A Self-Learning Map-Seeking Circuit For Visual Object Recognition

Rohit Shukla
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Email: rshukla3@wisc.edu

Mikko Lipasti
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Email: mikko@engr.wisc.edu

***Abstract*—The mammalian visual system is uniquely capable of robustly recognizing objects in its field of view regardless of their orientation, scale, or position, while learning new objects from a small number of training examples and generalizing robustly to a broad class of visually similar objects. The cortical structures that implement the visual system have been successfully emulated in several biologically-inspired synthetic vision systems. Developmental evidence lends credence to the claim that visual cortical structures emerge during development, i.e. they self-organize, when exposed to training stimulus. This paper demonstrates that a set of simple developmental rules can govern the emergence of a self-learning variant of a map-seeking circuit (SL-MSC) in a simulated visual system. The SL-MSC is capable of the same visual tasks as the original hand-crafted MSC: object recognition independent of rotation, translation, and scaling, and the ability to identify and learn new objects. The SL-MSC learns invariant visual transformations by relying on temporal association in its visual field, and is able to group the transformations into independent layers. Experimental results show that the SL-MSC can generalize the rotation, translation, and scaling transformations learned for one object to new objects, leading to learning and recognition of new objects with very few training samples.**

I. Introduction

The human visual system is very adept at recognizing objects. Even before a human infant first opens its eyes, spontaneous retinal activations are driving development of the visual system [1], [2]. From the very beginning the infant is able to recognize its mother's face. An early developmental task is to organize this input into objects and learn to recognize them despite variations in scale, rotation, and position in the visual field. As humans grow they start recognizing objects moving in complex scenarios. How is it that our eye is able to learn these transformations and be able to store them so that we can use them later for object recognition?

The human visual system is efficient in recognizing and classifying objects, but computers are still not robust enough to process the visual information in the same way as humans do [3]. A considerable number of articles have been published that discuss about how humans are able to recognize objects based on their invariant features and later extend the concepts of human visual system to computer visions for robust object recognition and classification [4], [5], [6] and [7]. One question that remains to be addressed is how a human being is able to learn invariant transformations that map the seen object to the reference object present in the memory. Prior work such as [8], and [9] have addressed this problem and proposed learning mechanisms such as Hebbian learning, to answer this question. To the best of our knowledge, there has been no other work that has proposed a computational model which explains how the mammalian visual system learns invariant transformations such as translation, rotation and scaling, and organizes them into independent layers.

In [10] and [11] authors have proposed a multi-layer hierarchical architecture of visual system that discounts image transformations and generates a discriminative feature vector, or signature, for learnt images. These signatures are invariant to local and global affine transformations. Sample complexity of learning models reduces significantly if these models are able to factor out image transformations during the development phase, as a result, achieving the goal of recognition with one or very few labeled examples. The proposed conjecture is to find a computational model for the ventral stream that learns to factor out image transformations. The formalization and proof of the conjecture was left as an open problem. The articles state that ventral stream learns and stores a group of transformations (G) as seen through the aperture of receptive field. The human eye stores an image as image patches (t^k, where $k \in [1, K]$ are the K templates of image patches stored in the memory) in the memory and simple cell present in human eye stores the transformed image patch ($g_i t^k$, where $\forall i, g_i \in G$). This learning operation is performed through Hebbian learning mechanism. A one-layer architecture can store all of the transformed image patches and achieve global invariance, whereas, to achieve local invariance and robust signatures for different parts of the image, authors have proposed a multi-layer hierarchical architecture that is similar to HMAX algorithm [10]. In the implemented models authors have hardwired translation and scaling transformations.

Using Map-Seeking Circuits (MSC) [12] as our basis, we present a computational model that demonstrates how a human is able to learn the invariant transformations and later use them to recognize various objects. This computational model gives us an insight into how human beings use invariant features of the learnt objects and try to learn a combination of independent invariant transforms to map the memory reference pattern to the input target pattern.

As presented in [13], MSC is a biologically inspired algorithm developed by David Arathorn at the Center for Computational Biology at Montana State University [12]. It is a model-based approach that assumes that the correspondence

between a model and an observation of it can be represented by a decomposition of invertible transformations, such as, scaling, translation and rotation. Unlike the HMAX algorithm [4] and other similar feedforward models, which simply report the presence (or absence) of a target object in the reference image or not, MSC seeks to find an appropriate set of transforms that maps a stored template to an unknown signal, returning both the recognition/classification response as well as the set of maps used to identify the object. The algorithm uses superposition with an iterative matching process to converge on the best set of transforms that map a template to a target in an input signal.

MSC is composed of one or more layers and a set of templates. Each layer represents a transformation such as translation, scale or rotation. The algorithm performs a set of transforms at each layer and sums the result. The result is then sent to the following layer where the process is repeated for another set of transformations. The algorithm depends on The Ordering Principle of Superposition [14]. The principle states that if matches are computed between a pattern, A, and a superposition of a set of patterns, the match will be greatest for the pattern within the superposition that is most like A. The use of superposition reduces the computational complexity from exponential growth to linear growth, thus making the problem tractable.

