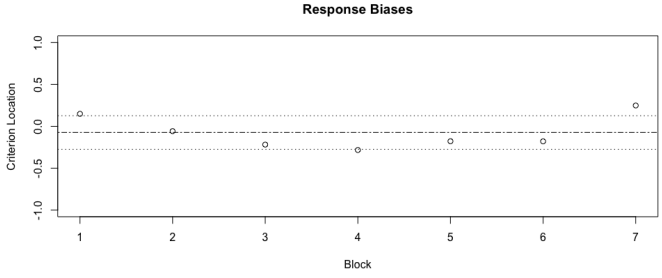

Figure 11: Experiment result: Plot showing the response bias measured in terms of criterion location for each block.

$$C = -[z(H) + z(F)]/2 \tag{3}$$

where $z(H)$ denotes the z score for fraction of hits and $z(F)$ denotes the z scores for fraction of misses. The aggregate response criterion for each block is shown in Figure 11. A sign test on the criterion locations for blocks (s = 2, p-value = 0.4531) reveals that there is no evidence for a systematic bias on criterion location (i.e., errors across blocks were random). A test for the location of criteria for all valid test subjects (s = 10, p-value = 1) revealed that there was no evidence of criteria being different than zero, showing no difference between the number of false alarm and misses in subject responses.

3.2. Model Results

The model computed a Gram matrix using an estimate of the z values that minimized the standard deviation of all 3D angles in the reconstructed shape. The experiment on the other hand measured the consistency of shape perception under various rotations of a given object. Since the actual reconstructed shape by human participants is never available to compare with the model estimate, a new metric was devised to quantify the performance of the model. The consistency of shape recovery by the model was measured by quantifying the similarity in the 3D angles estimated from different rotated views of a given object. The 3D angles are contained in the Gram matrix generated for all rotations of a given object. The standard deviation of euclidean distances between these Gram matrices is used as a proxy for measuring consistency of 3D shape recovery.

A network to process up to twenty vertices at a time was trained on a set of randomized cuboid-based shapes using respective SDA values. The output of the network is the SDA value for each stimulus. Since the network learned to minimize the SDA value, the error rate of the network was measured in terms of the mean SDA value per batch of input. The network was then tested on a set of unseen stimuli. The average SDA as measured by the stimulus rendering software was around 0.03 for the training and test stimulus sets (respectively). In the training phase, the model started with

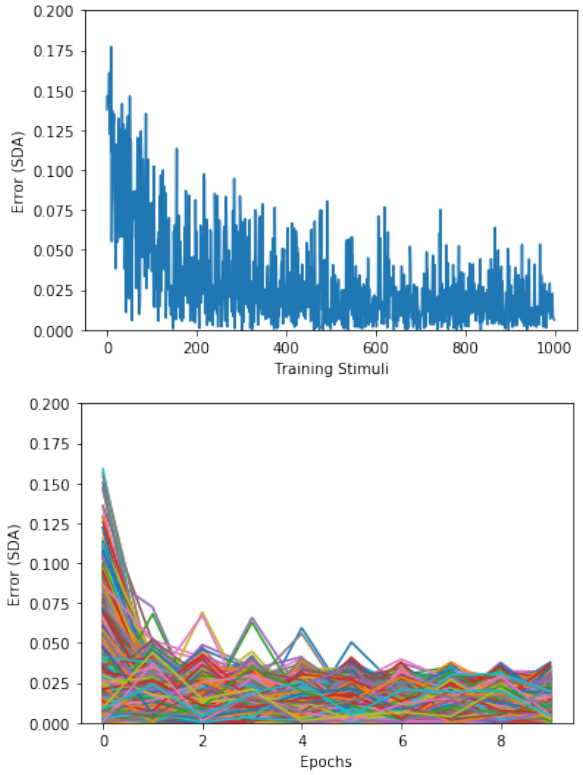

Figure 12: Top: Error plot during the first epoch of training samples; Bottom: Error rate showing model minimizing SDA values during 10 epochs of 1000 training samples each.

a high error rate of 0.2 and gradually decreased its error to 0.01 over 10 epochs of training with 1000 stimuli with the batch size of 5. Figure 12 shows the network error rate over a training sequence of the first 1000 objects.

