VISUAL CONSTRAINT OPTIMIZATION NETWORK

by

Pallavi Mishra

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

Department of Psychology

West Lafayette, Indiana

December 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Prof. Sébastien Hélie, Chair
Department of Psychological Sciences
Prof. Gregory Francis,
Department of Psychological Sciences
Prof. Gesualdo Scutari,
School of Industrial Engineering

Approved by:

Dr. David Rollock
Head of the Graduate Program

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

One of the most important aspects of visual perception is the inference of 3D shape from a 2D retinal image of the real world. The existence of several valid mapping functions from object to data makes this inverse problem ill-posed and therefore computationally difficult. In the human vision, the retinal image is a 2D projection of the 3D real world. The visual system imposes certain constraints on the family of solutions in order to uniquely and efficiently solve this inverse problem. This project specifically focuses on the aspect of minimization of standard deviation of all 3D angles (MSDA) for 3D perception. Our goal is to use a Deep Convolutional Neural Network based on biological principles derived from visual area V4 to solve 3D reconstruction using constrained minimization of MSDA. We conduct an experiment with novel shapes with human participants to collect data and test our model.

CHAPTER 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" (Pizlo, 2001b; Poggio & Koch, 1985; Tikhonov & Arsenin, 1977). An inverse problem is defined as a mapping from measurements (data from visual perception) to model parameters (objects in consideration). The inverse problem is the inverse of the forward problem - a mapping from the object to the measurements or data. Solving an inverse problem amounts to finding the estimations of parameters (of objects) from knowledge of the data. 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 real world. It has been postulated in Pizlo (2001b) that the visual system imposes certain constraints on the family of solutions in order to efficiently solve this inverse problem.

1.1 A Typical Formulation of Ill-posed Inverse Problem

A typical formulation of the ill-posed inverse problem in 3D vision as depicted in Pizlo (2001b) is shown in Equation 1.1. Here, I_{2D} is the 2D retinal image, η_{3D} is the actual object, $F_{projection}$ is a function that projects 3D object onto a 2D surface and ε_{2D} is the error in measuring I_{2D}. It is to be noted that the image is erroneous with error ε_{2D} embedded in its measurement and the inverse function $F^{-1}_{projection}$ has to take this into account. η'_{3D} in Equation 1.2 is the estimate of the 3D object as obtained by the inverse of projection function $F^{-1}_{projection}$. In order to solve this problem as a constrained optimization problem, one way to formulate the cost function is presented in Equation 1.3.

$$I_{2D} = F_{projection}(\eta_{3D}) + \varepsilon_{2D} \tag{1.1}$$

$$\eta'_{3D} = F^{-1}_{projection}(I_{2D}) \tag{1.2}$$

$$E_{total} = ||F_{projection}(\eta'_{3D}) - I_{2D}||^2 + \lambda ||e_{constraint}(\eta'_{3D})|| \tag{1.3}$$

The first term in the cost function Equation 1.3 for E_{total} evaluates how consistent the projected image of η'_{3D} is with the retinal image I_{2D}. The second term evaluates how well η'_{3D} satisfies some a-priori constraints. Recovering a 3D shape that best satisfies both these requirements is equivalent to finding the global minimum of $E_{total}(\eta'_{3D}, \varepsilon_{2D})$ in the space of all valid 3D shapes. Sometimes there are two (or more) local minima of the cost function.

1.2 Alternative Approaches

Several ideas have been explored to understand how human vision is able to derive 3D structure of objects from the 2D retinal images. For example, an interpretation scheme for deriving 3D structure from motion using rigidity principle has been proposed in Ullman (1979). The spatial-temporal integration of retinal information can be used to decipher various object properties (or cues) to approximate depth and shape information in Landy, Maloney, Johnston, and Young (1995). These properties include texture as proposed in Aloimonos (1988), perceived color as proposed in Cavanagh (1987), motion as proposed in Richards (1985), shading as proposed in Clark and Yuille (1990) and so on. The perceived depth information from these cues is probabilistic and contextual. None of these approaches will be discussed in the current work as these approaches for measurement of depth from image cues are different from the 3D shape perception approach in a significant way. All of these approaches are generally considered to be a part of the early visual processing system as categorized in Poggio and Koch (1985). In contrast, 3D shape reconstruction as inverse problem approach can be positioned at the next stage in processing where surface contours, edges and vertices have already been extracted from the image. The problem then remains to combine these sub-structures using what Pizlo, Sawada, Li, Kropatsch, and Steinman (2010) described are 'built-in' mechanisms of simplicity principles that automatically apply certain constraints without using any contextual information from the 2D scene. Apart from the use of regularization method discussed above, Bayesian methods have been used in augmenting missing information from 2D projection to infer depth and shape information Pizlo (2001a).

It is to be noted that in the inverse problem approach, there may even be a complete loss of depth information. The lost depth information may not be recovered by making assumptions and using constraints to approximate the 3D shape. In such degenerate cases, human vision often fails to achieve any shape constancy. This means that the 3D shape reconstruction is unreliable in those cases. Shape constancy is also difficult to achieve with irregular and unstructured 3D objects because application of simplicity constraints (discussed below) becomes difficult in case of completely unstructured 3D objects.

1.3 3D Perception and the Role of Visual Constraints

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 not enough to solve for a unique shape perception. The visual system may impose one or more of these constraints on the set of solutions in order to uniquely solve this inverse problem (Pizlo, 2001b).

1.3.1 Visual Constraints

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 traditional Gestalt approach based on 'Law of Prägnanz' or simplicity principle (the principles of closure, good continuation, regularity, symmetry, simplicity and so forth). These constraints are chosen specifically due to their demonstrated effectiveness in generating reliable 3D percepts in models of 3D vision (Li et al., 2009a).

1.3.2 Minimization of standard deviation of angles (MSDA Principle)

The principle of minimization of standard deviation of all the angles in the reconstructed object was first proposed in Marill (1991). Marill described this principle conceptually as considering the orthographic extension of a given two dimensional object (line drawing) as the input. For such an input, all three dimensional angles in the orthographic projection have to be determined and their standard deviation needs to be computed. Then, the 3D object for which the standard deviation of all 3D angles is minimum, is the acceptable interpretation. This means that regular, symmetric and simple 2D shapes such as regular polyhedrons which have a small standard deviation of angles are more preferred shapes than skewed irregular polygon. The same is true according to the Law of Prägnanz. Therefore the former will have a higher likelihood of achieving shape constancy by human subjects than the latter (Pizlo, 2001b). There have been several variants of the MSDA principle since Marill (1991) which include minimization of the standard deviation of the segment magnitudes of edges (MSDSM) (Brown & Wang, 1996) and minimization of entropy of angle distribution between line segments in a 3D wire-frame (MEAD) (Shoji et al., 2001). An illustration of the use of entropy of angle distribution for reconstruction of 3D shapes from 2D line drawings as presented in Shoji et al. (2001) is shown in Figure 1.

1.3.3 Symmetry

Symmetry can be present in the 3D structure of an object in several orders. The most simple symmetry is perhaps bilateral symmetry. Bilateral symmetry is also understood as plane symmetry as there exists a plane that divides the object into mirror image halves. This can also be referred to as mirror symmetry. A large majority of animals (almost 99%)(Finnerty (2005)) exhibit this type of symmetry. It has been shown by Vetter (1994) that for symmetry of higher orders (existence of more than one symmetry plane) the entire 3D Euclidean structure of the object can be recovered. Several computational models of 3D shape reconstruction from single 2D image (Pizlo, 2001b; Vetter, 1994)) restrict their inputs to having at least one plane of symmetry since shape constancy is limited in case of asymmetric 3D shapes as discussed earlier. In our case, the stimuli are generated with one plane of symmetry in order to achieve a reliable shape constancy by human subjects as well as our computational model.

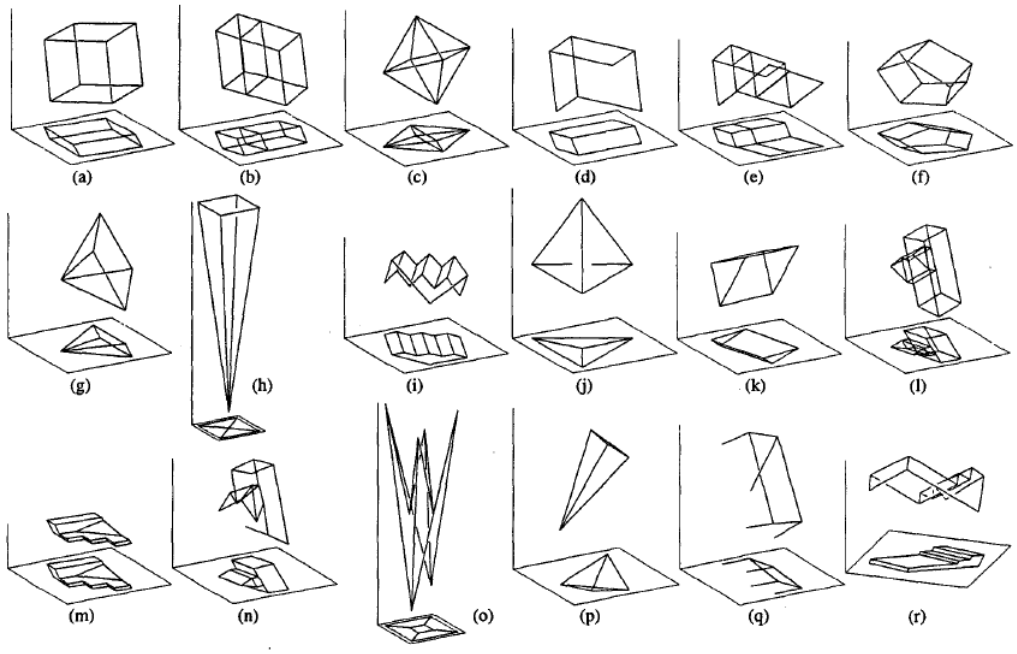


Figure 1. Example 3D objects reconstructed from 2D line drawings using a modified version of the MSDA principle that uses entropy minimization in angle distribution (Shoji et al., 2001). For every 2D line drawing presented in horizontal XY plane, a 3D interpretation is shown using vertical Z axis. The 3D interpretation that minimizes entropy in angle distribution is the preferred interpretation.

Bilateral symmetry can be used to restrict the family of 3D shapes recovered from a 2D model. The family of shapes are further restricted in Li et al. (2009a) using other priors such as compactness and surface area. A 2D model of the object can be represented as a set of point-wise feature vector X = $(x_1, y_1, x_2, y_2 \ldots x_n, y_n)$. Assuming mirror symmetry (based on known class of the object) one can find corresponding symmetric pairs of points across the symmetric plane: (x_L, y_L) and (x_R, y_R) as shown in Figure 2. A computational model to recover full 3D shape using this approach is presented in Li, Pizlo, and Steinman (2009b).

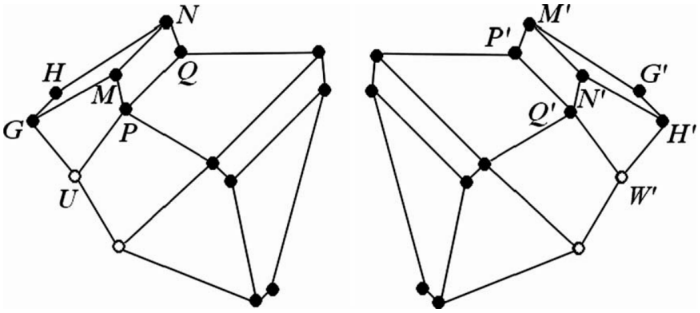


Figure 2. Image from Li et al. (2009a) showing original (left) and virtual (right) views of a symmetrical object. The corresponding vertices in the virtual view are named as H' for vertex H in the real image and so on. All pairs of visible vertices are shown in solid dots. The vertices that can be reconstructed using both symmetry and planarity constraint are shown by open dots. The symmetric counterpart of a vertex is obtained by reflecting that vertex on a symmetry plane. By assuming a planar surface, some vertices such as U can be reconstructed using the symmetric reflection operation.

1.3.4 Planarity

Planarity is also a simplicity constraint in a sense that a planar curve is simpler than a non-planar curve because there is more uncertainty of information in a non-planar curve than a planar curve. The constraint pf planarity has been applied mostly in conjunction with other constraints such as the MSDA constraint (Leclerc & Fischler, 1992; Liu, Cao, Li, & Tang, 2008). In certain cases, where all symmetric vertices are hidden from view, the assumption of planarity can help in the interpretation of the 3D shape by prioritizing the simplicity of shape over other more complex possibilities (Pizlo, 2001b).

1.3.5 Compactness

Typically defined as the ratio of a surface area to the perimeter of a given 2D object, compactness also maximizes what Gestalt psychologists called 'simplicity' of a shape. For a 3D case, Gestalt psychologists offered a physical model of a soap bubble since it minimizes surface area to volume ratio due to physical forces of surface tension. It has been quite challenging to boil down the concept of compactness or simplicity for a given solid into a physical, measurable parameter but several models in psychophysics have used regularization theory to make this task viable (Pizlo et al., 2010). This constraint has been used in conjunction with other constraint to create unique 3D percept in a situation where there are more than a single interpretation of 3D shape from 2D line drawing (Li et al., 2009a).

1.4 Summary

In this section, a brief overview of the principle of inverse problem formulation of 3D shape reconstruction by the human vision is presented. This formulation requires the use of various constraints to solve the inverse problem. A brief introduction to some of these visual constraints is presented using a few examples. In this work, the constraint of minimization of standard deviation of 3D angles (MSDA) will be used to solve the inverse problem of 3D shape reconstruction in a deep neural network based model. The reason for using this particular constraint is that, using this constraint is a good starting point for this line of research. Additionally, computing 3D angle pairs from 2D input is computationally simpler to do in a network than computing the other constraints. The details involving the computations required in a network for this constraint are discussed in the next chapter.

CHAPTER 2. MODELING PRINCIPLES

In order to build a computational model that is based on biological principles of information processing, areas in the visual cortex especially the computational anatomy of the striate cortex and some functional properties of the visual area V4 are taken into account. The goal is to demonstrate how a computational approach based on biological principles may perform constraint optimization in a network. The reason for emphasis on computation in a network is simply because the brain itself is a network. The choice of Deep Neural Network substrate (DNN) for our computational model is based on the recent discovery of interesting properties of DNNs embedding general purpose visual computations while displaying extraordinary task-trained accuracy on visual tasks. 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 and sensitivity to segmentation, crowding and shape has DNN end-computation correlates. It has also been shown (Cadieu et al., 2014; Khaligh-Razavi & Kriegeskorte, 2014; Yamins et al., 2014) that when the same images are processed by trained DNN and by 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 implementations 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.

2.1 Properties of visual area V4

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 et al., 2012). Studies (Desimone & Schein, 1987; Mountcastle, Motter, Steinmetz, & Sestokas, 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 & 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 structural shape coding scheme and also carries sufficient information for reconstruction of moderately complex shape boundaries. This insight is relevant because the proposed computational model uses stimulus object's relative coordinates to compute the edges in the stimulus. The edges are then used to extract properties about the overall shape of the object using matrix based operations.

2.2 Computational and functional architecture of Striate Cortex

In order to inform the architecture of our 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 is studied. It is known from several studies (Finlayson, Zhang, & Golomb, 2017; J. Fischer, Spotswood, & Whitney, 2011; Grill-Spector & Malach, 2004) that 2D spatial location information is encoded in several visual areas but it's magnitude or sensitivity decreases along the visual hierarchy. On the other hand, 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 of the 2D location coordinates and increasingly become 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 highly overlapped 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 depth dimension from lower dimensional inputs (including 2D coordinates) in the visual hierarchy. It was postulated in Schwartz (1980) that one way to encode higher dimension features such as depth using lower dimensional components such as 2D spatial coordinates of a scene can be demonstrated by the functional architecture of striate cortex. The columnar structures in striate cortex can allow encoding of higher 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 type of mapping algorithms present a way in which the computational architecture of striate cortex may allow 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 of these is presented in Fischer (2014)). Accordingly, in the model presented, the lower dimensional inputs reside in separate channels for each of the different dimensions. 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 visual constraint of MSDA. The model successively computes a differential mapping of the previous layer to extract higher order properties of the stimuli for the next layer. The detailed computation involved in each layer of the model will be covered in Chapter 3.

2.3 Computations required to minimize MSDA constraint

If 3D shape perception is solved at least in part by using constraints as described in Chapter 1, a network should consistently and correctly estimate the value of the missing depth dimension by optimizing that value based on these constraints. The scope of this work is limited to embedding the first constraint in the network - the minimization of standard deviation of angles (MSDA) principle. This constraint is the most interesting because the key set of computations to satisfy the MSDA constraint (such as edge computations using coordinates of the object) are most closely related to the area V4 functional architecture discussed in this chapter.

In the past, several computational approaches have been suggested to extract the missing depth information using the MSDA constraint. We take some time to describe them in detail while also making a case of our proposed approach.

- Strategy 1: Hill climbing algorithm (originally proposed in Marill (1991))

  This is a search based algorithm that finds an optimal set of depth coordinates for every vertex in the 3D structure so that the standard deviation of 3D angles between every pair of intersecting edges on all vertices is minimized. At each stage of the search, the SDA (standard deviation of angles) of the current vector is computed. Based on an arbitrary step-size, a number of new vectors (called child nodes) are computed. If s is the step-size and the current vector is $\vec{z} \equiv (z_0, z_l, z_{n-1}, ., z_n)$, then the children are: $(z_0 + s, z_l, ...), (z_0 - s, z_1, ...), (z_0, z_1 + s, ...), (z_0, z_l - s, ..., z_{n-1})$. These children are vectors that are one step-size away from the current vector. The value of SDA is computed for each of these $2n$ children, and the child with minimum SDA is selected as the new current vector. (If there is more than one, the first of these is selected.) The process then repeats, until no further improvement in SDA is obtained. The vector with the smallest SDA of those inspected is the result of the process.