In this article, we have introduced a self-learning MSC algorithm which learns invariant transformations during object recognition, rather than hardwiring these transformations into pre-assigned layers as described in [12]. The proposed MSC architecture initially consists of no transformation layers that would match the input image with the memory reference images. As the transformed reference images or new objects are given as input to SL-MSC, the proposed architecture will learn the required set of transformations or the new input objects and allocate them to appropriate layer or memory location. Object permanence in the visual field lends itself to temporal association across instances of the transformed image; the transforms themselves are learned by solving for the maps that connect pairs of transformed instances, similar to [15].

This algorithm gives us an insight into how we are able to recognize objects despite the extreme variabilities of the retinal image they might generate. Prior work such as [8], [10] and [11] mention that association between target and reference templates can be learnt through Hebbian learning mechanism and the learnt image transformations can be stored as group of transformations. But, such a method suffers from the problem of combinatorial explosion of template matching. Our model performs simple matrix algebra and proposes how mappings emerge in human visual system to be able to distinguish among independent image transformation functions and is also able to group similar transformations together. Based on recognition of object permanence, the proposed algorithm matches reference pattern with the input pattern by comparing an ordered list of interesting or invariant features present in the the two images and later learn the necessary affine transformation to match the two features. It is argued here that based on spontaneous activation feature of the eye and its ability to recognize temporal invariant features of the image, through object permanence and temporal association, the human brain

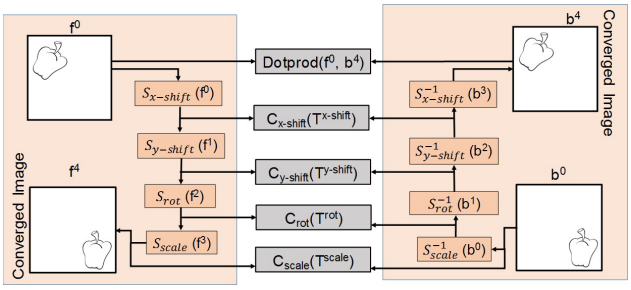

Figure 1. Architecture of Map-Seeking Circuits

is able to learn and retain invariant transformations.

We have used the MSC algorithm to focus on the problem of learning invariant transformations from the stream of unlabeled examples. It has been shown that through the use of simple matrix algebra we can group similar invariant transforms together and assign separate layers to independent transforms.

II. Map-Seeking Circuits (MSC)

The architecture of MSC is shown in Figure 1. Gregoire and Maher [13] provide a good overview of MSC which will be paraphrased in this section. The MSC consists of two paths, viz., the forward path and the backward path, roughly corresponding to the feedforward and feedback paths present in the mammalian visual streams. The forward path transforms the input image along predefined set of independent image transformations to find out the best set of transformations that will map the input image to match with the images stored in memory template. Sequence of transformations on input image along forward path have been denoted in Figure 1 by $S_{x-shift}$, $S_{y-shift}$, S_{rot}, and S_{scale}. Similarly, the backward path transforms the memory images by applying inverse independent image transformations on them. The backward path finds out the best set of transformations that will map memory template image to match with the input image. The sequence of transformations on memory templates along the backward path have been denoted in Figure 1 by S^{-1}_{scale}, S^{-1}_{rot}, $S^{-1}_{y-shift}$, and $S^{-1}_{x-shift}$. After each set of independent transforms have been applied on the input image, MSC superimposes the output of image transformations and sends it to the next transformation layer. The degree of best match between superimposed results in forward path and backward path is calculated using dot product operation, as illustrated in figure 1, by $C_{x-shift}(T^{x-shift})$, $C_{y-shift}(T^{y-shift})$, etc. As suggested in [11] and [12], a practical and biological plausible way to match invariances between images is obtained by taking dot-product between input and memory template. The templates that have the highest dot-product values indicate the most promising transforms that map the input to one of the several stored templates. The most promising transforms are left unchanged, whereas, rest of the transforms are penalized on the forward and backward path by virtue of a competition function.

As the iterations progress, the mismatched transforms are attenuated with respect to the better ones. Eventually, the algorithm forces all transforms to zero except for the best transform along each dimension. In the end, algorithm is able

to find those set of transformations that will map the input image to match with the images stored in memory template. If the algorithm can not find a mapping between the target and the template, then all the transforms are driven to zero resulting in a null result.

The MSC algorithm will always converge to either a set of unique transforms or a null condition where all the transforms in each layer are driven to zero. In case a unique set of transforms are found, the similarity of the input and the inverse transformed template must exceed a threshold for a positive mapping to result. The threshold is set by comparing the similarities of several positive inputs to negative inputs during a training session. Ideally, there will be a large discrepancy between the two. If there is overlap, then amount of false rejection errors can be traded for false acceptance errors depending on which error is considered more valuable for a given application.