The average SDA value in Radians after several epochs of training was 0.025, which translated to degrees is 1.43 Degrees. This value is very small because the network has been trained extensively until the point that a stable value for SDA persists. However, the value of SDA for a novel stimulus input to the model can be higher than this value. Depending on the complexity of the input, the SDA value computed by the network for an unseen 3D shape can range between 0.25 (i.e, 14.3 degrees) to 0.025 (1.4 degrees).

The network was finally tested with all the objects in the experiment blocks. In order to test for shape constancy using the model, each of the eight rotations of the experiment objects was presented to the model for comparison. The model estimated the missing depth (z coordinates) for each of these eight views of the object by minimizing the SDA value in the estimated 3D object. For each of these eight views, the Gram matrix of 3D angles is obtained from the model. To test the consistency of estimated 3D shapes across different rotation angles, the l2 norm of the Euclidean distance between the Gram matrices for the rotated and original views was computed. Since there is no access to the perceived 3D shapes from the experiment, this metric helps in comparing the model performance and the experiment out-

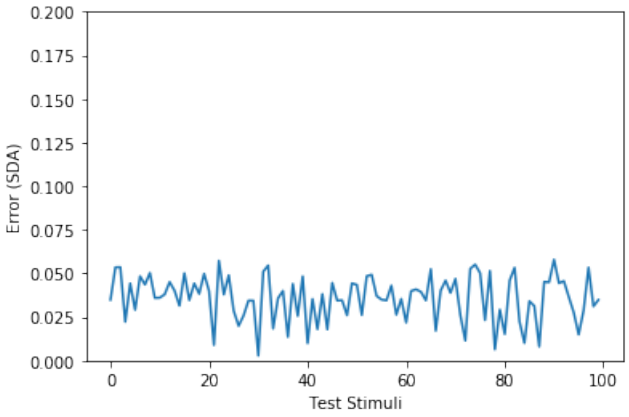

Figure 13: Model performance for 100 unseen randomly generated cuboid-based stimuli.

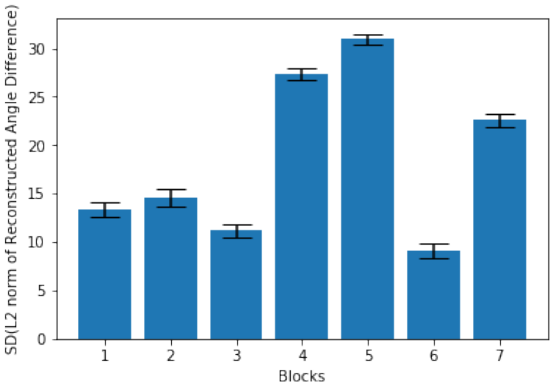

Figure 14: Performance of the model in the experiment blocks. The best performing blocks were Blocks 6, 1, and 2. The worst performing blocks were Blocks 4 and 5. The error bars denote the standard deviation of the computed metric across 20 different simulations of the trained model.

come. It is to be noted that object constancy is achieved only when it is perceived consistently across different rotational viewpoints. The experiment results therefore demonstrate the performance consistency at the block level for all tested stimuli. The standard deviation of this metric from the model shows the extent to which estimated 3D shapes deviate from the original estimate. Lower values of the standard deviation means that the shape recovery is more consistent across different viewing angles by the model.

The performance of the model qualitatively matched the results from the experiment as shown in Figure 14. The criterion of success was proposed to be how closely the order of reconstruction consistency from 2D input by the model matched the ordering of block difficulty in human subjects. The analyses of experiment outcome and the output from the model show similar results. As in the experiment, the blocks that contained high complexity objects outperformed those with lower complexity objects. Blocks with lower difficulty - Block 1, Block 2 and Block 6 - showed better performance than blocks with higher difficulty - Block 4 and Block

5. Block 3, which was of moderate difficulty in the Experiment, performed better than difficult blocks but worse than easier blocks in the model.

There are however some differences in the performance of the model on some blocks compared to the experiment outcome. For instance, although Blocks 4 and Block 5 are the worst performing blocks in both the experiment and the model, their order of performance was not the same. These differences can be expected because the human visual system uses several constraints at once to perceive a unique 3D structure. In that aspect, the model is highly limited because it used only one constraint. However, the results are encouraging and the constraint that the model used showed considerable effectiveness in modeling human performance.