- Strategy 2: Plane based optimization (originally proposed in Liu et al. (2008))

  Instead of minimizing the standard deviation of angles between all pairs of edges at every single vertex, one can use the faces of a 2D object to be the variables in the optimization process. The SDA principle can be applied indirectly in the form of geometric constraints. Suppose faces f_1, f_2, f_3 pass through a given vertex $v_1 = [x_1, y_1, z_1]^T$. If all faces meet at this vertex, they all share a common point z_1. Using this point we re-write the equations of the faces: $z_1 = a_1x_1 + b_1y_1 + c_1$, $z_1 = a_2x_1 + b_2y_1 + c_2$, $z_1 = a_3x_1 + b_3y_1 + c_3$. Now, eliminating z_1 and re-writing the same equations in matrix form:

  $$Pf = 0$$

  where P is a projection matrix of size M x $(3N)$ and

$f = [a_1, b_1, c_1, a_2, b_2, c_2 ... a_n, b_n, c_n]^T$ is called a face parameter vector. The objective function to minimize here can be written as:

$$\Psi(f) = \psi'(z_1(f), z_2(f), ..., z_N(f))$$

The task now is to find a face parameter vector f that minimizes the objective function $\Psi(f)$ subject to the condition that $f \in \text{NULL}(P)$. Where $\text{NULL}(P)$ is the null space for matrix P which is the set of all vectors v such that $P.v = 0$.

- Strategy 3: Gram Matrix based optimization (adapted from Boyd and Vandenberghe (2004))

  As in the case of the hill climbing based approach, this approach computes the 3D angles between all pairs of intersecting edges on a vertex so that the standard deviation of all angles in the 3D polyhedron is minimized. To this end, first a set of 3D edge vectors $v_1, v_2, ..., v_n \in \mathbb{R}^3$ are computed using an initial estimate of z. Each set of 3D vectors based on a given estimate of z is referred to as a configuration. Each configuration has a set of geometric properties that can be expressed in terms of a Gram Matrix given by:

  $$G = V^T V, \qquad V = [v_1, v_2, ..., v_n],$$

  so that, $G_{i,j} = v_i^T v_j$. The diagonal entries of G are given by:
  $G_{ii} = l_i^2, i = 1, ..., n, \quad l_1 = ||a_1||, ..., l_n = ||a_n||$

  The correlation coefficient between v_i and v_j is given by

  $$\rho_{i,j} = \frac{v_i^T v_j}{||v_i||_2 ||v_j||_2} = \frac{G_{i,j}}{l_i l_j}$$

  so that $G_{i,j}$ is a linear function of $\rho_{i,j}$. The angle $\theta_{i,j}$ between v_i and v_j is given by

  $$\theta_{i,j} = cos^{-1} \rho_{i,j} = cos^{-1}(G_{i,j}/(l_i l_j))$$

where $cos^{-1}\rho \in [0, \pi]$. This Gram matrix representation is invariant under orthogonal transformation. The Gram matrix is also symmetric and positive semidefinite. Since cos^{-1} is a monotonic function in $[0, \pi]$, one can minimize any particular angle $\theta_{i,j}$ by minimizing $G_{i,j}$. In order to minimize the SDA of all the angles, one can minimize the SDA of the matrix G itself.

2.4 Summary

Out of the three approaches discussed, the Gram matrix based approach is the most biologically realistic based on the discussion about computational and functional architecture of area V4. This approach allows for successive differential computation in the network based on 2D coordinates inputs of the stimuli. All the accompanying computations related to finding the 3D edge vectors and Gram matrix can be embedded into the network using a series of convolutional filters that emulate differential maps approach as discussed in Section 2.1.1. These computations will be discussed in further details in the next chapter. It is to be noted that the convolutional network based model learns to minimize SDA using gradient descent so, the best it can perform after training may be equal to or lower than the performance of an exact MSDA finding algorithm.

CHAPTER 3. MODEL DESCRIPTION

A DNN (Deep Neural Network) based model using Pytorch programming framework was developed based on the ideas developed in Chapter 2. The model attempts to use the MSDA constraint to estimate the missing depth parameter. The model does not attempt a full 3D reconstruction of the image but estimates the angles for the most plausible 3D structure based on the ideas in Chapter 1. The input for the model is a 2D canvas wherein coordinates of visible vertices are presented to the model. The model then computes the 3D angles from these vertices by learning to estimate the depth parameter that minimizes the standard deviation of all angles. In this chapter, the inputs and outputs of the model and the computations within the successive layers of the network is presented. The performance of this model was then compared to the performance of human subjects in a 3D shape perception experiment described in the next chapter.

3.1 Model Inputs

The input to the model consists of 2D coordinates of a stimulus object along with a connection vector representing edges existing between visible vertices. The model can process a batch of such objects at a time 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 are created programmatically in the fixed coordinate system, so the 2D coordinates system is consistent across all stimuli. For each stimulus, the following information is extracted: a) (x,y) coordinates of the vertices and b) list of edges between each pair of vertices. Figure 3 shows a sample of the input stimulus and a 3D voxel based rendering of the estimated object shape from the model.

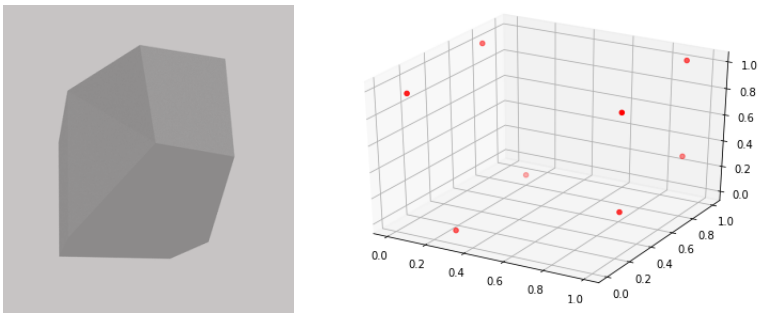


Figure 3. Left: Picture of a sample stimulus in 2D with some vertices hidden in orthogonal view; Right: A 3D voxel grid representing an estimated object shape by adding an estimated depth to each of the visible vertex in the 2D stimulus.

The 2D coordinates of vertices is in the order of vertex numbers starting from 0 to $(N_v - 1)$ where N_v is the total number of vertices. This order is preserved during the processing of the input in the model. The list of pairs of vertices connected by an edge is encoded using a simple scheme as shown in Algorithm 3.1. This algorithm encodes the edge connections in the same order that the DNN computes edges from vertices using convolutional layers. So the edge connection vector generated by the algorithm can be directly used inside the network to identify which connection to drop and which ones to keep while computing 3D angles of connected vertices.

Algorithm 3.1 Create Edge Connection Vector for a Stimulus

Require: $V_0 \ldots V_{N_v-1}$ ▷ All visible vertices in 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 value of 1 at the appropriate position in the array while non-existing connections have the value 0.)

```
1:  function SEARCHEDGES(V_0 ... V_{N_v-1})
2:      ConnVec ← [ ]
3:      for step ← 1 to (N_v − 1) do
4:          i ← 0
5:          for j ← (i + step) to N_v do
6:              if V_i is connected to V_j then
7:                  ConnVec ← [ConnVec, 1]
8:              else
9:                  ConnVec ← [ConnVec, 0]
10:             end if
11:             i ← i + 1
12:         end for
13:     end for
14:     return ConnVec
15: end function
```

Since the approach to using constraints is geometric, no other cues are fed into the model apart from the coordinate locations and a connection vector as shown above. This approach is in line with the discussion about visual area V4 where translation from retinal position to some reference coordinate system takes place and edges are computed before depth information is estimated.

3.2 Dimension Scaling

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 connection vector in the input needs to be re-structured to comply with the higher dimension of vertices. An algorithm that restructures a given connection matrix to a different dimension is presented in Algorithm 3.2. The maximum number of vertices is set to be twenty in our simulations.

Algorithm 3.2 Create Edge Connection Vector for a Stimulus

Require: N_t ▷ Total number of vertices, non-visible vertices are just 0.

Require: N_v ▷ Number of visible vertices.

Require: $ConnV$ ▷ Connection vector of visible vertices.

Ensure: $ConnV_{new}$ (An new array of connections between existing vertex pairs in the total list of vertices.)

1: **function** EXPANDCONNVEC

2: $ConnV_{new} \leftarrow [\,]$

3: **for** $i \leftarrow 1$ to $(N_t - 1)$ **do** ▷ Here i represents the ith convolution operation in the network.

4: $t \leftarrow length(ConnV)$ ▷ Total length remaining in original connection vector for processing.

5: $L_i \leftarrow (N_t - i - 1)$ ▷ The vector length for ith convolution operation.

6: $N_0 \leftarrow (N_t - N_v)$ ▷ The maximum number of empty cells in any given pass.

7: $pad \leftarrow \min(L_i, N_0)$ ▷ Minimum zero padding needed to expand to the required length in this pass.

8: $N_i \leftarrow (N_v - i - 1)$ ▷ Number of units in original vector eligible to be copied into new vector.

9: $n1 \leftarrow \max(Ni, 0)$ ▷ $n1$ units to be copied should always be non-negative.

10: $n2 \leftarrow (t - n1)$ ▷ $n2$ units remain in original vector for the next pass.

11: $C1, C2 \leftarrow \text{split } ConnV[n1, n2]$ ▷ Connection vector is split into two parts of length n1 and n2. Part n1 is expanded for extra vertices. Part n2 is the remaining vector for the next iteration.

12: $ConnV_{new} \leftarrow [C1, pad]$

13: $ConnV \leftarrow C2$

14: $i \leftarrow i + 1$

15: **end for**

16: **return** $ConnV_{new}$

17: **end function**

3.3 Visible Vertex Extraction

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 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 are equal to number of incomplete vertices. This assumption is based on the work of Cao, Liu, and Tang (2008) where psychophysical constraints are used to extract hidden structure from a partially visible object.

3.4 Model Output

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. The z-coordinates themselves are not accessible owing to the architectural limitations of deep convolutional networks. We will therefore need to use specific techniques to extract z-coordinate data from the model. The implementation details of these techniques will be discussed later in this chapter.

3.5 Overview of Model's Computations

The model receives the input containing 2D coordinates of the visible vertices along with the connection matrix. 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 3.1.

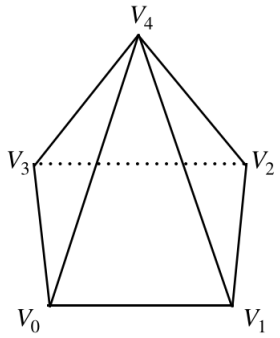


Figure 4. A sample stimulus with 5 vertices

$$E_{i,j} = V_j - V_i \tag{3.1}$$

Since the edge computations in each of the three dimension are identical operations, the model can work on the three dimensions in parallel as shown in Equation 3.2. Here, z^* represents the estimated z coordinates for vertices V_i and V_j.

$$E_{x_{i,j}} = x_j - x_i, \quad E_{y_{i,j}} = y_j - y_i, \quad E_{z^*_{i,j}} = z^*_j - z^*_i \tag{3.2}$$

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 5. The sample stimulus (Figure 4) encoded into model inputs.

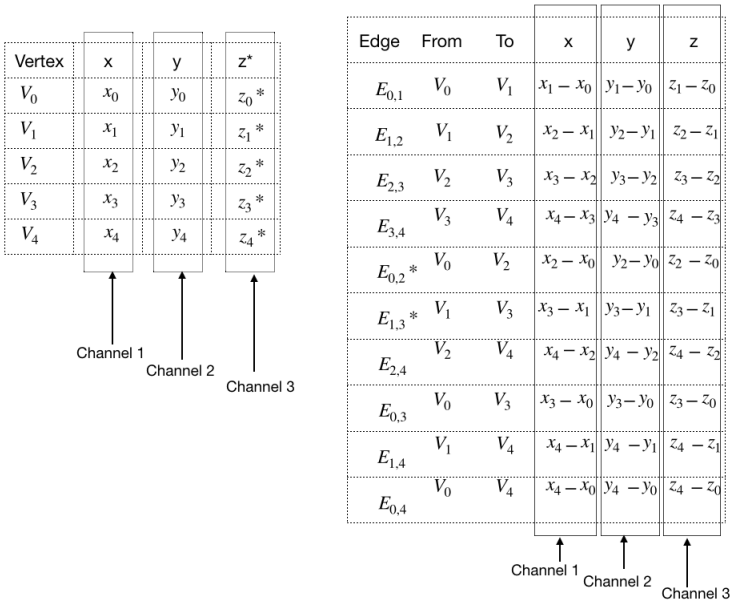


Figure 6. Identically divided computations across three separate channels of the network using the sample stimulus shown in Figure 4

Figure 4 shows a simple stimulus for illustration. The vertices in the sample stimulus and their coordinates in three dimensions are shown in Figure 5. Only the X and Y dimension is input into the model. The edge vector as shown in Figure 5 encodes the information about existing connections in the stimulus. The edges that exist correspond to the value 1 and the ones that do not exist have the value 0. Figure 5 also shows the connection between vertices that each edge represents. This relationship was established in Algorithm 3.1. It is to be noted that the last table in Figure 5 is shown only for illustration, the model input consists of the left and the middle tables in the figure. Figure 6 illustrates how the computation is distributed on 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 7. As depicted in the illustration, the first convolution operation computes the edge between vertices which are adjacent, the second convolution computes edges which have 1 vertex between them and the last convolution computes edges for vertices that are most further apart. All possible combination of vertices are covered in this process. This process computes edges in the same order as they are computed in Algorithm 3.1. An example showing the final set of computed edges for an object with 5 vertices is shown in Figure 8. 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 edge that does not exist in the object. Given our sample input stimulus in Figure 4 and corresponding edge vector table in Figure 5, the edges which do not exist in the stimulus have been marked with a * in the edge vector computed by the network in Figure 6. These edges will be dropped from all further computations. All the edges are then normalized so as to make it simpler to keep track of computations and intermediate results generated. Figure 8 illustrates the overall relationship between computations of edges by convolutional layers and the different channels for the three dimensions.

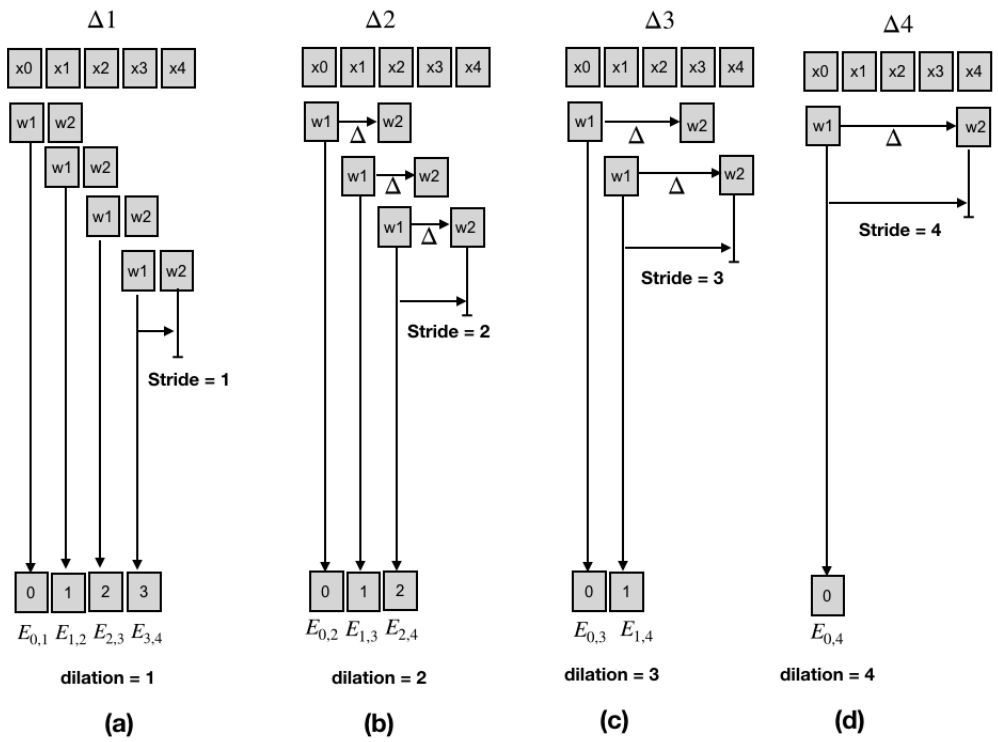


Figure 7. Illustration of edge computation operation in Convolutional layer for a particular channel (X in this case) (a) Dilation of 1 (b) Dilation of 2 (c) Dilation of 3 (d) Dilation of 4. The same operation is repeated in Y and Z dimensions on the second and third channel respectively.

Further details regarding specific values used to configure the convolutional layers along with details of mathematical operations involved in a typical convolution are presented in Appendix Section I.1.