III. Learning Invariant Transforms

The proposed algorithm learns new unlabeled objects and new transformations (maps), and later uses these learnt objects and maps to recognizing objects in the visual input. Initially, there are no reference images or maps in the SL-MSC memory. When a new object is presented to the MSC, it will first notice that there is no learnt object in its memory to match with. Thus, the MSC will identify the object of interest and store this object in its memory. Subsequent presentations of the first object at different scale, translated location, and rotation are used to learn generalizable maps that the SL-MSC utilizes to achieve invariant recognition.

When images with new objects are presented to the SL-MSC, it will try to map its existing memories to the input image after performing all the transformations present in different layers of the SL-MSC. If the SL-MSC does not converge to a correct solution then the SL-MSC will check whether the input image is of a new object or whether it requires a previously-unseen transformation.

Perspective transformations are one of the most common transformations that visual system has to deal with. Two-dimensional affine transformations cannot implement perspective transformations, hence, an object that has undergone a change in orientation in 3D space will appear as a new object to SL-MSC. As a result, SL-MSC will learn the perspectively transformed object and store it in the memory. Due to computational complexity, MSC [16] suggests to store perspective transformation of rigid objects as separate memory templates.

A. Learning new objects

To check whether the input image is that of a new memory object, SL-MSC will check all the objects stored in the memory and, through invariant feature extraction technique, match the input image against each reference image stored in the memory. If none of the reference templates match with the input image, this means that input image is that of a new object and this object should be stored in the memory.

The process of recognizing objects by matching its invariant features with the reference images is similar to the biological counterpart of how our eye recognizes objects. As argued in [8] and [17], invariance for novel objects can be inherited from the invariant detection of templates, which is accomplished via replicated feature detectors and location- and scale-invariant pooling performed by algorithms like HMAX [4]. A realistic model of mammalian vision should be capable of learning new objects from a very small number of examples. Reliance on a pre-existing (or previously-developed) set of invariant transforms can enable this capability, since any subsequent presentation of the new object can be transformed to match its initial presentation.

B. Learning new transforms

As the object is translated, rotated or scaled in the visual input, the dot product between the memory template and the input template will be less than the dot product of the input template with itself, as shown in equation 1. This happens because the visual system has not yet learned the new transformations that will make the memory image similar to the input image. And, when the visual system tries to find the correlation between the input image and the corresponding image in the memory using dot product, the current visual system will get a low value for this correlation. The human visual system experiences similar mismatches due to both eye movements (saccades, which shift the center of the visual field) and physical movement of the object with respect to the human eye. When this occurs, the human visual system learns the transformation required to map the spatially transformed object to the temporally-correlated reference object present in the memory. Correspondingly, the SL-MSC visual system will try to learn the necessary transformations that will result in higher dot-product values between input image and its corresponding memory reference image [9].

$$\langle T_{input}, T_{mem} \rangle \; < \; \delta \tag{1}$$

Where $\langle ., . \rangle$ represents the dot product between two vectors, T_{input} is the input image template and T_{mem} is the memory image template. If the value of $\langle T_{input}, T_{mem} \rangle$ is less than a threshold, say δ, the MSC will try to map the input template to the memory template by learning the necessary invariant or affine transformation. To learn this new affine transformation, MSC will perform following set of functions

1) Extract the invariant features of the object shown in the input image F_{input} and in the memory image F_{memory}. The ordered list of invariant features present in input image should match the ordered list of invariant features in the memory image. As presented in [9] and [18], human eyes will saccade to the interesting or invariant features present in the image and will try to match the two images by comparing these features. To successfully match the two images, human eye would have to learn the necessary set of transformations that will be later stored in memory.
2) The visual system will learn to map the input image to the stored memory image by learning the necessary 3X3 affine transformation matrix for forward path and backward path. These matrices can be learnt by performing division of coordinates of invariant features between memory image and the input image. For the forward path MSC will learn the set of transformations that will map the input image to match with the memory image as represented in equation 2.

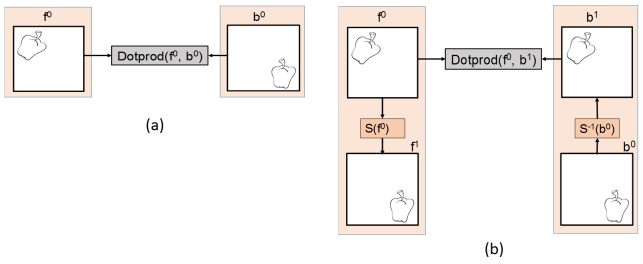

Figure 2. An overview of the algorithm to assign new layer in Map-Seeking Circuits. (a) First, MSC will compare the input image with memory images by taking the dot-product between two images. (b) If the value of dot-product is below a certain threshold, the required transformation is learnt by MSC and it is either appended to an existing layer of transformations or assigned a new layer

In the backward path, MSC will learn transformations to map memory image to match with the input image, as represented in 3.

$$T^{fwd} = \frac{F_{memory}.[X,Y,1]}{F_{input}.[X,Y,1]} \tag{2}$$

$$T^{bkwd} = \frac{F_{input}.[X,Y,1]}{F_{memory}.[X,Y,1]} \tag{3}$$

3) Check whether the learnt affine transformation is dependent on previously learnt affine transformations or is it an independent transformation. If it is dependent on previously learnt transformations then append it with the set of transformations on which the new transformation is dependent, else, assign a new layer in MSC for this new transformation.