4. Conclusion and Future Work

The task performed by this DNN model is different compared to most of the DNNs in the current literature that perform classification and regression tasks. Even within the domain of neural networks used for 3D reconstruction problems, this network is different in the sense that it does not get the ground truth of the 3D shape during the training. This model learns to minimize SDA value of 3D output. By learning to solve the optimization problem of minimizing SDA, the network is able to find the most suitable z-parameters for a particular stimulus.

Similar to other neural networks driven by backpropagation, this network minimizes a loss function that is computed at the final layer of the network. It is well-known that backpropagation uses gradient descent to minimize the value of this loss function. It can be said that by virtue of backpropagation all neural networks are solving an optimization problem. However, for a network claiming to solve an optimization problem, the problem needs to be formulated explicitly. In specific cases, optimization problems can be defined explicitly and neural networks can be designed to solve the problem directly or as a surrogate solver in conjunction with other mathematical solvers.

In this particular case, the problem of finding the best z parameter has not been formulated strictly as an optimization problem. If this problem was formulated as an optimization problem, a mathematical solver would be required to assist the network at some level. Incorporating a mathematical solver would have defeated the goal of finding a network architecture based on biological principles to solve this problem. In this case, the architecture and the geometrical computations inside the layers encode the constraints of MSDA into the network. By computing a Gram matrix of pairs of 3D angles and minimizing the standard deviation of this matrix using backpropagation, this network geometrically encodes the problem rather than mathematically. Therefore it can be said that this network implicitly solves an optimization problem by finding the best z parameters while minimizing a loss function that represents the MSDA constraint.

This network can generalize well to different sets of inputs given sufficient training. For instance, the set of novel objects used to test the model was never seen by the model. The training set consisted of cuboid shapes of different sizes. Each shape in the training set was created by taking a symmetrical cuboid and adding a random amount of displacement on each vertex along each of the x, y and z axes.

A limitation of this model is that it only works with 2D inputs in a coordinate system which is relative to the model's input canvas. In that sense it cannot generalize to any given 2D line drawing unless correctly formatted and presented to this model. Also, the input requires an encoding of which edges in the stimulus are visible. This is different from the DNNs in current literature for image processing, which often take raw images as input.

In conclusion, the goal of the model was to demonstrate a computational approach to optimize psychophysical constraints within a network. The model used only the constraint of minimization of standard deviation of all angles to estimate 3D structure. All computations required to compute and minimize this constraint were embedded within the network itself.

An experiment to test human subjects for 3D perception of novel and unfamiliar objects is described and the results are presented. The goal was to use the results from this experiment to test the validity of the proposed model. Based on the output of the model from the objects used in the experiment, it was shown that the model may reproduce the shape constancy achieved by human subjects on a similar set of novel stimuli. The degree of accuracy to which the model can do this can vary significantly since the human visual system uses several other constraints for 3D perception of object shape. Since the model and subjects from the experiment failed on the same type of stimuli (at the block level), the analysis of these failures suggests that the MSDA constraint is an effective first step for reliable shape perception. The similarity of outcome of the model with the experiment results shows that a network-based model can implement the visual constraint of MSDA. The results also provide a proof of concept for this biologically-inspired network to compute the required constraint.

A future extension of this work can be to implement more constraints into the model to generate 3D shapes for rotated views of a 2D stimulus. These 3D shapes can then be used to test whether human observers agree with the reconstruction by designing a similar experiment. The observers can be shown different valid reconstructions for the 2D stimuli to gauge if their preference agrees with the model or not. Computationally, embedding more than one constraint in the same network can show new insights of how networks can achieve 3D shape recovery using psychophysical principles.

In conclusion, embedding the constraint of MSDA in a network is shown to be effective in predicting human performance on a set of novel shapes. The model provides a proof of concept for how a biologically-inspired network may achieve such a task. It will be an interesting future research path to explore whether embedding other psychophysical constraints in the network can shed more light on how the human vi-

sual system uses built-in constraints to understand our three-dimensional environment.

A. Appendix

A.1. Algorithms

Algorithm 1 Create Edge Connection Vector a for given stimulus

Require: $V_0 \dots V_{N_v-1}$ ▷ All visible vertices in the current view.

Require: 3D Mesh object containing all vertices and connections.