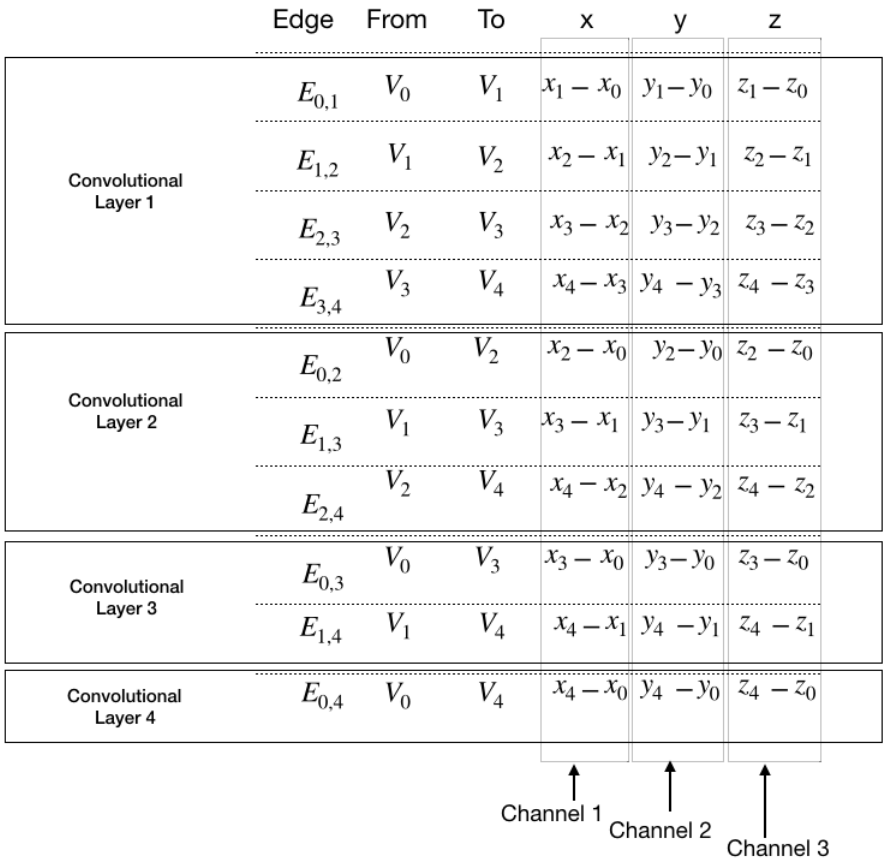


Figure 8. Table showing edges computed as a result of convolution operation in the network.

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 as shown in the matrix of Figure 9. In the matrix shown in this figure, the pair of numbers on the row and columns represent the edges between those vertices. Each cell corresponds to a combination of any two edges. The binary sequence on top of the rows and columns depicts the connection vector. The vertices are numbered in the figure starting from one for convenience.

The angle between a pair of edges is defined as:

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

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

[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.

3.5.1 Removal of Spurious Connections from Gram Matrix

The Gram matrix contains cells corresponding to all possible pairs of edges in the 3D object. For an object with N vertices, taking any two vertices at a time, there are $N_e = (N * (N-1))/2$ possible edges. Each cell in the Gram matrix represents an inner product between $E_i * E_j$ where $i, j \in 0 \ldots N_e$. So there are N_e x N_e angles represented in the Gram matrix. However, if there is no common vertex between any two given pair of edges, no angle can possibly exist between them. An illustration depicting all valid and invalid angles represented in cells of the Gram matrix for a cuboid object is shown in Figure 9. Since the Gram matrix is symmetric, it is sufficient to processes the information contained in the upper (or lower) triangular matrix. In order to remove the effect of these spurious connections from the computation of MSDA, Algorithm 3.3 is used to compute a filter matrix. This filter matrix is used to zero-out all invalid connections from the Gram matrix before it is used for computing the MSDA constraint.

		1	1	1	0	1	1	1	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	0	0	0	0	0
		12	23	34	45	56	67	78	13	24	35	46	57	68	14	25	36	47	58	15	26	37	48	16	27	38	17	28	18
1	12	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0
1	23	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	34	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	56	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
1	67	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	78	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0
0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
1	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 9. A sample Gram matrix for a cuboid object (eight vertices). The color coding is used to depict valid and invalid connections in the matrix for the cuboid object. Red colored cells depicts an invalid connection because no vertex is shared between the edge pairs in the corresponding row and column. A green colored cell depicts that a connection is possible.

Algorithm 3.3 Compute a Filter Matrix to Remove Spurious Connections.

Require: N_C: Total number of convolutional layers in a network.

Ensure: *FilterMat* (A filter matrix has same dimension as Gram matrix. It zeroes out cells that have no common vertex between the pairs of edges. The inner product of Gram matrix with Filter matrix gives only the viable angles in the object.)

1: **function** CREATEFILTERMAT(N_v)
2: FilterMat $\leftarrow$ [ ]
3: **for** $r \leftarrow (N_C)$ to (1) **do** ▷ For each convolutional layer, a row containing N_C smaller matrices is computed.
4: RowMat $\leftarrow$ [ ] ▷ Stores N_C smaller filters.
5: **for** $n \leftarrow r$ to 1 **do**
6: fMat $\leftarrow r \times n$ matrix of ones ▷ $Nc \times n$ filters units.
7: **for** $row \leftarrow 1$ to r **do**
8: **for** $col \leftarrow 1$ to n **do**
9: $E_{row} \leftarrow [row, row + (N_C - r + 1)]$
10: $E_{col} \leftarrow [col, col + (N_C - n + 1)]$ ▷ Identify vertex pairs represented in row and col of the filter units.
11: **if** $E_{row}[0] \notin E_{col}$ or $E_{row}[1] \notin E_{col}$ **then**
12: fMat$[i, j] \leftarrow 0$
13: **end if**
14: **end for**
15: **end for**
16: fMat $\leftarrow$ Transpose(fMat) & fMat ▷ Mirror the matrix diagonally.
17: **end for**
18: RowMat $\leftarrow$ vstack(fMat) ▷ Vertically stack all fMats.
19: **end for**
20: FilterMat $\leftarrow$ hstack(RowMat) ▷ Horizontally stack all rowMats.
21: **return** FilterMat

3.5.2 Layer Architecture

The architecture can be seen as composed of two parts - the first part computes 3D vertices using an estimate of z-coordinates and the second part minimizes the SDA (minimization of standard deviation of angles) constraint by improving on the z-coordinate estimates as shown in Figure 10 and Figure 11.

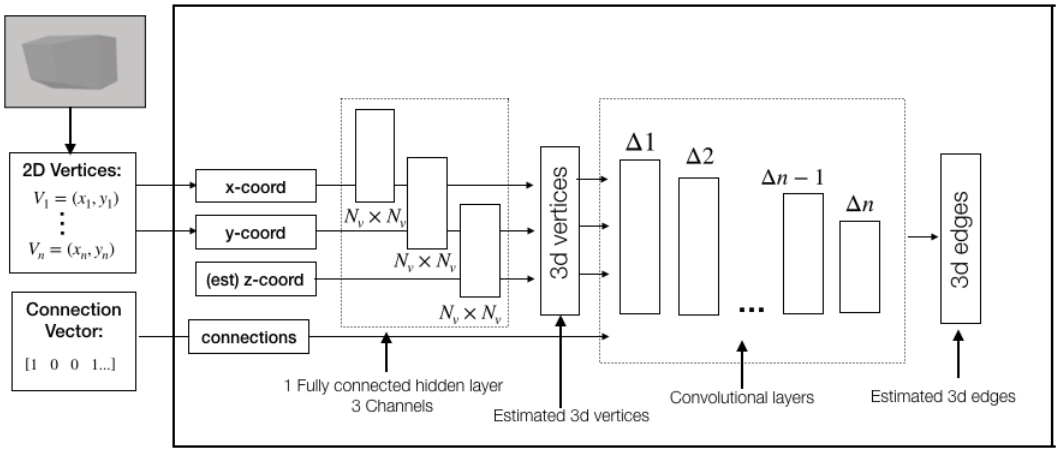


Figure 10. Model computes 3D edges from 2D inputs consisting of: 1) (x,y) coordinates of the vertices 2) a vector encoding connection between vertices that are connected via an edge

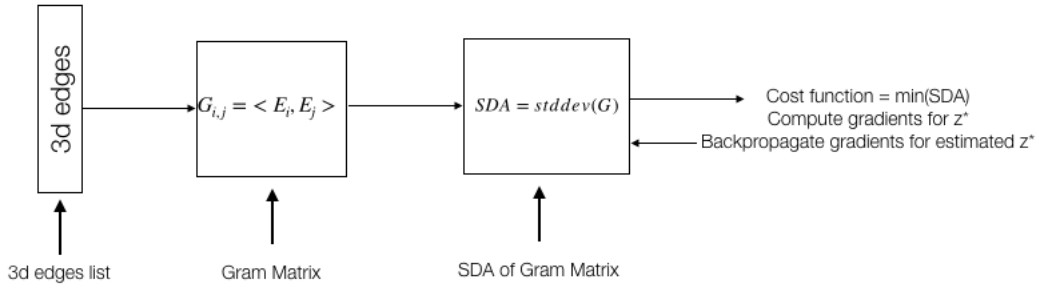


Figure 11. Model computes Gram Matrix and minimizes the SDA

3.5.3 Z parameter extraction

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 12, 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 12).

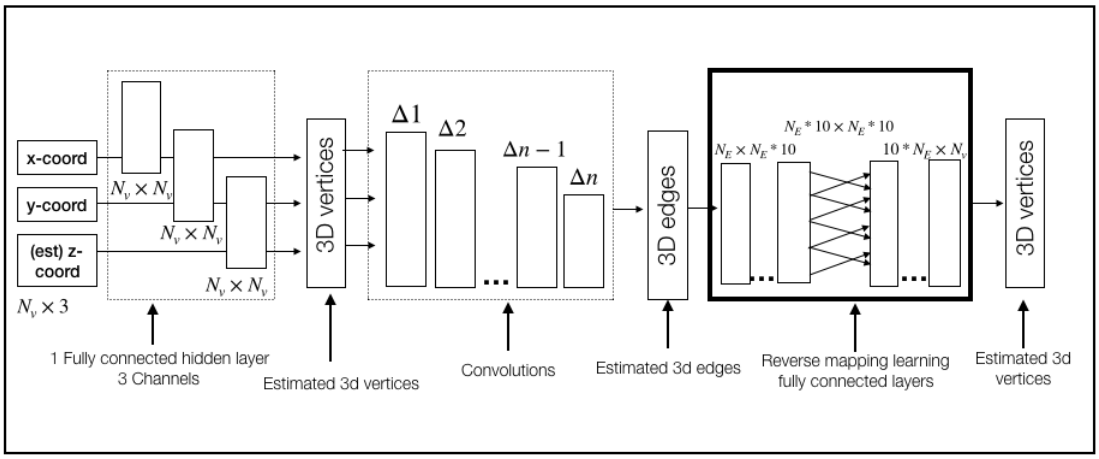


Figure 12. Complete network model to additionally compute missing z parameters using fully connected layers. The reverse mapping layers are highlighted using a darker box frame within the Figure.

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-coordinates configuration. This means that as the network trains on minimizing the SDA constraint, it needs to simultaneously learn the reverse mapping for each of the training example. Since our training is based on stochastic gradient descent method, the model has two parallel and nested learning paths for each set of training example in each epoch as shown in Figure 13.

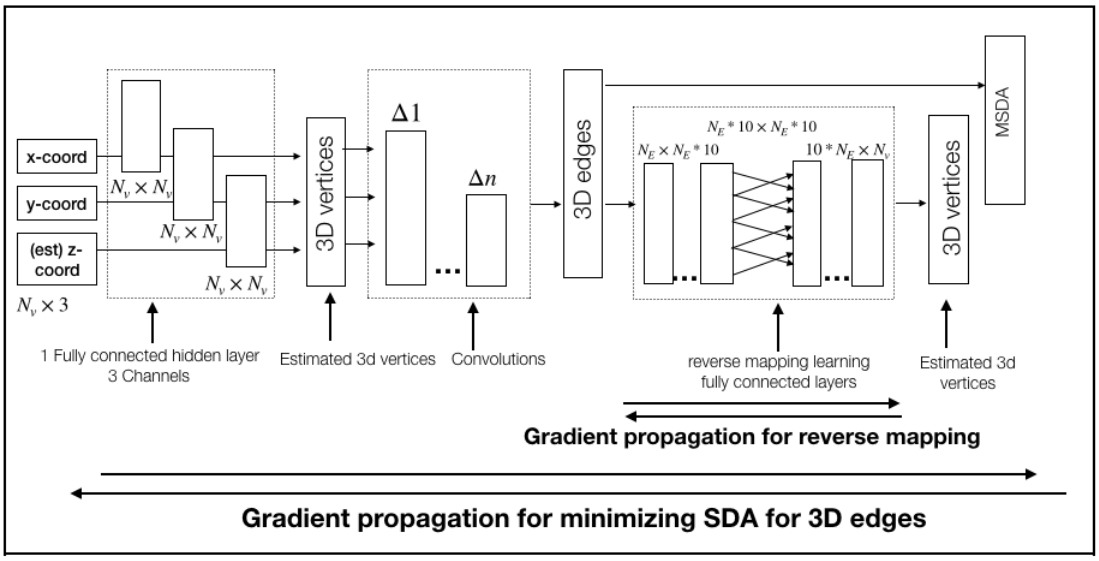


Figure 13. The model has two separate learning mechanisms in the same network. Here, MSDA represents the part of the network the computes the Gram matrix and minimizes the standard deviation of the matrix.

3.6 Summary

In this chapter, a high-level overview of the model and its architecture is presented. This model was trained to estimate missing depth from a set of 2D training stimuli. The stimuli generation process is discussed in the next chapter. The model was finally tested on the same stimuli that are used in the experiment. The model results are presented in Chapter 7.

CHAPTER 4. STIMULUS GENERATION

The shape perception experiment requires 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 were 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 case of structured 3D objects but not in the case 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 constraint (such as symmetry). Therefore all the 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 are presented in the Appendix Section I.2. 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 previous experiment. Some examples of objects in the pilot test that failed to be recovered above chance level are shown in Appendix Section I.3. It was observed from the pilot tests that objects with fewer number of vertices were more difficult to recover. Based on this finding, sufficiently complex but regular and novel set of shapes were created. These set of shapes were then divided into Blocks based on their level of complexity for the final version of the experiment. The set of stimuli used for training and testing the model and those used to test human subjects were the same. This requirement was imposed in order to make direct comparison between the performance of the model and the experiment outcome.

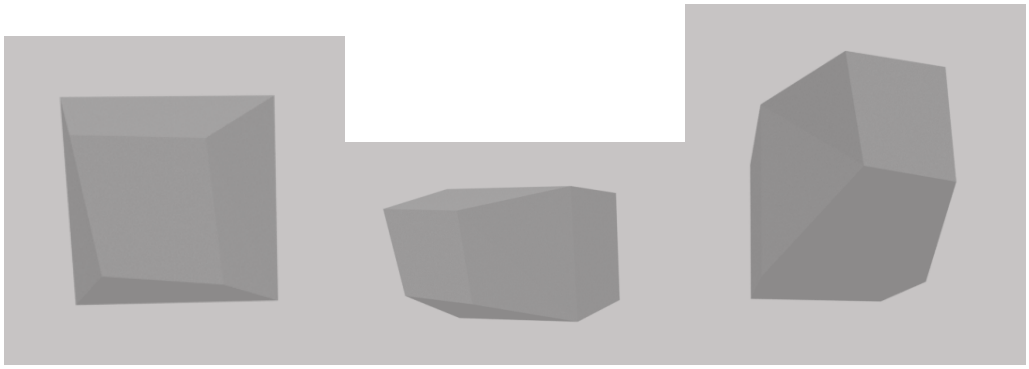


Figure 14. Stimuli example: Same object shown from 3 different projection viewpoints

An open source 3D graphic rendering tool called Blender was used to create 3D models of the novel structured stimuli objects. Since this 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 14. 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 is created programatically by applying a set of transformations on an original cuboid object. The set of transformations applied in order are:

1. Randomization: This transformation operation randomly displaces 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.

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 15. Table showing parameters used to generate objects used in Experiment Blocks.

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 opertation takes a minimum distance parameter to enforce symmetry from the central pivot point.

Novel, partly symmetric and structured objects are created from a cuboid by choosing the amount of randomization and pivot points and merging distances for mirroring and symmetry operation. In general, fewer randomization operations lead to simpler shapes. However, 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 objects used in the experiment blocks and configuration parameters for each of them is presented in Figure 15.

The obtained stimulus object from these operations is then rotated a fixed number of times on 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. The code used to generate our stimuli is made available at `https://palmishr.github.io/3DStimuliBlender/`.

CHAPTER 5. EXPERIMENT

5.1 Experiment objectives

A shape constancy experiment was designed to estimate consistent shape perception from a group of human participants in order to test the following:

1. Isolate and identify cases where human subjects can perceive shapes of novel stimuli consistently.
2. Isolate and identify cases where our model succeeds in achieving a consistent 3D shape estimation measured by angular estimation on same stimuli with different rotations.
3. Compare the outcomes of points 1 and 2 - are the success and failure cases of shape perception between human subjects and our model's estimates similar?

5.2 Experiment details

Stimuli: The previous chapter discussed the structure of the stimuli in detail. Appendix Section I.2 shows all the stimuli used in the experiment. Each stimulus object was rendered from eight different projection viewpoints. The set of these eight projections were used in the experiment.

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. Twenty of them took the experiment for credit and five of them 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: The total number of blocks in the experiment were seven. Blocks were designed in a way that they 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. There were unequal number of objects in blocks to rule out the case where subjects can only discriminate between objects at a time but do not perceive them uniquely. All blocks except Block 7 contain similar but distinct shapes. Block 7 contains objects from other blocks (2, 3 and 4). Since Block 7 has objects of dissimilar shapes, it was used to test whether subjects only discriminated between objects instead of perceiving them individually. In case the former is true, the performance of this block should be above all the other blocks.