An overview of this algorithm has been presented in Figure 2. Pseudo-code for the three steps of MSC have been presented in algorithms 1, 2 and 3.

Algorithm 1 Check whether MSC needs to learn a new transformation

while MSC has not converged to a valid image **do**
  $ReferenceImage \leftarrow MSC(Input)$
  if $\langle ReferenceImage, Memory \rangle < \delta$ **then**
    $(T^{fwd}_{new}, T^{bkwd}_{new}) =$ LEARNTRANSFORMATION
    ASSIGNLAYER(T^{fwd}_{new})
    ASSIGNLAYER(T^{bkwd}_{new})
  end if
end while

C. Assigning thresholds for SL-MSC to learn new object and transformations

One of the tasks of the proposed SL-MSC is to identify whether there is any need for MSC to learn the new object or transformation, or can it recognize the input image from the reference memory images with the given set of transformations. For this decision SL-MSC relies on equation 1, thus, selecting the value of threshold is crucial so that the scaled up images that have more number of pixels than the original image, do not have an unfair advantage over dot-product between T_{input} and T_{mem}. To take care of this issue, the proposed SL-MSC has defined δ in equation 4 as,

Algorithm 2 Learn the new transformation matrices for forward and backward paths

Input: Invariant feature vectors of input image and memory image
Output: Learnt transformation matrices for forward (T^{fwd}_{new}) and backward (T^{bkwd}_{new}) paths

procedure LEARNTRANSFORMATION
  $T^{fwd}_{new} = \frac{F_{memory}.[X,Y,1]}{F_{input}.[X,Y,1]}$
  $T^{bkwd}_{new} = \frac{F_{input}.[X,Y,1]}{F_{memory}.[X,Y,1]}$
end procedure

Algorithm 3 Check linear dependence of learnt transformation

Input: learnt transformation matrix T_{new}
Output: T_{new} was appended to an existing layer or it was allocated to a new layer

procedure ASSIGNLAYER(T_{new})
  if $|\vec{T}_{new}| \neq 1$ **then** Assign a new layer for T_{new}
  else
    for $i = 1$ *to* L **do**
      $R_{Layer} \leftarrow rank(\vec{I}_v, \vec{T}^{fwd}_i, \vec{T}^{bkwd}_i)$
      $R_{Transf} \leftarrow rank(\vec{T}_{new}, \vec{I}_v, \vec{T}^{fwd}_i, \vec{T}^{bkwd}_i)$
      if $R_{Transf} = R_{Layer} + 1$ **then**
        $\vec{T}_{new}$ is linearly independent from transformations in $Layer_i$
      else
        $\vec{T}_{new}$ is linearly dependent on transformations in $Layer_i$
        break
      end if
    end for

    if $\vec{T}_{new}$ is linearly dependent on $Layer_i$ **then**
      Append T_{new} to $Layer_i$
    else
      Assign a new layer for T_{new}
    end if

  end if
end procedure

$$\delta = c\frac{\sum_{i=1}^{M} P_{mem,i}}{M \times P_{input}}\langle T_{input}, T_{input}\rangle \tag{4}$$

where, $\sum_{i=1}^{M} P_{mem,i}$ represents the total number of pixels of all the images stored in the memory, M is the number of images in the memory and P_{input} is the total number of pixels in the input image. The tunable parameter $c \in [0,1]$ can be set by the user so that the MSC is able to learn new object or transformations.

D. Identifying linear independence of learnt transformations

One of the challenging tasks in learning new invariant transformations is to be able to figure out independent transformations. Due to the linear nature of affine matrices, we can rep-

resent dependent transformations in terms of linear combinations of learnt matrices, whereas, independent transformations cannot be expressed as linear combination of learnt matrices. Thus, after we have identified an independent transformation we assign it a new layer.

Since the affine transformations can be represented as linear combination of variables, we use rank and determinant of matrices to determine whether a new transformation is dependent on previously learnt transformation.

Given a set of *n* affine transformation vectors, viz. $\vec{T_1}$, $\vec{T_2}$,...,$\vec{T_n}$ (where $\vec{T_n}$ is newly learnt affine transformation), a linear combination of affine vectors can be represented as

$$L = \sum_{i=1}^{n} \alpha_i \vec{T_i} \tag{5}$$

Where α_1, α_2,...,α_n represent linear coefficients. The transformations are said to be linearly independent if *L* is equal to 0, and $\forall$ *i*, α_i is equal to 0. That is, $\vec{T_n}$ cannot be expressed as a linear combination of any of the previously learnt affine transformations. We can check the independence of affine matrices using rank of the vectors.