Ensure: $ConnVec$ (An array of all possible connections between every vertex pair in the stimulus. An existing connection carries the value of 1 at the appropriate position in the array while non-existing connections have the value 0.)

function SEARCHEDGES($V_0 \dots V_{N_v-1}$)
  $ConnVec \leftarrow [\,]$
  for $step \leftarrow 1$ to $(N_v - 1)$ **do**
    $i \leftarrow 0$
    for $j \leftarrow (i + step)$ to N_v **do**
      if V_i is connected to V_j **then**
        $ConnVec \leftarrow [ConnVec, 1]$
      else
        $ConnVec \leftarrow [ConnVec, 0]$
      end if
      $i \leftarrow i + 1$
    end for
  end for
  return $ConnVec$
end function

A.2. Convolutional Layer Configuration

This section explains the operational details of convolutional layers including the values of key parameters used in the model.

A.2.1. Configured Parameters:

Channels: 3

Different channels allow for parallel operations on the set of inputs. There are 3 separate channels for 3 different coordinates: [x,y,z]. The operations across channels are fully independent but identical.

Filter size: 2

Filter or kernel size describes the size of the smallest matrix operation in the convolution layers. The filter or kernel is convolved with the input to produce the output. Since the model operates on a pair of vertices at a time to compute the edge vector, the filter size is set to 2.

Stride: 1

The rate at which the kernel passes over the input. A stride of 1 moves the kernel in increments of 1 unit.

Dilation: 1,2,,..Number of Edges

Distance between two consecutive units in a layer to be considered in the convolution operation. In order to compute all possible lists of edges, successive convolution layers compute a distance d dilation apart where d is the dilation value.

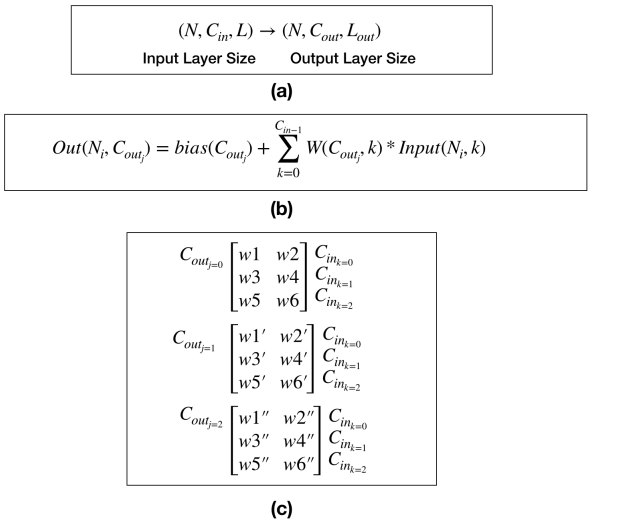

Figure 15: The relationship between the input layer parameters and output layer parameters in the convolutional layers

Weights:

$$\begin{bmatrix} 1 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 1 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & -1 \end{bmatrix}$$

Each convolution layer is initialized with these weight matrices. Each 3x2 matrix represents an input channel. The filter of [1,-1] is used to compute the differences between the x, y and z coordinates in successive channels.

A.2.2. Mathematical Details

Figure 15 (b) shows the computations involved for each batch for the convolution. Here W is the weight matrix associated with the layer. Figure 15 (c) visually depicts how the weight matrix is related to the input and output channels based on the equation shown in part (b). The parameters for batch size, layer size, channels, kernel size, and weight matrix that were utilized in the convolution operation were individually described in Section 3 along with the values used in the model.

$$L_{out} = \frac{L_{in} + 2 * \text{padding} - \text{dilation} * (\text{kernelsize - 1}) - 1}{\text{stride}} + 1 \tag{4}$$

Figure 15 (a) shows the high level relationship between the input and output layer sizes (L) of a convolution operation given the number of channels (C) and number of batches processed (N). The way L_{out} is related to L_{in} is shown in Equation 4.