Task: Within each block, every 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

All stimuli were symmetrical in 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 in Y and Z axis each (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. Figure 16 shows all stimuli shown to subjects in Block 4. A total of eight stimuli are created by four rotations of an object on Y and Z axis respectively. There were two objects in this particular block so the total stimuli presented was sixteen. Block 1 has three objects leading to a total of twenty-four stimuli as shown in Figure 17.

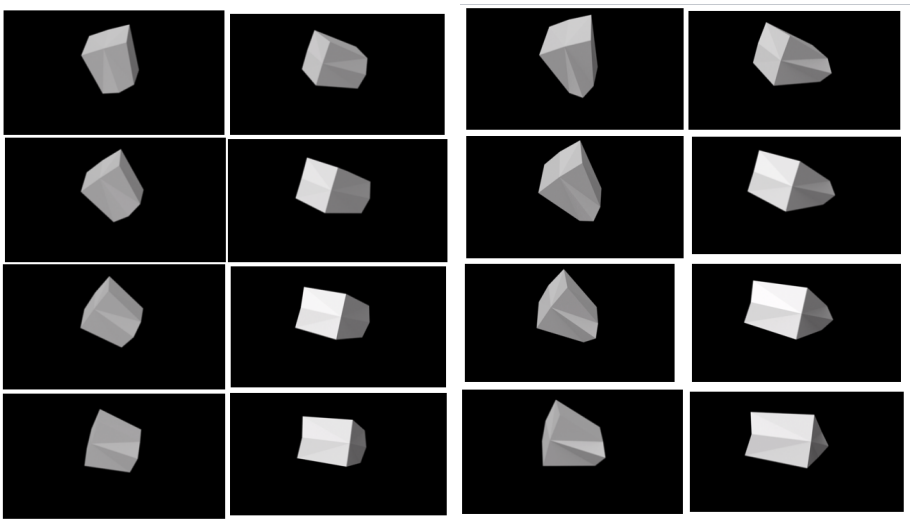


Figure 16. Block 4: Object 1 in left 8 images, Object 2 in right 8 images

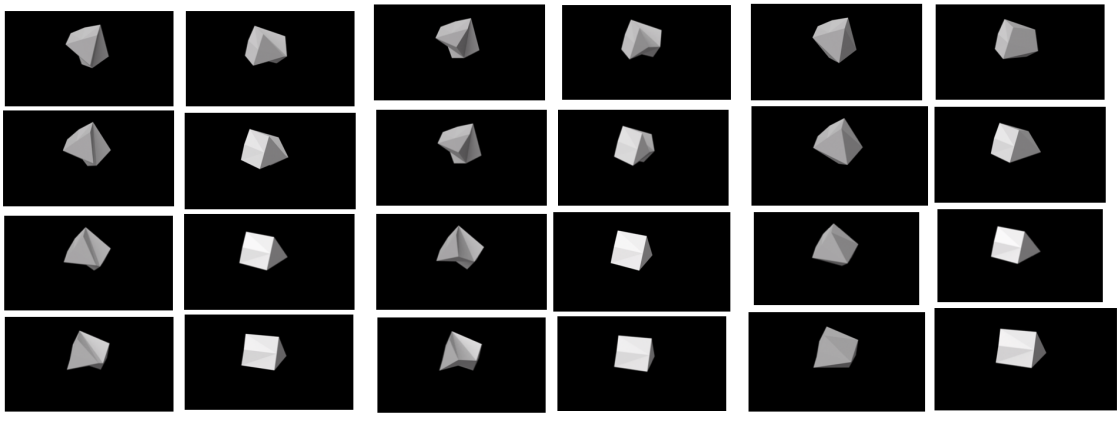


Figure 17. Block 1: Object 1 in left 8 images, Object 2 in middle 8 images, Object 3 in right 8 images

CHAPTER 6. EXPERIMENT RESULTS

For each participant, 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 answered 'yes' then a correct response was recorded. If the two stimuli were different objects and the subject answered 'no' then also, a correct response was recorded. In other cases, an incorrect response was recorded.

The first step of the analysis was to select 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%. The accuracy cutoff calculated using this binomial distribution was found to be 55%. The accuracy of response 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 experiment was their accuracy on all the objects combined. In pilot tests, it was noted that engaged subjects could perform considerably above chance (up to 80% accuracy overall). Out of total twenty-five subjects, five were eliminated based on this cutoff.

It is to be noted that the experiment was not designed to measure the perceptual reconstruction of the stimulus, only whether shape constancy is achieved when accuracy of response is above chance. A suitable method to compare experiment outcome against the model was to compare their 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 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.

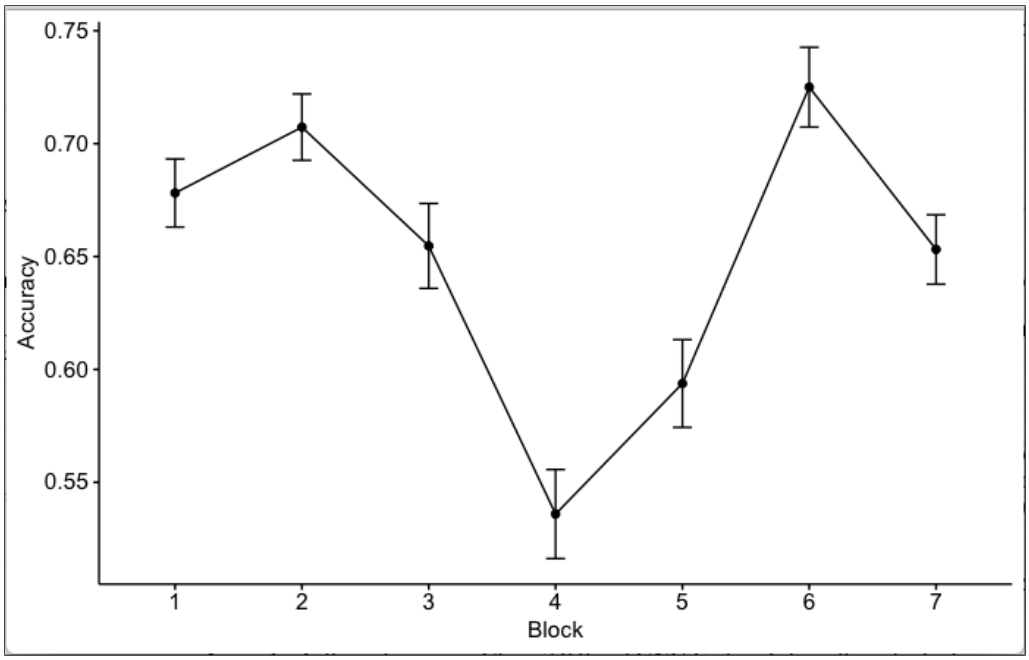


Figure 18. Experiment result: Plot showing the average accuracy statistics for each block.

6.1 Accuracy Analysis

The next step was to determine whether the accuracy of performance on the task as measured by correct (or incorrect) response per trial was affected by factors specifically, the block and the rotation angle. The overall accuracy per block is shown in Figure 18. In order to test for the effect of block and rotation angle in Y and Z axis on the binary response outcome (correct or incorrect), a generalized linear model based on maximum likelihood estimate was fit to the data. The generalized linear mixed affects model used a logit link function for the binomial distributed dependent variable. There were three generalized linear models fit to the data. The first one 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 predictor 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 the block and the rotation angle on response accuracy was carried out by comparing the two corresponding nested models one with and other without these predictors. The results of the significance test on blocks is presented in Table 2. It was observed that blocks had a significant effect on the outcome (correct or incorrect) using analysis of deviance statistic $(Chisq(6) = 79.49, p < 0.001)$. The effect of rotation on either 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 estimates (beta) for Y and Z versus the angle of rotation. The plots are presented in Figure 19 show the relationship between the betas for Y and Z and the angle of rotation for each ($R^2 = 0.95$ for Y and $R^2 = 0.97$ for Z). It may be concluded that as the rotation in Y and Z axis increases, the accuracy of performance in the task decreases.

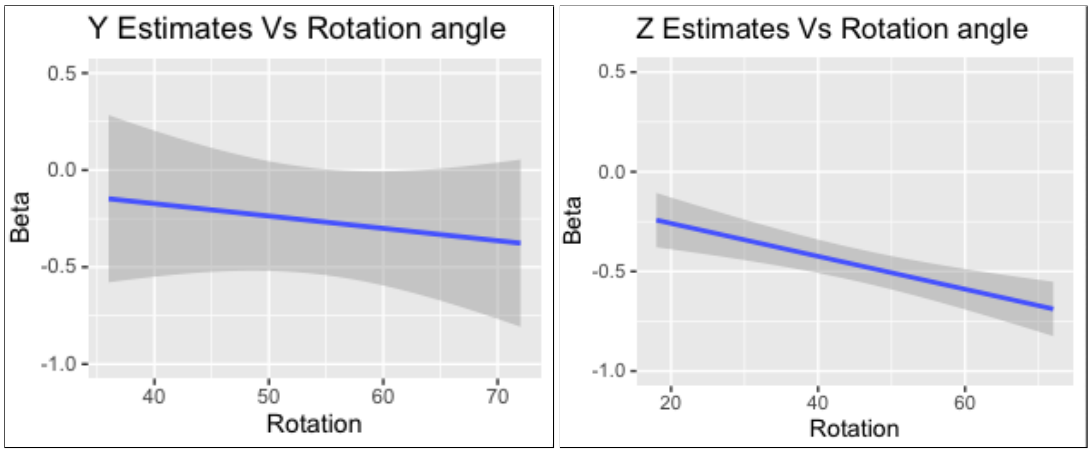


Figure 19. Experiment result: Plot showing the relationship between Y and Z rotation angles on GLMz model estimates for Y and Z respectively.

The accuracy for individual objects within each block is presented in Appendix Section I.4. Since the effect of blocks was found to be significant, not all blocks should have the same accuracy. Based on this result, Block 4 and Block 5 can be categorized as higher difficulty. Block 1, 2, 3 and 6 are easier blocks. As a reminder, Block 4 contains objects with the least complexity i.e. fewer number of vertices and simpler shapes compared to Block 1, 2 and 6. This suggests that it may be easier to consistently perceive the shape of a more complex, structured object compared to a less complex or less structured object. It should be noted that Block 7 is not categorized because it was used to test whether the novel objects were uniquely perceived on their own or not. Since Block 7 did not outperform other blocks, there was no evidence to conclude that comparing more distinct objects made the task substantially easier for subjects. It can therefore be ruled out that this task was only a discrimination based task rather than a perceptual task.

Table 1. *Experiment result: Tables showing the results from Generalized Linear Model. 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. *Experiment result: The ANOVA table is displayed for the block significance test by comparing two nested models, one with blocks as predictor of response accuracy and other without blocks as the 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***							

Table 3. *Experiment result: The ANOVA table is displayed for the rotation significance test by comparing two nested models, one with rotation (4 rotations on Y axis and 4 rotations of Z axis) as predictor of response accuracy and other without rotation as the predictor of response accuracy.*

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

6.2 Signal Detection Theory Analysis

The discrimination sensitivity measure related to the performance within each block is shown in Figure 20 where

$$d' = [z(H) - z(F)]$$

H is proportion of hits, F is proportion of false alarms. $z(H)$ denotes the z score for fraction of hits and $z(F)$ denotes the z scores for fraction of misses. Since higher discriminability should lead to higher accuracy on 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 are the same.

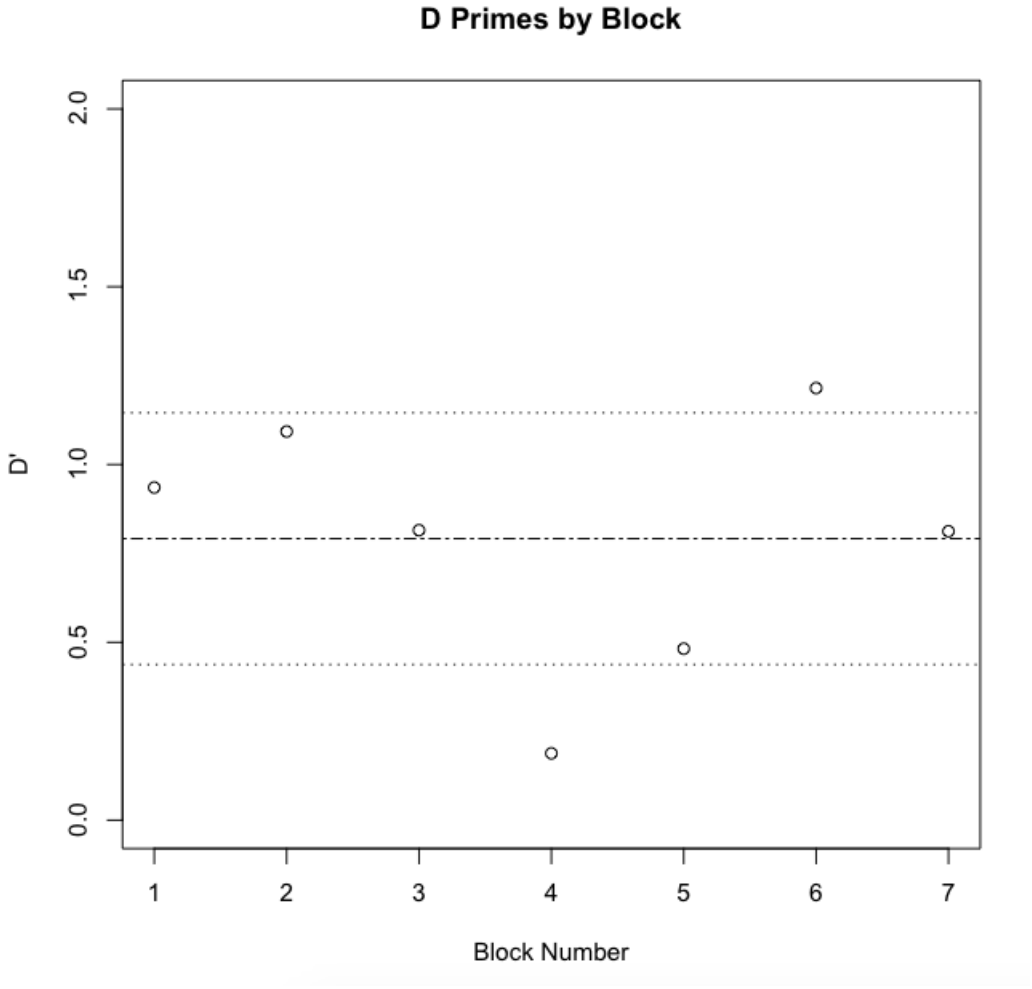


Figure 20. Experiment result: Plot showing the average discriminability for all objects within each block.

Due to the nature of the task involving a forced choice a 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 stimuli presented were different objects 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:

$$C = -[z(H) + z(F)]/2 \tag{6.1}$$

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 bias for each block is shown in Figure 21. A sign test on the criterion locations for blocks (s = 2, p-value = 0.4531) reveals that there is no evidence for bias being different than zero. The 95% confidence interval for location of criterion is (-0.2627309, 0.2174412). A similar test for location of criterion for all valid test subjects (s = 10, p-value = 1) revealed that there was no evidence of bias being different than zero as well.

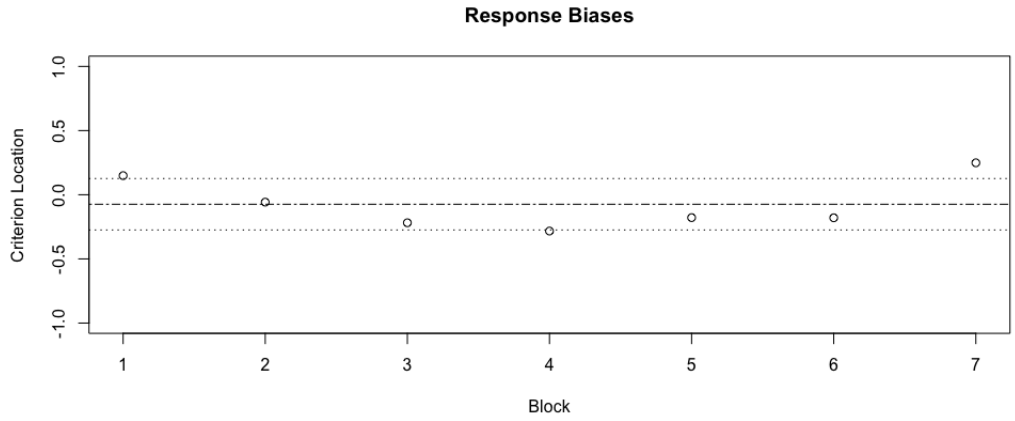


Figure 21. Experiment result: Plot showing the response bias measured in terms of criterion location for each block.