As we are focusing on rigid body transformations, we assume that affine transformations do not change shape of the object. To account for scaling separately, we look at the determinant of the learnt transformation. If the determinant is not equal to 1, this indicates that we are performing a scaling operation or changing the size of the object. Thus, we factor out the scalar value of the learnt transformation and assign it to the layer that is responsible for changing the size of the rigid body.

E. Algorithm for assigning layers to new invariant transformations

Algorithm 3 presents the pseudo-code of the proposed algorithm that makes the decision whether T_{new} should be allocated to an existing layer or to a new layer.

To calculate rank of the transformations in $Layer_i$ (R_{Layer}), we convert 3X3 affine matrices, viz., identity matrix, and transformation matrices of forward and backward paths, into 9X1 vectors i.e., $\vec{I_v}$, $\vec{T_i}^{fwd}$, and $\vec{T_i}^{bkwd}$, respectively. Rank of the new transformation matrix (R_{Transf}) is calculated between 9X1 vector representation of new transformation matrix, i.e., $\vec{T}_{new}$ and $\vec{I_v}$, $\vec{T_i}^{fwd}$, and $\vec{T_i}^{bkwd}$. If the value of R_{Transf} is more than R_{Layer}, this implies that the new transformation is linearly independent from previous transformations. Hence, a new layer should be assigned. Otherwise, T_{new} is linearly dependent on the transformations present in $Layer_i$. Therefore, there is no need to loop through other layers, and T_{new} can be appended to the current i^{th} layer.

Even though a scaling transformation seems to be linearly dependent on other affine transformations, we can still treat it as an independent transformation since multiplication between a scalar unit and a vector does not depend on the combination of other vectors. Hence, we factor out the scaling unit from learnt transformation and assign it to a separate layer.

Claim: To calculate rank of $Layer_i$ we need three vectors, viz., vector of identity matrix ($\vec{I_v}$), forward transformation matrix ($\vec{T_i}^{fwd}$) and backward transformation ($\vec{T_i}^{bkwd}$).

Proof: As shown in equations 2 and 3, a 3X3 affine transformation matrix can be learnt by dividing invariant features of input image, with the invariant features of the image stored in memory. An affine transformation matrix can be represented as shown in equation 6

$$A = \begin{bmatrix} sr_0 & sr_1 & 0 \\ -sr_1 & sr_0 & 0 \\ X & Y & 1 \end{bmatrix} \tag{6}$$

Where s is the scaling factor, r_0 and r_1 are the cosine and sine factors, respectively, by which an image is rotated, and X and Y are the amount by which the image is translated along horizontal direction and vertical direction, respectively.

To calculate the rank of transformations at $Layer_i$, the transformation matrices are first converted to their respective 9X1 vector forms. For example, the fwd and $bkwd$ transformation matrices for rotation are shown in equation 7.

$$\vec{T}_{rot}^{fwd} = \begin{bmatrix} r_0 \\ r_1 \\ 0 \\ -r_1 \\ r_0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \vec{T}_{rot}^{bkwd} = \begin{bmatrix} r_0 \\ -r_1 \\ 0 \\ r_1 \\ r_0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{7}$$

Equation 7 tells us that $\vec{T}_{rot}^{fwd}$ and $\vec{T}_{rot}^{bkwd}$ represent rotations in opposite directions. Considering the concept of MSC, if we learn to rotate the input image in clockwise direction so that it can match with the image stored in the memory, then we would have to rotate memory image in counter-clockwise direction so that it can match with the input image.

If a new rotation transformation is learnt by the MSC, we would not be able to find independence/dependence of the learnt transformation with the two forward path and backward path transformation vector. That is, MSC would not be able to find independence/dependence of $\vec{T}_{new}$ with just $\vec{T_i}^{fwd}$ and $\vec{T_i}^{bkwd}$. Let us assume, that T_{rot} is the learnt rotation transformation and its vector representation is $\vec{T}_{rot}$. As per equation 5, if $\vec{T}_{rot}$ is dependent on $\vec{T}_{rot}^{fwd}$ and $\vec{T}_{rot}^{bkwd}$, then, L can be equal to 0, even though α_1, α_2 and α_3 are not equal to 0. But this is not true. With all three vectors representing rotation transformations, L is equal to 0, only when α_1, α_2 and α_3 are all equal to 0. Therefore, to show dependence among similar transformation matrices, we add identity matrix or $\vec{I_v}$ as a correctional matrix for each layer.