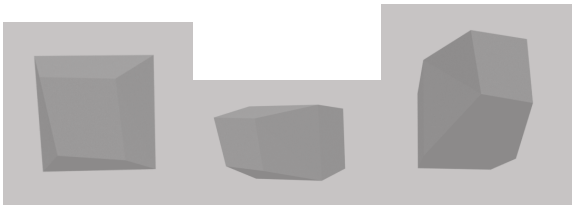

Figure 16: Stimuli example: Same object shown from 3 different projection viewpoints

A.3. Novel Stimuli Generation

An open source 3D graphic rendering tool called 'Blender' was used to create 3D models of the novel structured stimulus objects. The software allows for Python based programmatic creation, manipulation, and extraction of data. Object parameters can be extracted in the form of a text file along with images from a variety of rotation viewpoints and projections as shown in Figure 16. The stimulus parameters exported from the software can be used as input into the model and the corresponding images can be used for the experiment.

Each novel stimulus object was created programatically by applying a set of transformations on an original cuboid object. The set of transformations applied in order were:

1. Randomization: This transformation operation randomly displaced the location of selected vertices. The amount of displacement along an axis can be specified. A random offset is added to the given displacement value to obtain a randomized transformation. A seed value is used to control this random transformation by controlling the offset. A different seed will produce a new result whereas the same seed will result in the same output every time.
2. Mirroring: Mirrors the geometry of an object along an axis. The resulting geometry is joined together using a merge distance parameter. Pairs of original and newly mirrored vertices can be welded together using the merge distance parameter, which defines the minimum distance for the welding operation to happen.
3. Symmetrizing: Makes the mesh object symmetrical. Unlike mirroring, it only copies in one direction, as specified by the "direction" parameter. The edges and faces that cross the plane of symmetry are split as needed to enforce symmetry. Just like mirroring, this operation takes a minimum distance parameter to enforce symmetry from the central pivot point.

Novel, partly symmetric and structured objects were created from a cuboid by choosing the amount of randomization, pivot points, and merging distances for mirroring and symmetry operations. Fewer randomization operations lead to simpler shapes. The final number of vertices in a transformed object depends on the mirroring and symmetrizing operations. These operations are controlled by the merge distance parameter. A table containing the objects used in the experiment blocks and configuration parameters for each of them is presented in Figure 17.

Block	Object 1	Number of Randomizations	Cursor Position	Merge Distance Mirror	Merge Distance Symmetry
1	B1-Obj 1	5	(1,0,0)	0.8	0.1
	B1-Obj 2	5	(2.5,0,0)	0.8	0.1
	B1-Obj 3	5	(0.6,0,0)	0.8	0.1
2	B2-Obj 1	20	(0,0,0)	1	0.1
	B2-Obj 2	20	(1,0,0)	0.1	0.45
	B2-Obj 3	20	(1,1,1)	0.01	0.01
3	B3-Obj 1	10	(1,1,1)	0.01	0.01
	B3-Obj2	10	(0.5,0.5,0.5)	0.01	0.01
4	B4-Obj 1	3	(0.001,0.1,01)	0	0
	B4-Obj 2	3	(0,0,0)	0.01	0.01
5	B5-Obj 1	8	(0.1,0,0)	0.6	0.6
	B5-Obj 2	8	(0.1,0,0)	0.6	0.6
6	B6-Obj 1	5	(1.5,0,0)	0.8	0.1
	B6-Obj 2	5	(0.5,0,0)	0.8	0.1
7	B7-Obj 1	20	(0.001,0.1,01)	0	0
	B7-Obj 2	20	(1,1,1)	0.01	0.01
	B7-Obj 3	20	(1,0,0)	0.1	0.45

Figure 17: Table showing the parameter values used to generate objects for the Experiment Blocks.

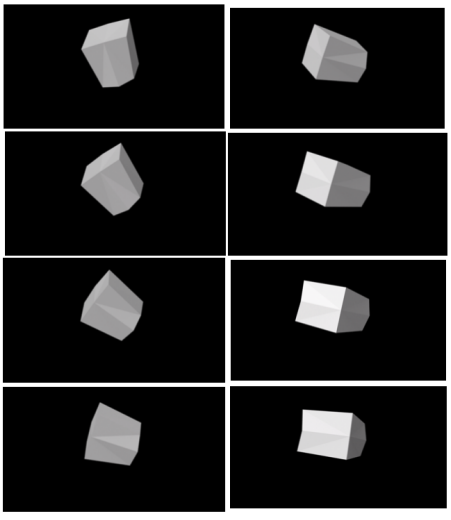

Figure 18: Example stimulus: Object 1 in Block 4 is shown from 8 different orientations.