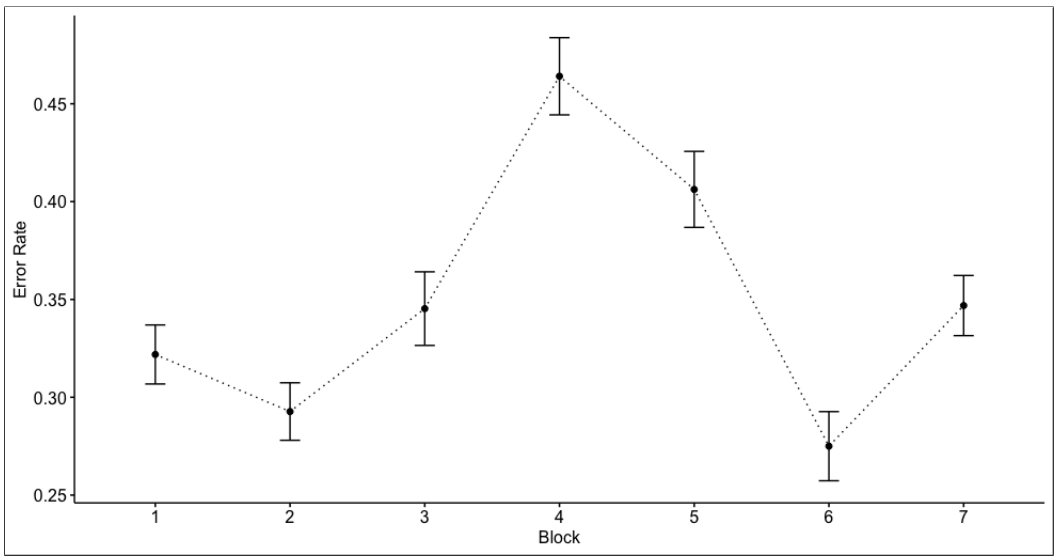


Figure 22. Experiment result: Plot showing the average error rate (measured in terms of incorrect responses) statistics for each block.

6.3 Summary

In this Chapter, the block-wise performance in the experiment was analyzed by two separate measures. First, by aggregating the overall accuracy of response for all subjects within that block. The second analysis is done using sensitivity as measured by D' using signal detection theory. Figure 20 shows the aggregate discriminability of all stimuli within each block. Both analysis lead to the same ordering of blocks - Blocks 6, 2, 1 and 3 show better performance (in that order) than Blocks 4 and 5. This result was then used to compare the model performance in the following Chapter. In order to facilitate this comparison, a conversion from average accuracy to average error rate was found to be useful. The error rate plot is shown in Figure 22.

It has been shown that the blocks have a significant effect on the performance of the task as different blocks contain a different level of object complexity. The performance of the model on blocks would be measured using consistency of 3D angle reconstruction for each stimulus in the block from different viewpoints. The same rotations of the object would be used to test the model as in the experiment. The comparison of the model's performance on different blocks should help answer questions posed in Chapter 5. Therefore the order of blocks based on the performance of subjects in each block is a crucial result to compare the model performance. If the model and the experiment outcome agree on the shapes that are more consistently perceived than others, then the model achieves the objective of using the constraint of MSDA to recover shapes. At the same time, if the model's performance on shapes is inconsistent, then perhaps the constraint of MSDA is not enough to recover those shapes.

CHAPTER 7. MODEL RESULTS

The model computes a Gram matrix using an estimate of z values that minimizes the standard deviation of all 3D angles in the reconstructed shape. The experiment on the other hand measures 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 is measured by quantifying the similarity in the 3D angles estimated for 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 per stimulus. Since the network learns to minimize the SDA value, the error rate of the network is measured in terms of the mean SDA value per batch of input. The network was then tested on the new set of unseen stimuli. The table shown in Figure 23 shows the configuration parameters used to generate training samples for the model. The average SDA as measured by the stimuli rendering software was around 0.03 for the training and test stimuli set. In the training phase, the model starts with a high error rate of 0.2 then gradually goes to 0.01 over 10 epochs of training with 1000 stimuli with the batch size of 5. Figure 24 shows the graph of network error rate over a training sequence of first 1000 objects.

Stimlus Type	Size	Number of Randomizations	Cursor Position	Subdivide	Merge Distance Mirror	Merge Distance Symmetry
Training	random(0,1)	random(2,20)	(0,0,0)	FALSE	0.1	0.1
Test	random(0,1)	random(2,20)	(0,0,0)	FALSE	0.1	0.1

Figure 23. Table showing parameters used to generate objects used in training and testing phase of the model.

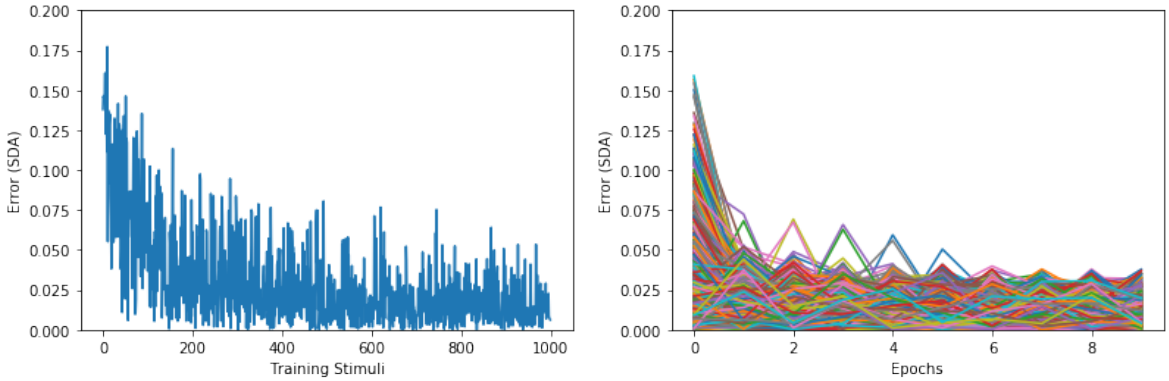


Figure 24. Top: Error plot during the first epoch of training samples; Bottom: Error rate showing model minimizing SDA values during 10 epochs of 1000 training samples each.

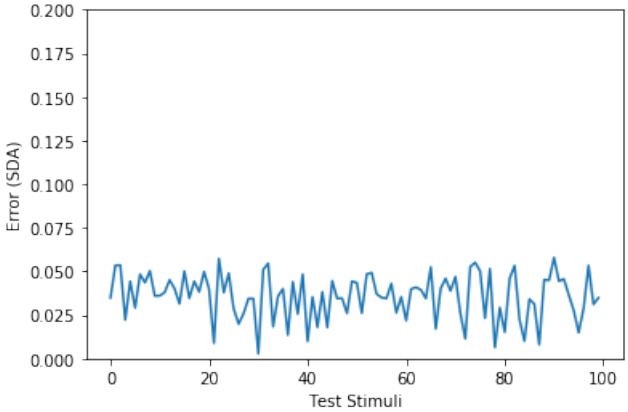


Figure 25. Model's performance for 100 unseen randomly generated cuboid based stimuli.

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 rotation of the experiment object was presented to the model for comparison. The model estimated 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 matrix for a rotated view and the original view is computed. Since there is no access to the perceived 3D shapes from the experiment, this metric helps in comparing the model's performance and the experiment outcome. 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 signifies the extent to which estimated 3D shapes are deviant 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.

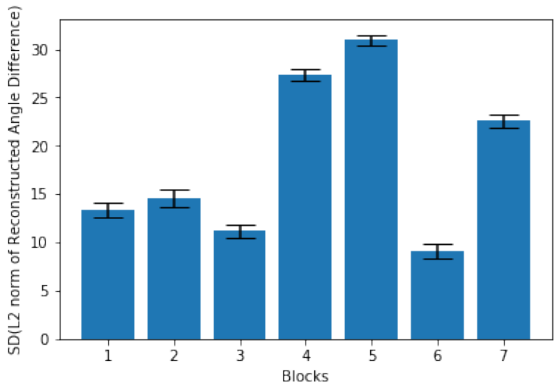


Figure 26. Performance of the model on the experiment blocks - the best performing block is Block 6, Block 1 and Block 2 while the worst performing blocks are Block 4 and Block 5. The error bars denote the standard deviation of the computed metric across 20 different simulations of the trained model.

7.1 Summary

The performance of the model corroborated that with the results from the experiment as shown in Figure 26. The criteria of success was proposed to be how closely the reconstruction consistency from 2D input by the model matches the ordering of block difficulty in human subjects. The analysis of experiment outcome and the output from the model show similar results. As for the experiment, the blocks which 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 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 as well as the model, the order of performance is 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 uses only one constraint. However, the constraint that the model is used shows considerable effectiveness in modeling human performance.

CHAPTER 8. SUMMARY AND CONCLUSION

The goal of this research was to develop a biologically principled network for 3D perception of object shape from 2D inputs. The architecture of the network was inspired in part by the ideas derived from the computational structure of visual areas such as the V4 and the striate cortex. 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 in the estimated 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 on the objects from the experiment, it may be concluded that the model may predict shape constancy for 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's shape. Since the model and subjects from the experiment fail on the same type of stimulus (at the block level), the analysis of these failures show that the MSDA constraint is effective for a reliable shape perception. The similarity of outcome of the model with the experiment results shows that a network based model can implement visual constraint of MSDA. The results also demonstrate a proof of concept for this biologically inspired network to compute the required constraint.

The results presented in this work are based on the validity of the chosen metric to test model performance. Since the perceived 3D structure by human subjects is never available for comparison, the model was tested for shape constancy separately. In general shape comparison is a computationally difficult problem to solve (Biasotti, Cerri, Bronstein, & Bronstein, 2014), comparing the z coordinate estimate of the model with the actual shape was out of scope for this work. 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 about how networks can achieve 3D shape recovery using psychophysical principles.

8.1 Future Extensions

There are several other constraints as discussed in Chapter 1 that have been found to be useful for consistent 3D perception. A future extension of the model can be adding these constraints into the network to work in tandem with the existing MSDA layer. An important constraint besides MSDA is symmetry. The determination of plane of symmetry and measurement of variation of symmetry from the determined symmetry plane is a complex problem. Since MSDA is a mechanism to increase symmetry of the object in general, one can re-purpose another measure from shape analysis called shape circularity in place of symmetry. The advantage of using this measure is that inherently this measure maximizes the number of symmetry planes for a 3D object i.e. for a perfect sphere with infinite number of symmetry planes, circularity measure is maximum. This may be a sufficiently good measure of overall symmetry in 3D object. The constraint of compactness can also be incorporated into the network.

Montero and Bribiesca (2009) describe how shape circularity and shape compactness may be measured for pixelated digital objects. In order to measure compactness, one needs to estimate volume to surface area ratio. Compactness has been used extensively in several domains of engineering and psychophysics to describe shape in shape analysis tasks. It has been associated with ratio of $(perimeter)^2/area$. In fact 3D shape compactness is the same concept extended in third dimension. There have been several ways to calculate compactness on a 2D regions and most of them can be extended for 3D shapes but in order to keep the formulation as simple as possible mathematically, a method called normalized discreet compactness (Bribiesca, 2000) that has been successfully applied to 3D shapes. Another important consideration in choosing this method of computing compactness is that this measure is invariant under translation, rotation and scaling. A mathematical description of measuring compactness and symmetry based on these ideas is presented in Appendix Section I.5.

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 into the network sheds more light on how the human visual systems uses build-in constraints to understand our three-dimensional environment.

REFERENCES

Aloimonos, J. (1988). Shape from texture. *Biological cybernetics*, *58*(5), 345–360.

Biasotti, S., Cerri, A., Bronstein, A. M., & Bronstein, M. M. (2014). Quantifying 3d shape similarity using maps: Recent trends, applications and perspectives. In *Eurographics (state of the art reports)* (pp. 135–159).

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Bribiesca, E. (2000). A measure of compactness for 3d shapes. *Computers & Mathematics with Applications*, *40*(10-11), 1275–1284.

Brown, E., & Wang, P. S. (1996). Three-dimensional object recovery from two-dimensional images: a new approach. In *Intelligent robots and computer vision xv: Algorithms, techniques, active vision, and materials handling* (Vol. 2904, pp. 138–148).

Cadieu, C. F., Hong, H., Yamins, D. L., Pinto, N., Ardila, D., Solomon, E. A., ... DiCarlo, J. J. (2014). Deep neural networks rival the representation of primate it cortex for core visual object recognition. *PLoS computational biology*, *10*(12), 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*(3), 507–517.

Cavanagh, P. (1987). Reconstructing the third dimension: Interactions between color, texture, motion, binocular disparity, and shape. *Computer Vision, Graphics, and Image Processing*, *37*(2), 171–195.

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*(7), 1124–1139.

Clark, J., & Yuille, A. (1990). Shape from shading via the fusion of specular and lambertian image components. In *Pattern recognition, 1990. proceedings., 10th international conference on* (Vol. 1, pp. 88–92).

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*(3), 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.

Finnerty, J. R. (2005). Did internal transport, rather than directed locomotion, favor the evolution of bilateral symmetry in animals? *BioEssays*, *27*(11), 1174–1180.

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*(1), 119–136.

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

Haralick, R. M. (1974). A measure for circularity of digital figures. *IEEE Transactions on Systems, Man, and Cybernetics*(4), 394–396.

Khaligh-Razavi, S.-M., & Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain it cortical representation. *PLoS computational biology*, *10*(11), e1003915.

Landy, M. S., Maloney, L. T., Johnston, E. B., & Young, M. (1995). Measurement and modeling of depth cue combination: In defense of weak fusion. *Vision research*, *35*(3), 389–412.

Leclerc, Y. G., & Fischler, M. A. (1992). An optimization-based approach to the interpretation of single line drawings as 3d wire frames. *International Journal of Computer Vision*, *9*(2), 113–136.

Li, Y., Pizlo, Z., & Steinman, R. M. (2009a). A computational model that recovers the 3D shape of an object from a single 2D retinal representation. *Vision Research*, *49*(9), 979–991. Retrieved from `http://dx.doi.org/10.1016/j.visres.2008.05.013` doi: 10.1016/j.visres.2008.05.013

Li, Y., Pizlo, Z., & Steinman, R. M. (2009b). A computational model that recovers the 3d shape of an object from a single 2d retinal representation. *Vision research*, *49*(9), 979–991.

Liu, J., Cao, L., Li, Z., & Tang, X. (2008). Plane-based optimization for 3d object reconstruction from single line drawings. *IEEE Trans. Pattern Anal. Mach. Intell.*, *30*(2), 315–327.

Marill, T. (1991). Emulating the human interpretation of line-drawings as three-dimensional objects. *International Journal of Computer Vision*, *6*, 147-161.

Montero, R. S., & Bribiesca, E. (2009). State of the art of compactness and circularity measures. In *International mathematical forum* (Vol. 4, pp. 1305–1335).

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*(7), 2239–2255.

Pasupathy, A., & Connor, C. E. (2001). Shape representation in area v4: position-specific tuning for boundary conformation. *Journal of neurophysiology*, *86*(5), 2505–2519.

Pizlo. (2001a). Introduction : Uniqueness of Shape.

Pizlo. (2001b). Perception viewed as an inverse problem. *Vision research*, *41*(24), 3145–3161.

Pizlo, Sawada, T., Li, Y., Kropatsch, W. G., & Steinman, R. M. (2010). New approach to the perception of 3d shape based on veridicality, complexity, symmetry and volume. *Vision research*, *50*(1), 1–11.

Pizlo, & Stevenson. (1999). Shape constancy from novel views. *Perception & Psychophysics*, *61*(7), 1299–1307.

Poggio, T., & Koch, C. (1985). Ill-posed problems early vision: from computational theory to analogue networks. *Proc. R. Soc. Lond. B*, *226*(1244), 303–323.

Richards, W. (1985). Structure from stereo and motion. *JOSA A*, *2*(2), 343–349.

Roe, A. W., Chelazzi, L., Connor, C. E., Conway, B. R., Fujita, I., Gallant, J. L., ... Vanduffel, W. (2012). Toward a unified theory of visual area v4. *Neuron*, *74*(1), 12–29.

Schwartz, E. L. (1980). Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding. *Vision research*, *20*(8), 645–669.

Shoji, K., Kato, K., & Toyama, F. (2001). 3-d interpretation of single line drawings based on entropy minimization principle. In *Computer vision and pattern recognition, 2001. cvpr 2001. proceedings of the 2001 ieee computer society conference on* (Vol. 2, pp. II–II).

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

Ullman, S. (1979). The interpretation of structure from motion. *Proc. R. Soc. Lond. B*, *203*(1153), 405–426.

Vetter, P. (1994). Symmetric 3D shapes are an easy case for 2D object recognition.

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*(23), 8619–8624.

APPENDIX I. APPENDIX

I.1 Convolutional Layer Configuration

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

I.1.1 Configured Parameters:

Channels: 3

Different channels allow for parallel operations on the set of inputs. We have 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 layer. The filter or kernel is convolved with the input to produce the output. Since we work with a pair of vertices at a time to compute the edge vector, our 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 list of edges, successive convolution layer compute distance d dilation apart where d is the dilation value.

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 difference between the x, y and z coordinates in successive channels.

I.1.2 Mathematical Details

Figure 27 (b) shows the computation involved per batch for the convolution. Here W is the weight matrix associated with the layer. Figure 27 (c) visually depicts how the weight matrix is related to input and output channels based on the equation shown in part (b). The parameters of batch size, layer size, channels, kernel size and weight matrix are that are utilized in the convolution operation are individually described in Chapter 3 along with the values set for these in our model.