Equation 9 provides an algebraic proof as to why three vectors are required to check dependence of learnt matrix, and equation 10 proves that even with three vectors MSC will be able to distinguish among linearly independent affine transformations using rank of the matrices. Let the vector $\vec{T}_{rot}$ represent the learnt rotational matrix and $\vec{T}_{y-shift}$ represents the learnt vertical shift matrix. The two vectors, viz., $\vec{T}_{rot}$ and

$\vec{T}_{y-shift}$ are presented in equation 8, where x_1 and x_2 are the cosine and sine values of unknown rotation angle and Y represents the vertical shift of the image.

$$\vec{T}_{rot} = \begin{bmatrix} x_1 \\ x_2 \\ 0 \\ -x_2 \\ x_1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \vec{T}_{y-shift} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ Y \\ 1 \end{bmatrix} \tag{8}$$

$$\begin{aligned} \alpha_1\vec{I}_v + \alpha_2\vec{T}_{rot}^{fwd} + \alpha_3\vec{T}_{rot}^{bkwd} + \alpha_4\vec{T}_{rot} = 0 \\ \Rightarrow \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 0 \\ \alpha_1 + \alpha_2 r_0 + \alpha_3 r_0 + \alpha_4 x_1 = 0 \\ \alpha_2 r_1 - \alpha_3 r_1 + \alpha_4 x_2 = 0 \end{aligned} \tag{9}$$

$$\begin{aligned} \alpha_1\vec{I}_v + \alpha_2\vec{T}_{rot}^{fwd} + \alpha_3\vec{T}_{rot}^{bkwd} + \alpha_4\vec{T}_{y-shift} = 0 \\ \Rightarrow \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 0 \\ \alpha_1 + \alpha_2 r_0 + \alpha_3 r_0 + \alpha_4 = 0 \\ \alpha_2 r_1 - \alpha_3 r_1 = 0 \\ \alpha_4 Y = 0 \end{aligned} \tag{10}$$

Equation 9 can be satisfied with α_1, α_2, α_3, and $\alpha_4 \neq 0$. Hence, $\vec{T}_{rot}$ can be appended with the layer that contains the rotational matrices. But, equation 10, can be true if α_1, α_2, α_3, and α_4 are all equal to zero. Or, equation 10, can be true when r_1 is 0, which means that we are learning rotational layer with 0 degree rotation, a contradiction. This implies, that rank of the set of vectors, $\vec{I}_v$, $\vec{T}_i^{fwd}$, $\vec{T}_i^{bkwd}$ and $\vec{T}_{y-shift}$ has a value one more than rank of $Layer_i$, so, matrix representation of $\vec{T}_{y-shift}$ will be either appended to another layer or will be assigned a new layer.

As a result, we have proved that to calculate rank of $Layer_i$ we need three vectors, viz., vector of identity matrix ($\vec{I}_v$), forward transformation matrix ($\vec{T}_i^{fwd}$) and backward transformation ($\vec{T}_i^{bkwd}$). ■

F. Arrangement of transformation layers

The preceding sections described an algorithm that detects new transformations and later assigns them to appropriate, linearly independent layers. Another important part that needs to be discussed is how these layers have to be arranged so that the proposed MSC is able to learn the transformations in a way that is similar to how the human visual system behaves.

As shown in figure 1, when an input image appears, MSC first performs translation on it, followed by rotation, and then scaling. In biological sense, eyes will first saccade to the location of the object through translation, then rotate it and scale it so that input image matches the image in the memory. As presented in [18], when humans look at an image, the visual system uses rapid eye movements (saccades) to actively scan the image. During these movements their eyes fixate on interesting or invariant parts of the image. Because the central part of the retina, the fovea, has a higher resolution than surrounding areas, the invariant locations are examined with high precision, because the entire picture is not processed. Hence, the human eye tends to learn scaling and rotational transformations first because the eye quickly saccades to the invariant positions for these two transformations. As the object moves closer to the periphery of the vision, the human eye starts learning about the translational transformations so that eye can later saccade to the interesting portions of the object.

Furthermore, visual acuity is initially poor, and hence best suited for learning the scaling transform, e.g. as the infant's mother's face approaches and recedes. Acuity improves rapidly during the first six postnatal months [19], gradually providing enough resolution to support learning of rotation of familiar objects, and eventually translation, as peripheral vision provides enough acuity to sustain temporal association for objects as they move to and from the receptive field periphery.

IV. Results

The proposed SL-MSC algorithm was simulated in MATLAB. For detecting new transformations and new input images, SURF feature extraction technique was used, which is a part of MATLAB's computer vision toolbox [20]. SURF was used to get the matching feature descriptors present in input image and corresponding reference memory image. Based on these feature descriptors, the required affine transformation was learned using the formula presented in equations 2 and 3. The algorithm was implemented on thirty different objects from ALOI dataset [21] as shown in figure 3. ALOI dataset consists of images that have varying illumination direction, illumination color and object viewpoint for real world objects. Since MSC requires sparse data representation, edge-detection in pre-processing step was implemented using the steps mentioned in [22].

Table I. Affine Transformations for which SL-MSC was Trained

Affine transformation	Values
Scaling factor	0.6, 0.8, 1.2 and 1.6
Rotation	Clockwise and counter-clockwise 15°, 30°, 45°, 60°, 75° and 90°.
x-translation	five along positive x-direction and five along negative x-direction.
y-translation	five along positive y-direction and five along negative y-direction.