The stimulus objects obtained from these operations are then rotated a fixed number of times in the Y and Z axes. All these views are then rendered in 3D for the different rotation angles. The output consists of a set of images for each object and a text file containing object properties including the 3D coordinates of its vertices and a connection matrix that encodes the pairs of vertices that are connected via an edge in the object.

An example of novel object created using Blender and captured in different orientations is depicted in Figure 18. The code used to generate the stimuli is available at `https://palmishr.github.io/3DStimuliBlender/`.

Acknowledgements

The authors would like to thank Rohit Mallick for his help in developing code for the experiment as well as running the experiment.

References

Cadieu, C.F., Hong, H., Yamins, D.L., Pinto, N., Ardila, D., Solomon, E.A., Majaj, N.J., DiCarlo, J.J., 2014. Deep neural networks rival the representation of primate it cortex for core visual object recognition. PLoS computational biology 10, e1003963.

Cao, L., Liu, J., Tang, X., 2008. What the back of the object looks like: 3d reconstruction from line drawings without hidden lines. IEEE Transactions on Pattern Analysis and Machine Intelligence 30, 507–517.

Chan, M.W., Stevenson, A.K., Li, Y., Pizlo, Z., 2006. Binocular shape constancy from novel views: The role of a priori constraints. Perception & Psychophysics 68, 1124–1139.

Dekel, R., 2017. Human perception in computer vision. arXiv preprint arXiv:1701.04674 .

Desimone, R., Schein, S.J., 1987. Visual properties of neurons in area v4 of the macaque: sensitivity to stimulus form. Journal of neurophysiology 57, 835–868.

Finlayson, N.J., Zhang, X., Golomb, J.D., 2017. Differential patterns of 2d location versus depth decoding along the visual hierarchy. NeuroImage 147, 507–516.

Fischer, 2014. Model of all known spatial maps in primary visual cortex. Master's thesis, The University of Edinburgh, UK .

Fischer, J., Spotswood, N., Whitney, D., 2011. The emergence of perceived position in the visual system. Journal of Cognitive Neuroscience 23, 119–136.

Grill-Spector, K., Malach, R., 2004. The human visual cortex. Annu. Rev. Neurosci. 27, 649–677.

Khaligh-Razavi, S.M., Kriegeskorte, N., 2014. Deep supervised, but not unsupervised, models may explain it cortical representation. PLoS computational biology 10, e1003915.

Li, Y., Pizlo, Z., Steinman, R.M., 2009. A computational model that recovers the 3D shape of an object from a single 2D retinal representation. Vision Research 49, 979–991. URL: http://dx.doi.org/10.1016/j.visres.2008.05.013, doi:10.1016/j.visres.2008.05.013.

Mountcastle, V.B., Motter, B., Steinmetz, M., Sestokas, A., 1987. Common and differential effects of attentive fixation on the excitability of parietal and prestriate (v4) cortical visual neurons in the macaque monkey. Journal of Neuroscience 7, 2239–2255.

Pasupathy, A., Connor, C.E., 2001. Shape representation in area v4: position-specific tuning for boundary conformation. Journal of neurophysiology 86, 2505–2519.

Pizlo, 2001. Perception viewed as an inverse problem. Vision research 41, 3145–3161.

Pizlo, Stevenson, 1999. Shape constancy from novel views. Perception & Psychophysics 61, 1299–1307.

Poggio, T., Koch, C., 1985. Ill-posed problems early vision: from computational theory to analogue networks. Proc. R. Soc. Lond. B 226, 303–323.

Roe, A.W., Chelazzi, L., Connor, C.E., Conway, B.R., Fujita, I., Gallant, J.L., Lu, H., Vanduffel, W., 2012. Toward a unified theory of visual area v4. Neuron 74, 12–29.

Schwartz, E.L., 1980. Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding. Vision research 20, 645–669.

Tikhonov, A., Arsenin, V.Y., 1977. Methods for solving ill-posed problems. John Wiley and Sons, Inc.

Yamins, D.L., Hong, H., Cadieu, C.F., Solomon, E.A., Seibert, D., DiCarlo, J.J., 2014. Performance-optimized hierarchical models predict neural responses in higher visual cortex. Proceedings of the National Academy of Sciences 111, 8619–8624.