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

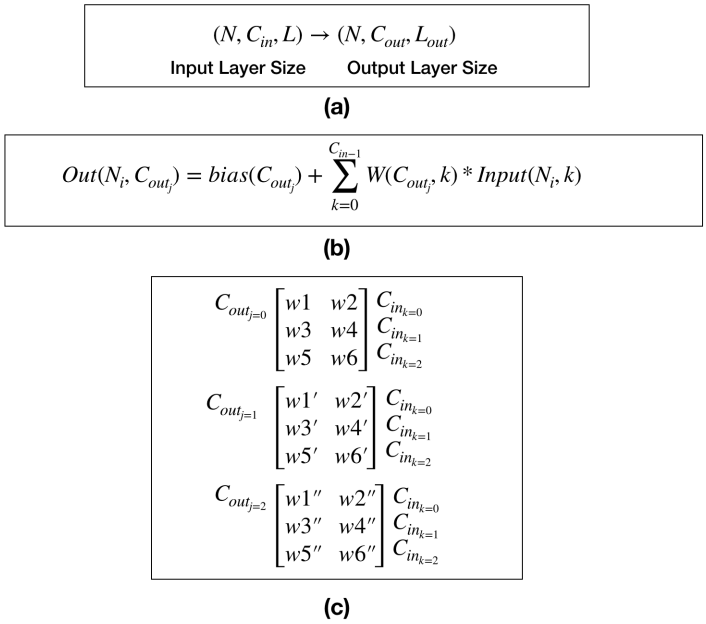


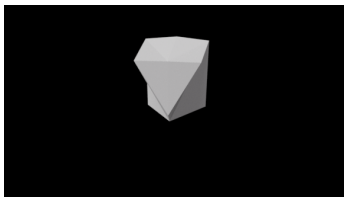
Figure 27. The relationship between input layer parameters to output layer parameters in Convolutional layer

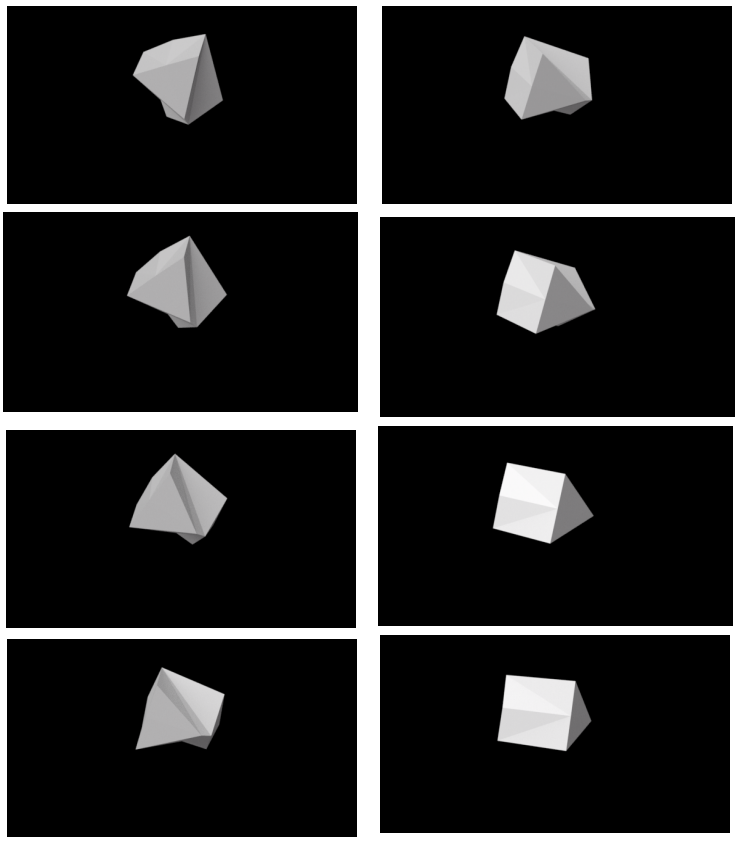
Figure 27 (a) shows the high level relationship between input and output layer size (L) of a convolution operation given the number of channels (C), number of batches processed (N). The way L_{out} is related to L_{in} is shown in Equation I.1.

I.2 Novel Stimuli

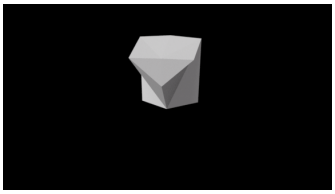
The novel, unfamiliar and structured stimuli objects are presented in this section.