To train the proposed SL-MSC, those images from ALOI dataset are selected that do not have any variations in illumination direction, illumination intensity and orientation, shown in figure 3. To train the proposed architecture for affine transformations, sequences of image frames were used, that were created in Adobe photoshop [23]. These training images had undergone only one transformation, i.e., the input images were either scaled, rotated or translated, and never a combination of two or more transformations. To get the architecture as shown in figure 1, SL-MSC was first trained with scaling, followed by rotation and finally translation. It was trained for 26 different affine transformations as shown in table I. These image frames are meant to simulate the environment where a human eye learns new objects and transformations by continuously saccading to moving objects and later use combination of these learnt values to identify transformations and objects.

Figure 3. ALOI images that were used to train and test the proposed SL-MSC

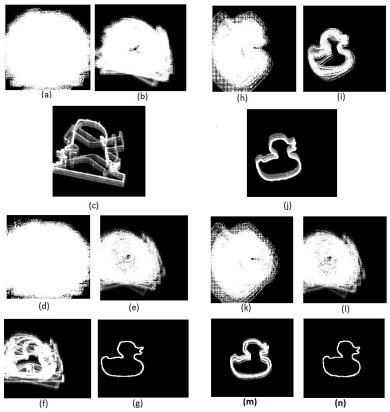

Figure 4. Represents the case when a horizontally translated duck image was given as input to SL-MSC. (a), (b) and (c) Show the scenario where SL-MSC is trying to get the memory image converge to a solution so that the result matches with the input image. (h), (i) and (j) Show the scenario where SL-MSC is trying to get the input image converge to a solution so that the result maps to one of the memory reference images. After SL-MSC learns the necessary transformation it restarts and tries to match the two images. (d), (e), (f) and (g) show the steps where memory reference images converge to a solution that is similar to the input image. Similarly, (k), (l), (m) and (n) represent the steps where input image converges to a solution that maps to the smiling duck image stored in memory template.

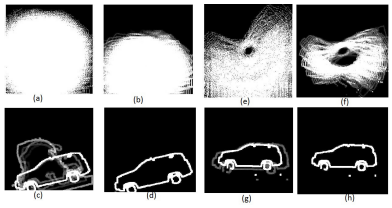

Figure 5. The white car image was used to test the proposed SL-MSC. (a)-(d) show the steps where memory reference template converges to a solution to exactly match the input image. Similarly, (e)-(h) show the steps where input image converges to a solution to exactly match the memory reference template.

The proposed algorithm was tested by randomly selecting an object from ALOI dataset that we had used to train our MSC and applying random combinations of affine transformations on it. We also used different variations of ALOI dataset objects to test whether SL-MSC is able to converge to correct solution even though there is different illumination direction, illumination intensity and orientation applied on the object. Threshold parameters that were set while training and testing SL-MSC have been summarized in table II.

Figure 4 shows the case when SL-MSC was not able to find the object and learned the required transformation to map input image to the memory reference image. Figure 5 shows the case when input image is present in the memory and SL-MSC converges to the correct solution after all the necessary transformations have been learned.

The results obtained after using the images from ALOI dataset suggest that SL-MSC is invariant to changes in illumination temperatures. Plots of figure 6 show the percentage of number of objects that SL-MSC was able to recognize under varying illumination directions. Figure 6 also shows the setup of the experiment conducted under ALOI dataset [21] to observe variation of illumination direction on real world objects. The object was kept on a rotating table and 5 light sources were mounted on a light bow that is perpendicular to the plane of the table. The five light sources were kept at angles -30°, -60°, 0°, 60°, and 30°, respectively, on the light bow. l1, l2, l3, l4, and l5 represent the case where exactly one light source was on. Lighting condition l6 means that lights l1 and l2 were on and l7 means that lights l4 and l5 were on. In case of l8, all of the five lights were illuminating the object. The images of objects were taken with cameras kept at three different azimuthal angles, viz., c1 (0°), c2 (-15°), and c3 (-30°). As stated earlier, MSC relies on sparse data representation to match the input object with the stored memory templates, thus, edge detection while performing image pre-processing becomes an important step. If the edge detection algorithm is not able to detect the proper edges of the input object either due to illumination differences or due to low brightness of colors, SL-MSC fails to converge on a correct solution. As observed from figure 6, since camera c3 is in front of the object and l1 and l2 light sources are behind the object, significant amount of lights gets cut off. As a result, a considerable portion of objects were not visible for ALOI conditions l1c3, l2c3 and l6c3, and edges of the object are not properly detected. As the light source l5 is closer to the object and camera c3, images are able to capture complete shape of the object, hence, SL-MSC gives 100% recognition since it is able to detect all of the edges of the object.