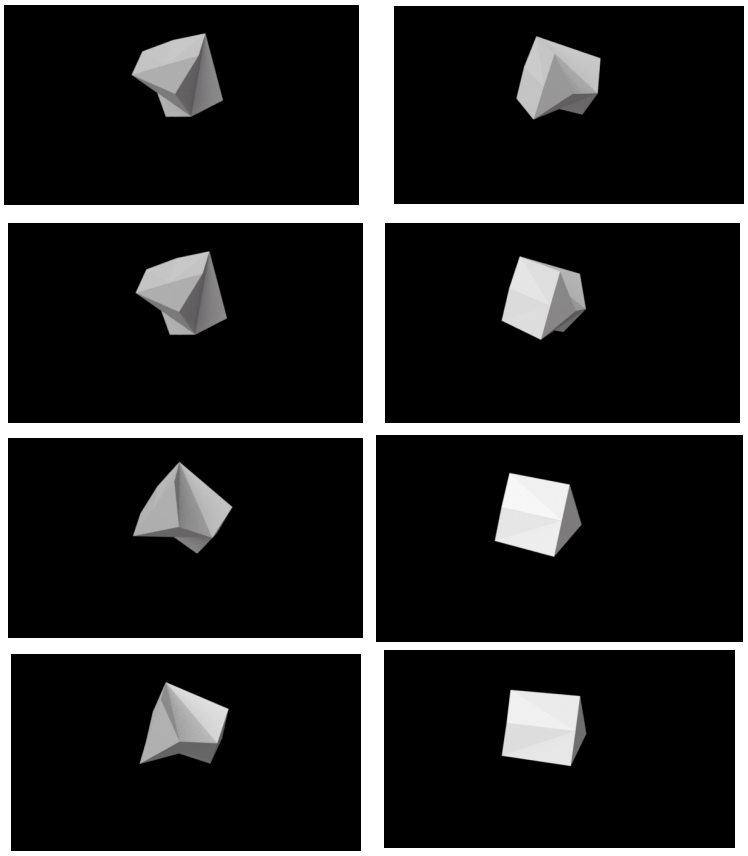
Stimulus Object 1



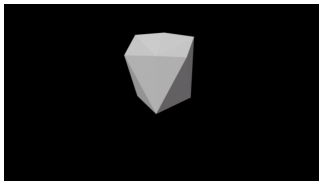


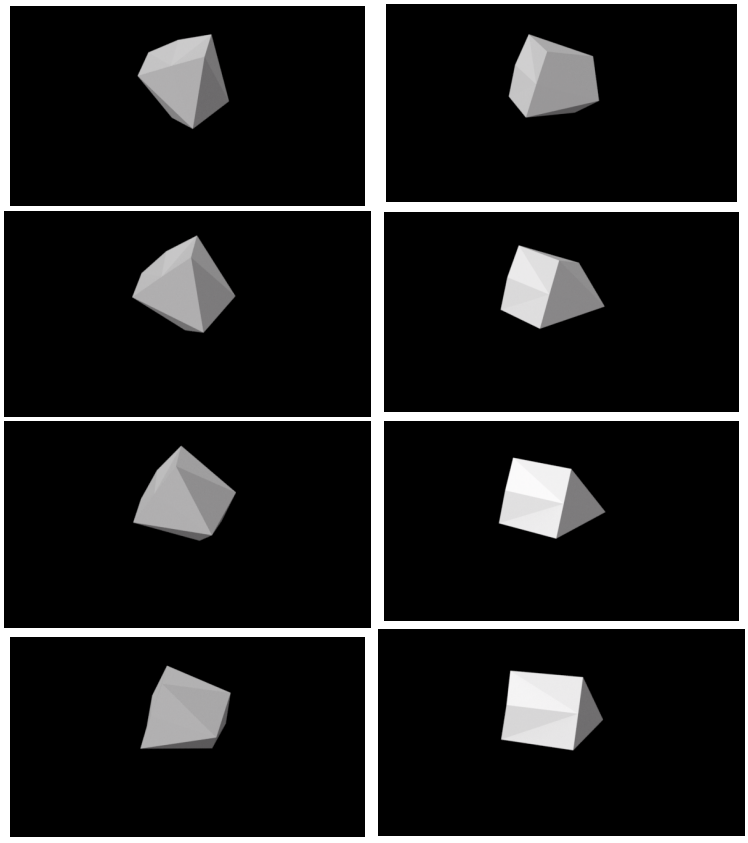
Stimulus Object 2



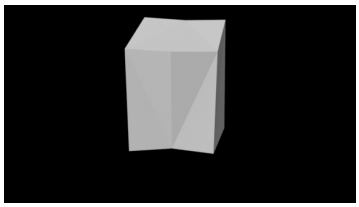


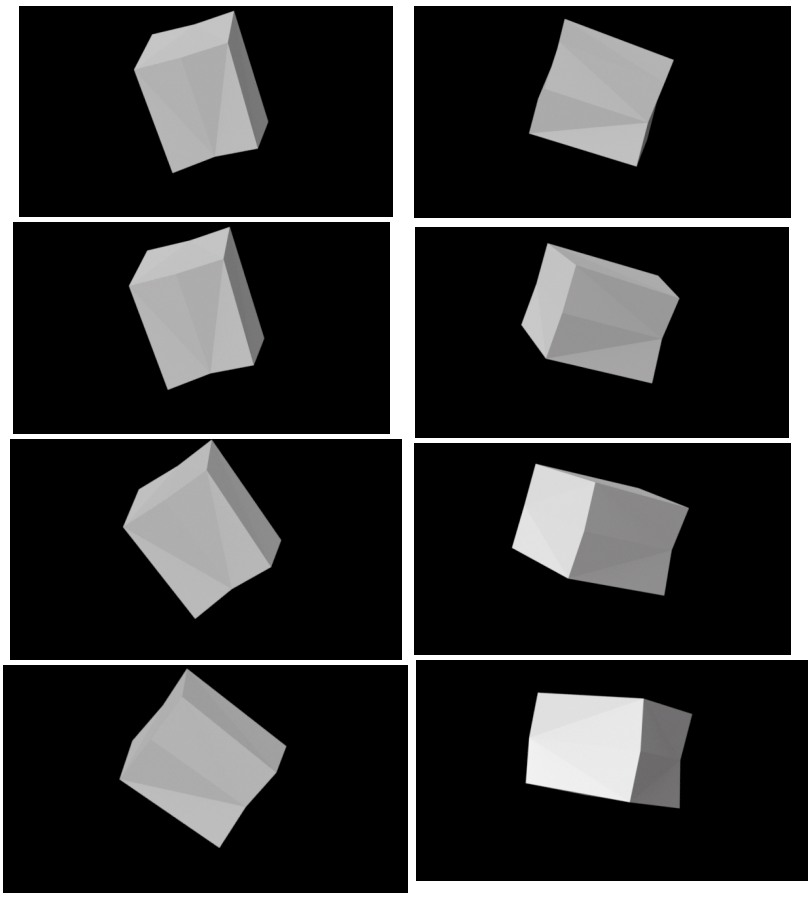
Stimulus Object 3



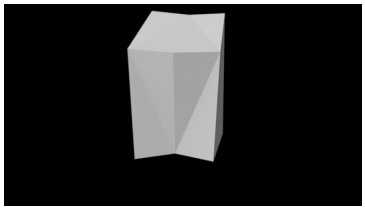


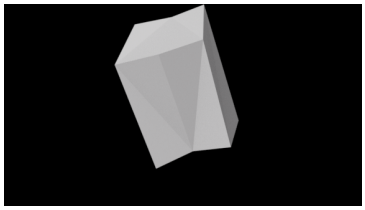
Stimulus Object 4

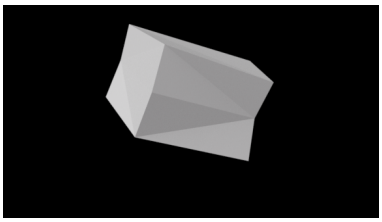


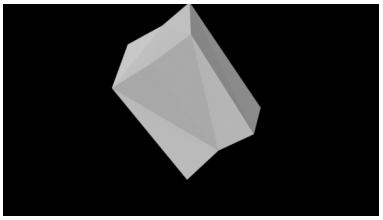


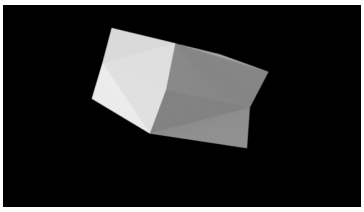
Stimulus Object 5

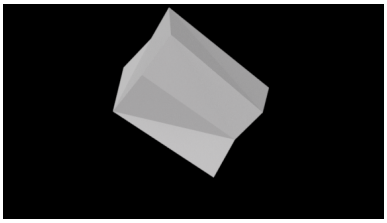


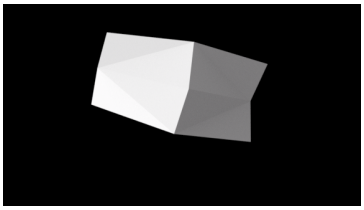


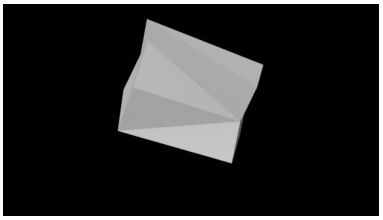


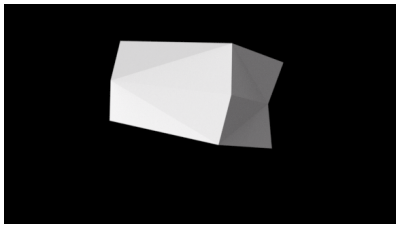




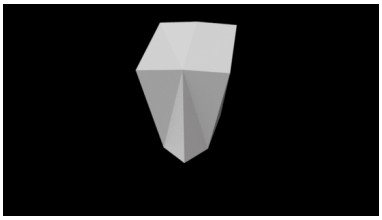


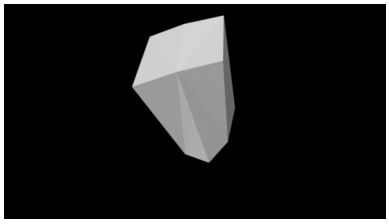


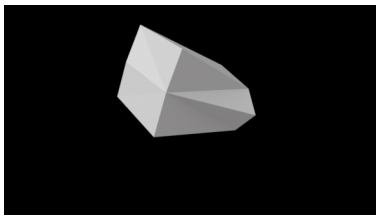


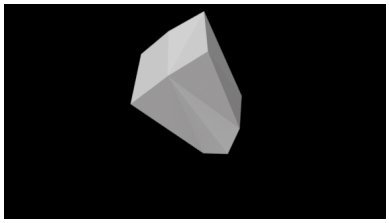


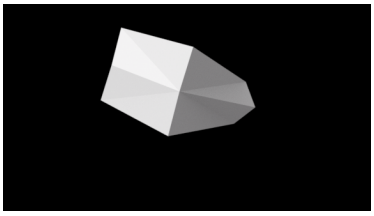
Stimulus Object 6

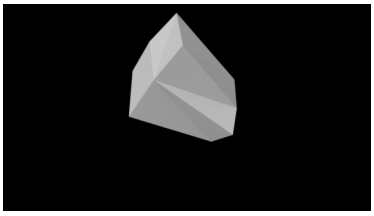


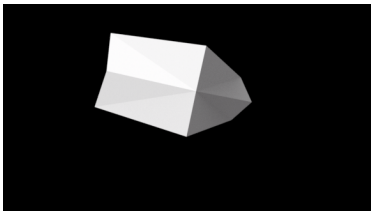


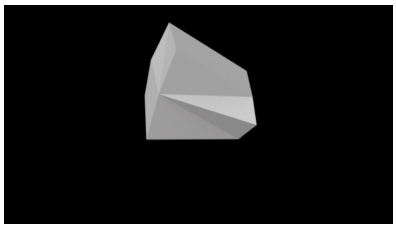


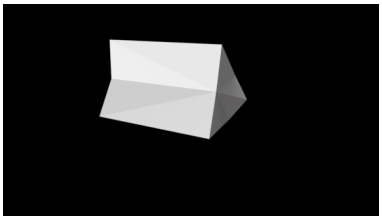




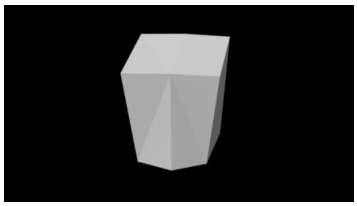


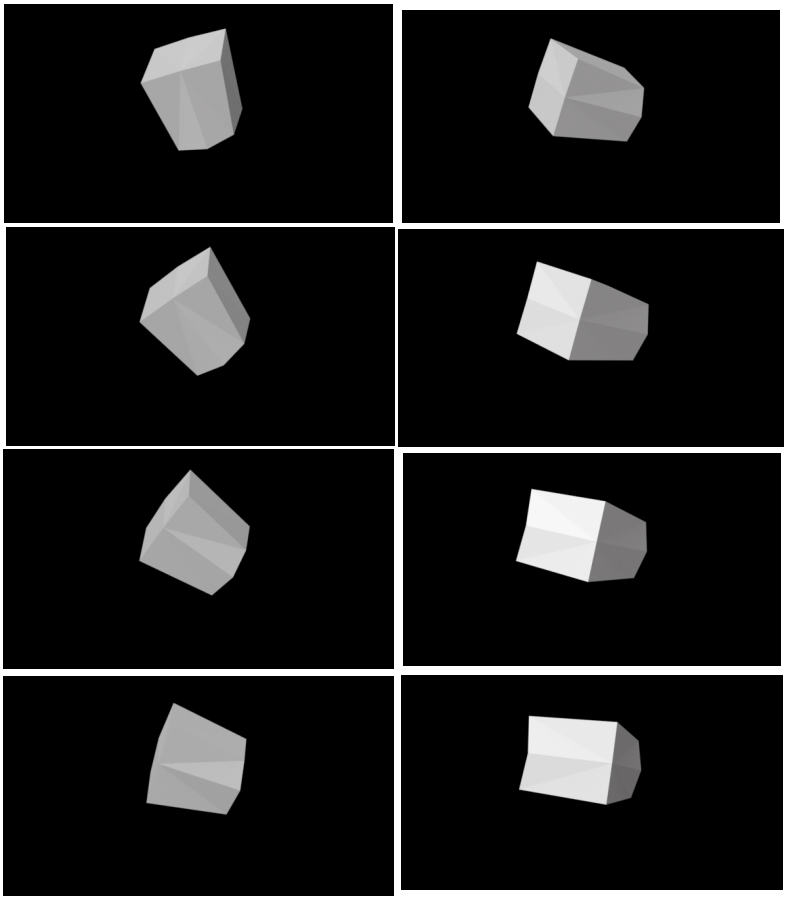




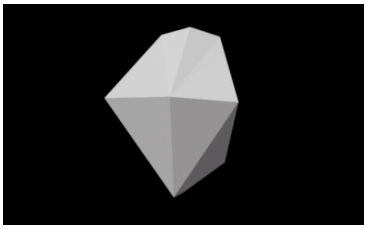


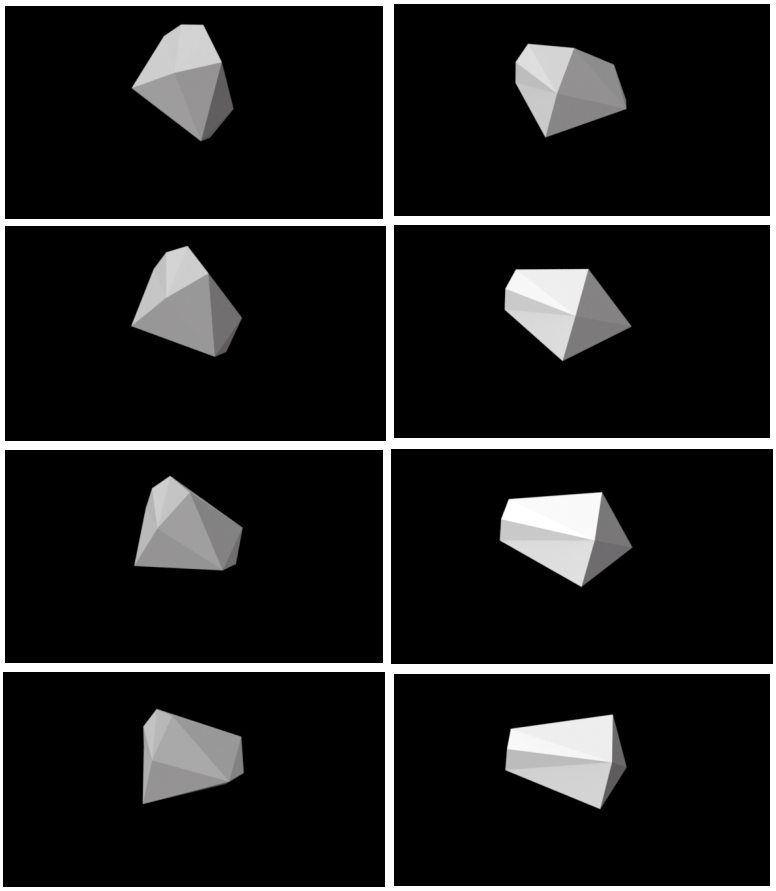
Stimulus Object 7



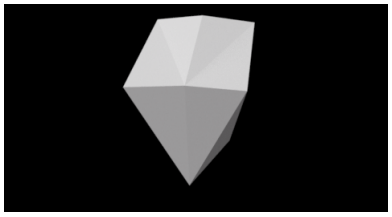


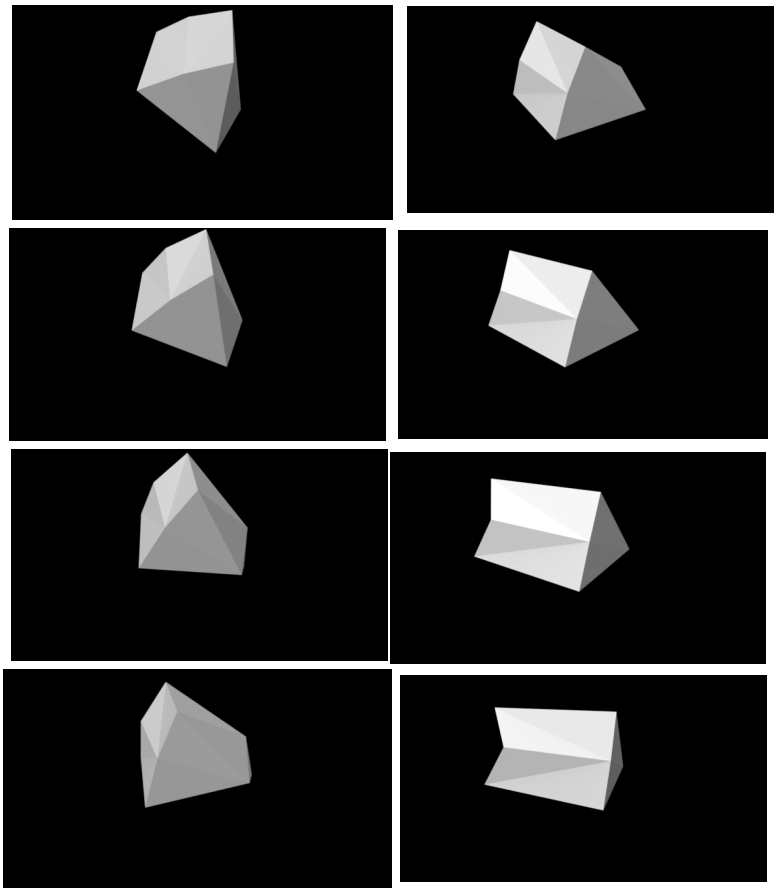
Stimulus Object 8



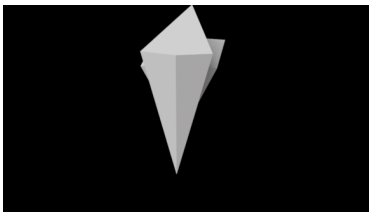


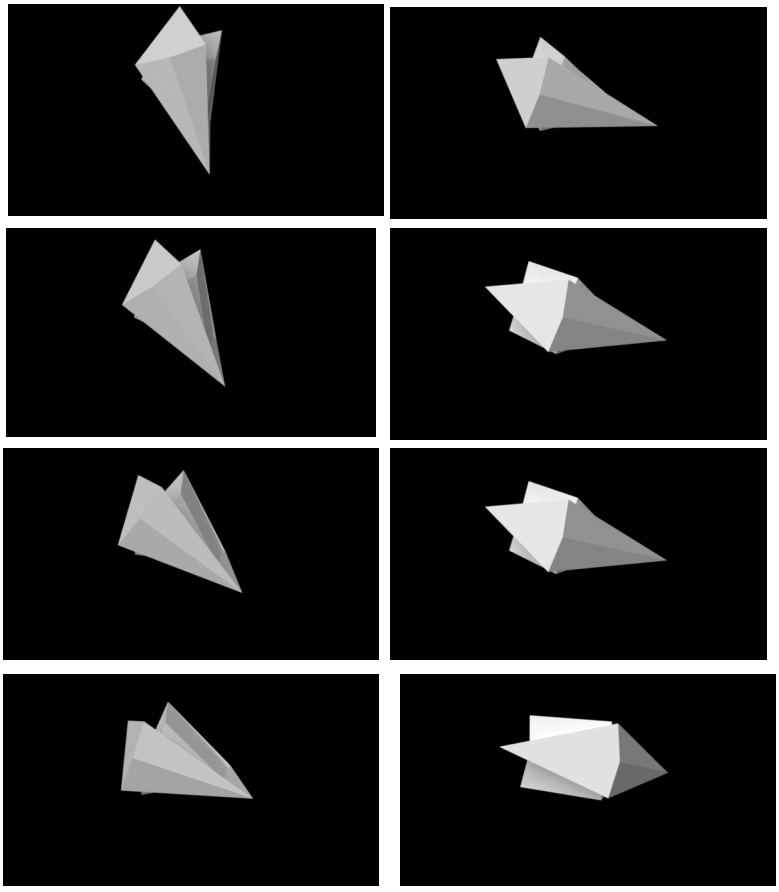
Stimulus Object 9



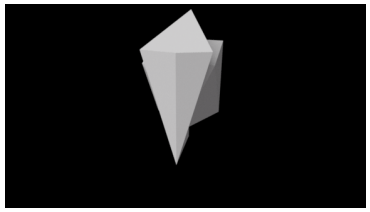


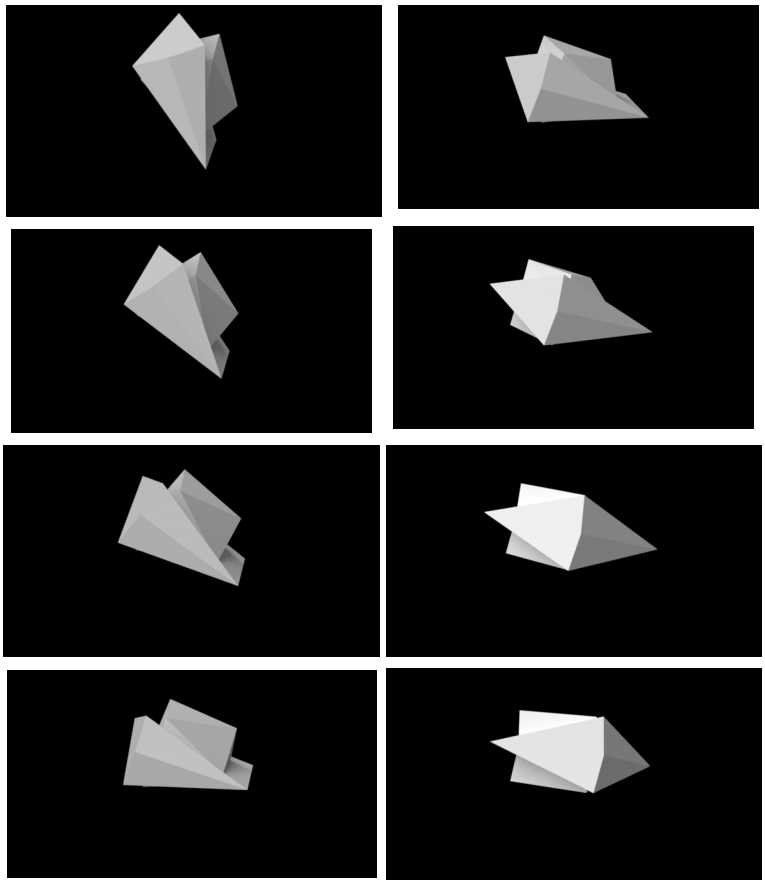
Stimulus Object 10



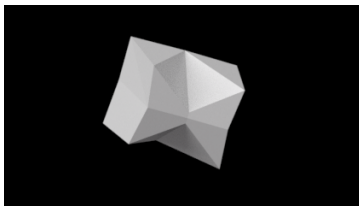


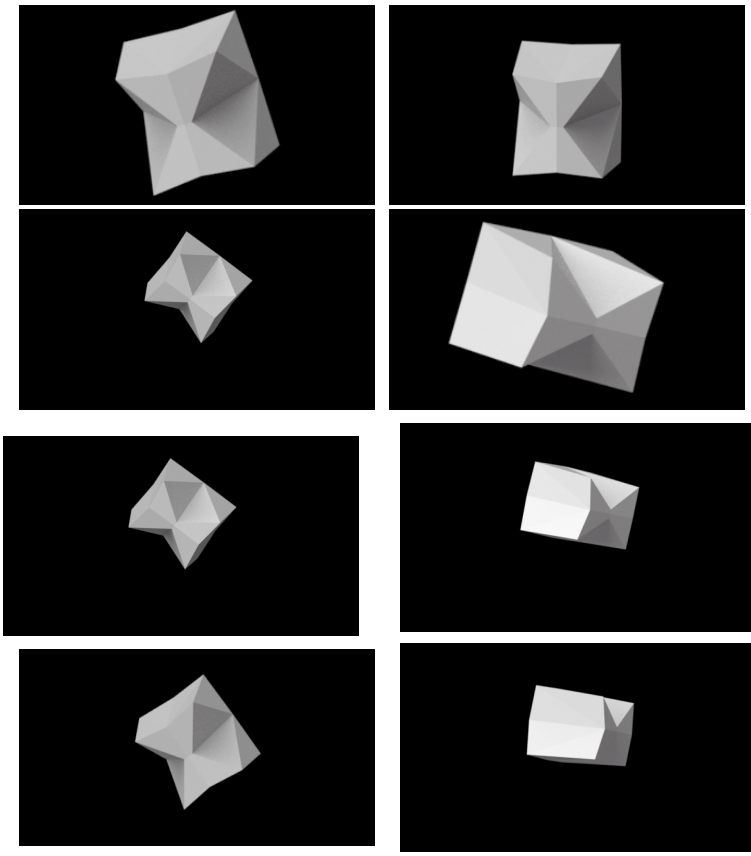
Stimulus Object 11



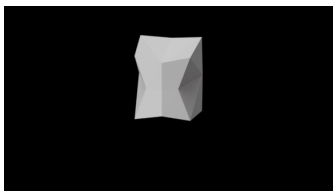


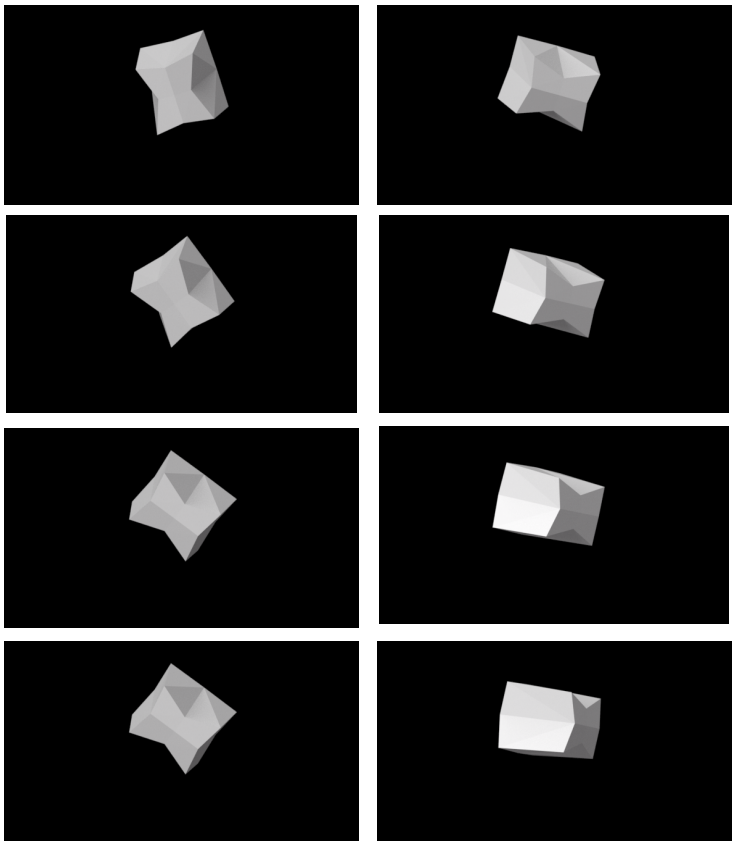
Stimulus Object 12





Stimulus Object 13





I.3 Stimuli from Pilot Versions of Experiment

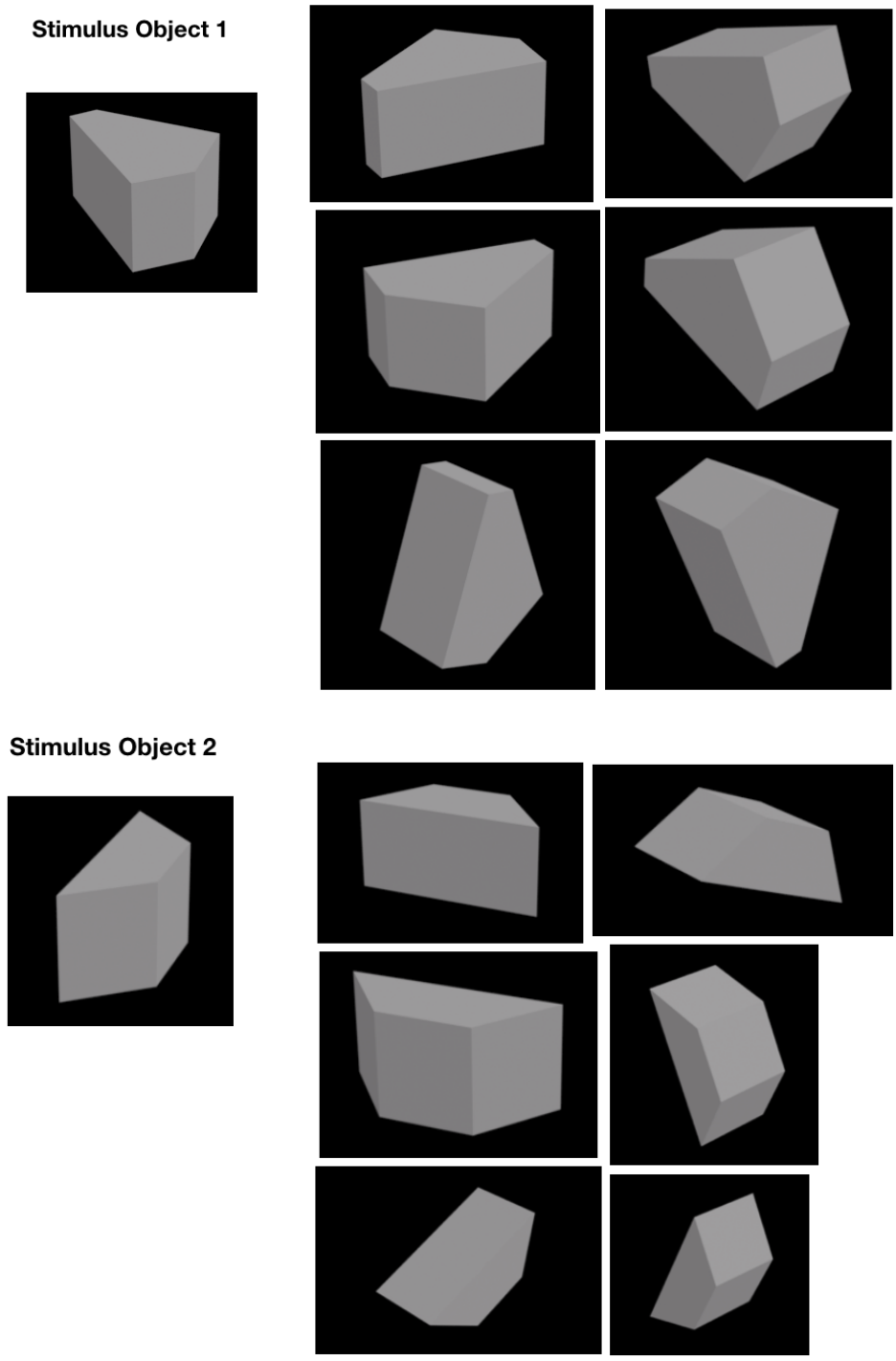


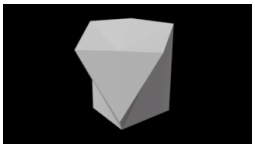
Figure 28. Stimulus objects lacking regularity and enough complexity failed to be consistently recovered during the pilot versions of our experiment. Complexity is related to the number of vertices and faces in the object. Objects with eight vertices were not recovered consistently during pilot tests.

I.4 Experiment Results

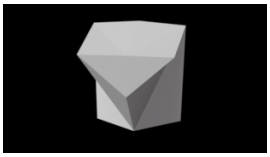
I.4.1 Block-wise Task Performance by Subjects

For each of the blocks, the accuracy of response for all objects is presented below:

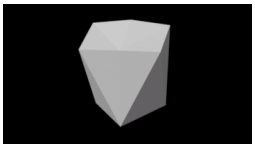
Block 1



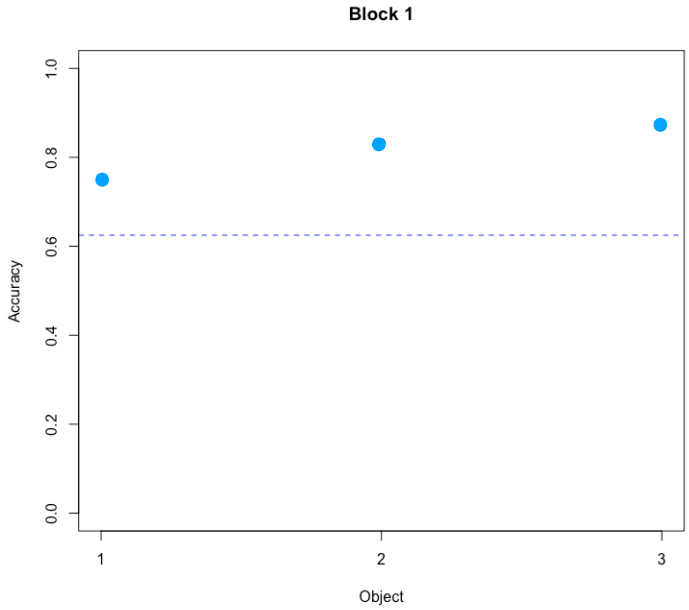
Object 1



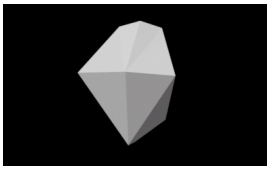
Object 2



Object 3

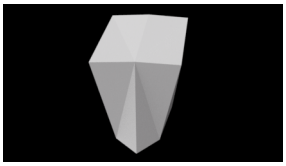


Block 2

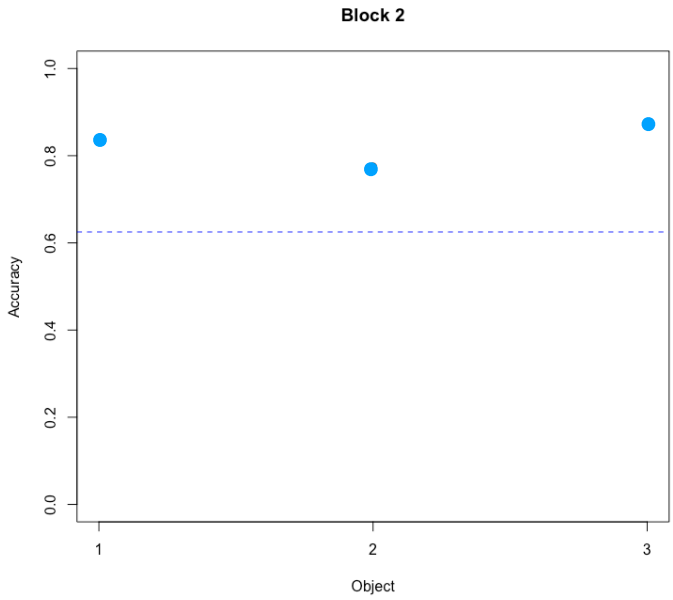


Object 1

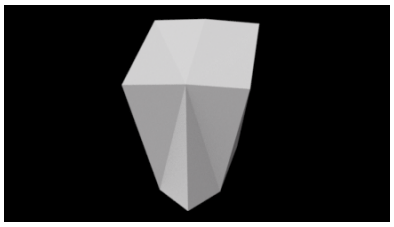
Object 2



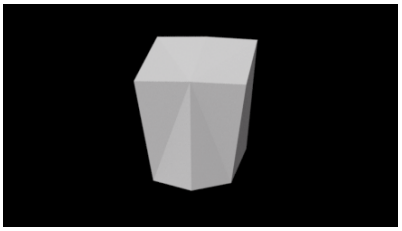
Object 3



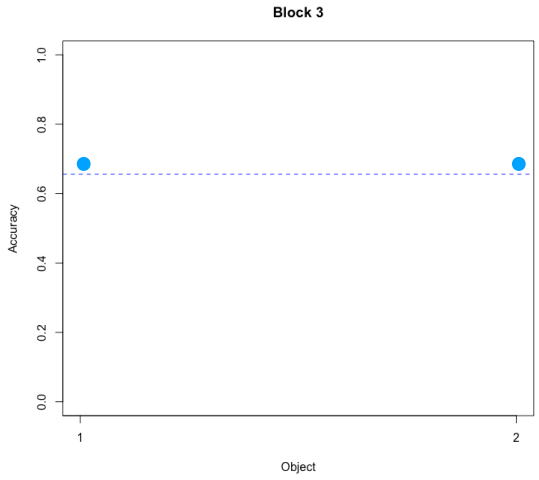
Block 3



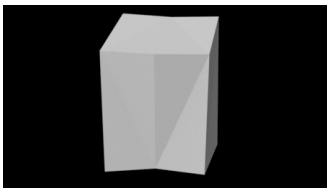
Object 1



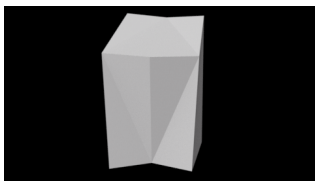
Object 2



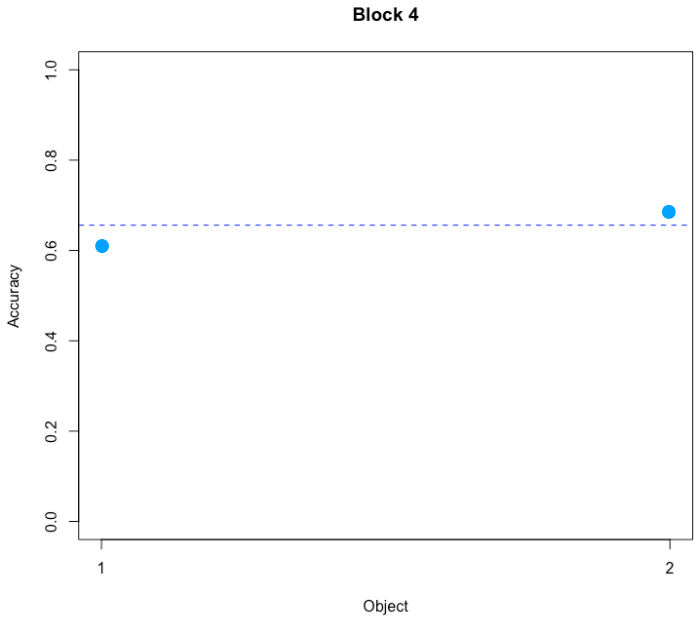
Block 4



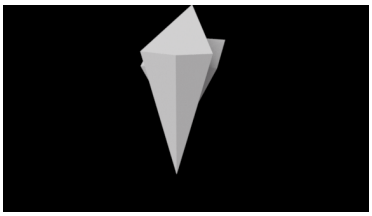
Object 1



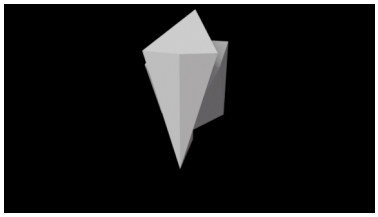
Object 2



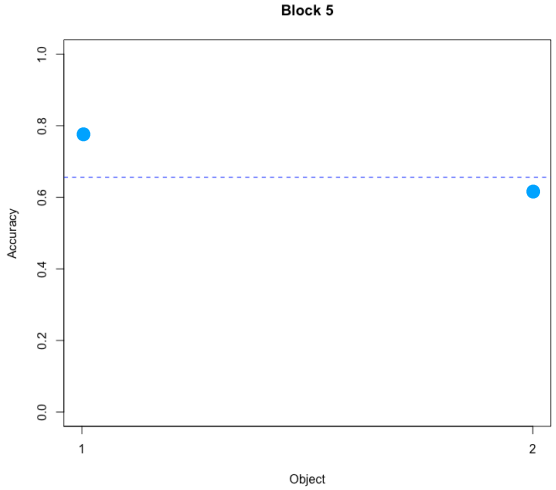
Block 5



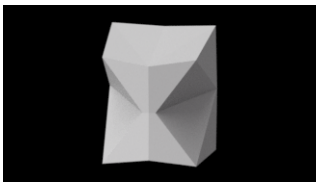
Object 1



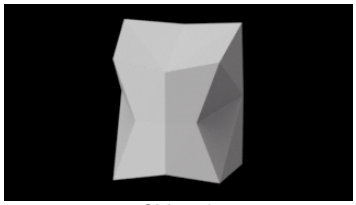
Object 2



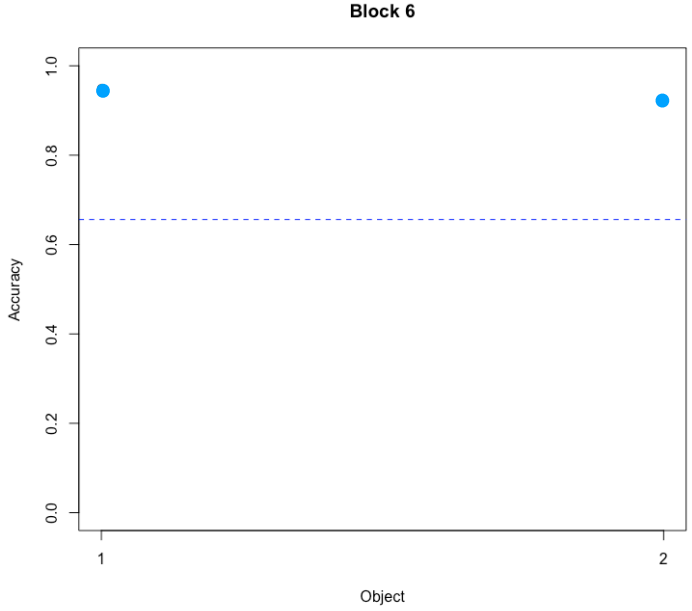
Block 6



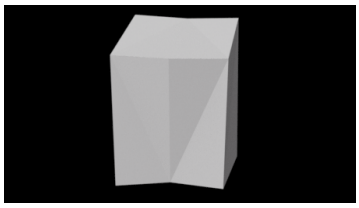
Object 1



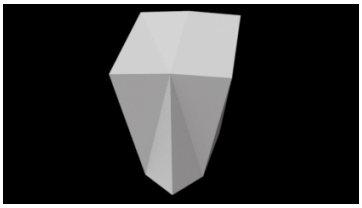
Object 2



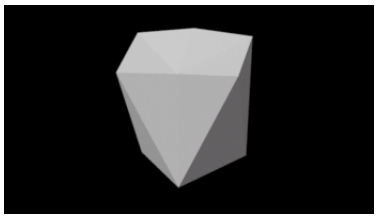
Block 7



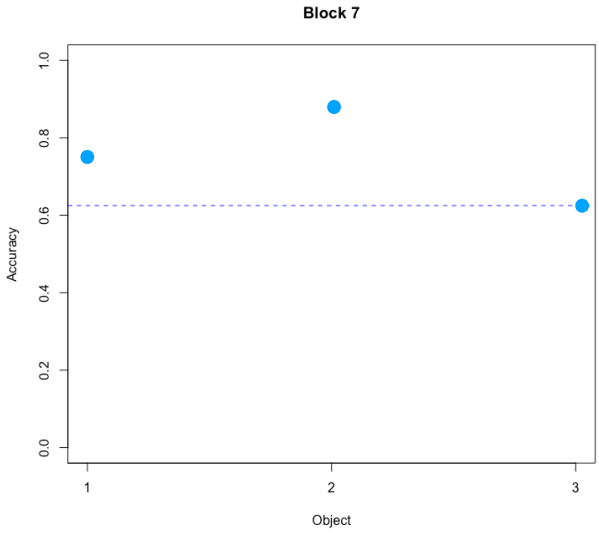
Object 1



Object 2



Object 3



I.5 Future Extensions - Mathematical Discussion

Circularity measure: First proposed in Haralick (1974), a simple way to calculate circularity of a shape is by using shape centroid and measuring all Euclidean distances from the centroid to each boundary pixel. With this set of distances, the median μ and standard deviation σ can be calculated. These statistical parameters can then be used to calculate a ratio that measures the circularity C of a shape. This measure is defined as:

$$C = \mu_R / \sigma_R$$

C is measured lowest (0) for a perfect sphere and would increase continuously for more skewed shapes.

Compactness measure: As proposed in Bribiesca (2000) and reviewed in Montero and Bribiesca (2009), compactness can be computed for a solid composed of voxels, area A of the enclosing surface of a rigid solid composed of finite number n of voxels corresponds to the sum of areas of pixels which form the visible faces of the solid. The contact surface area A_c of a rigid solid composed of a finite number of voxels corresponds to the sum of voxels which are common of two faces. The contact surface area can be computed as the following:

$$A_C = (Fan - A)/2$$

where F is the number of faces. The minimum and maximum contact areas:

$$A_{c_{min}} = a(n-1)$$

$$A_{c_{max}} = (aFn - 6a(n)^{2/3})/2$$

where a is the area of a voxel.

The measure of discrete compactness is defined as:

$$C_D = (A_c - A_{c_{min}})/(A_{c_{max}} - A_{c_{min}})$$

The measure of discrete compactness if maximized by a cube and values vary from 0 to 1 continuously.

After computing the first estimate of missing z-values, the model can then optimize circulairty and compactness measures (C and C_D) further using constrained optimization. At each layer, the objective function is the minimization of one particular constraint while keeping the z-values within the limits of two other constraints. Here is an example:

$$f_{SYM}(z*) = argmin_z(\mu_R/\sigma_R)$$
$$s.t. \frac{(A_C - A_{min})}{(A_{C_{max}} - A_{C_{min}})} - e = 0$$

Now, after finding the $z*$ value that satisfies these constraints, the gradients of the parameters G, A are also computed wrt $z*$ and these gradients are used to back-propagate gradients back to the previous layers. After several iterations, the values of the parameters $A, \mu_R \sigma_R$ should converge. This is a point where all three major constraints in the model are satisfied for a particular set of training inputs. In order to test if these learned parameters are generalizable or not, the estimates from the model can then be used to test against human perception at object level as obtained from our experiment.

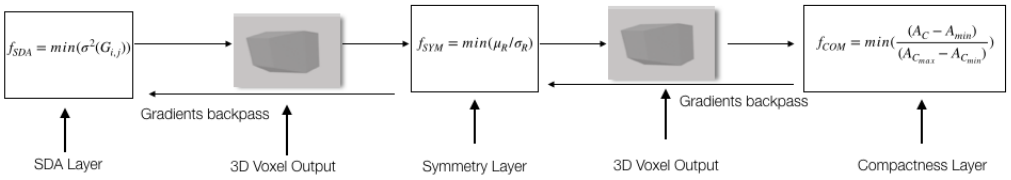


Figure 29. Model minimizes each constraint and computes gradients for other constraints for backward pass