Table II. THRESHOLD PARAMETERS AND THEIR VALUES FOR SL-MSC

Threshold parameter	Value of the parameter
Edge Detection (Prewitt Algorithm in MATLAB)	0.012
Tunable parameter for competition functions	0.5
Tunable parameter for detecting new transformation or object (c in eqn. 4)	0.1

V. CONCLUSION

SL-MSC proposes an algorithm of how a human visual system might be able to find the mappings between the brain and the eye to match the input images with the images stored in the memory, by looking for the correct set of transformations and pruning out all of the other transformations. In this article we have presented how a human visual system might be able to learn the set of invariant transformations given the visual stimuli that we get bombarded with when we open our eyes. The main contributions of this paper are as follows:-

1) The proposed SL-MSC algorithm provides us with an

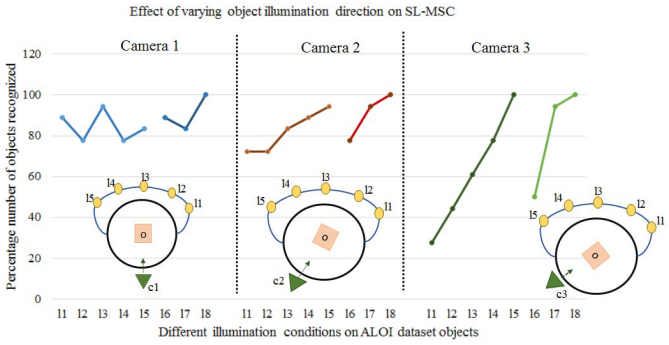

Figure 6. This graph shows the percentage of objects that SL-MSC can detect when illumination directions on the objects in varied. l1, l2, l3, l4 and l5 represent the case where only one of the 5 lights was on. In l6 lights l1 and l2 were on, whereas in l7, lights l4 and l5 were on and in l8 all of the 5 lights were on.

intuitive sense of how we learn invariant transformations. The visual system learns about new objects by saccading to the interesting features present in it. As the objects move in space, the spatial location of features in the object will also change. Therefore, the visual system would learn the transformation required to map the spatially transformed object to the temporally correlated reference objects present in its memory.

2) Once the visual system has learnt the invariant transformations, these transformations will be remembered in the memory and will be referred to again for mapping different objects to the corresponding reference objects in the memory.
3) The human eye is able to map independent transformations in separate layers; using linear algebra MSC can do the same by allocating independent functions to separate layers.
4) The arrangement of invariant transformations in the MSC is based on the higher resolution present around the fovea of the human eye and the movement of objects along the periphery of human vision, as well as evidence for improvement of visual acuity over the first six post-natal months of development [19].

Since the proposed algorithm uses simple matrix algebra to learn new transforms, instead of a Hebbian learning mechanism, it can be easily and efficiently implemented on conventional computing hardware to emulate the development and function of the human visual system. Future work will investigate thorough comparison of SL-MSC with object recognition algorithms and neural implementations of the proposed learning rules, along the lines suggested in [12].

VI. Acknowledgment

Mikko Lipasti has a financial interest in Thalchemy Corp.

References

[1] M. Albert, A. Schnabel, and D. Field, "Innate visual learning through spontaneous activity patterns," *PLoS Computational Biology*, vol. 4, no. 8, 2008.

[2] J. Chang, R. Renteria, M. Kaneko, X. Liu, D. Copenhagen, and M. Stryker, "Development of precise maps in visual cortex requires patterned spontaneous activity in retina," *Neuron*, vol. 48, no. 5, pp. 797–809, 2005.

[3] B. Zhang, "Computer vision vs. human vision," in *Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on*, July 2010, pp. 3–3.

[4] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

[5] R. Memisevic and G. Exarchakis, "Learning invariant features by harnessing the aperture problem," in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013, June 2013.

[6] A. G. Hashmi and M. H. Lipasti, "Discovering cortical algorithms," in *Proceedings of the International Conference on Neural Computation (ICNC)*, October 2010.

[7] Z. Ji and J. Weng, "Wwn-2: A biologically inspired neural network for concurrent visual attention and recognition," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*, July 2010, pp. 1–8.

[8] J. Z. Leibo, J. Mutch, L. Rosasco, S. Ullman, and T. Poggio, "Learning generic invariances in object recognition: Translation and scale," Massachusetts Institute of Technology, Tech. Rep. MIT-CSAIL-TR-2010-061, Dec 2010.

[9] E. Rolls, "Invariant visual object and face recognition: Neural and computational bases, and a model, visnet," *Frontiers in Computational Neuroscience*, vol. 35, Jun 2012.

[10] T. Poggio, J. L. Jim Mutch, L. Rosasco, and A. Tacchetti, "Learning generic invariances in object recognition: Translation and scale," Massachusetts Institute of Technology, Tech. Rep. MIT-CSAIL-TR-2012-035, Dec 2012.

[11] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio, "Unsupervised learning of invariant representations in hierarchical architectures," *CoRR*, vol. abs/1311.4158, 2013. [Online]. Available: http://arxiv.org/abs/1311.4158

[12] D. W. Arathorn, *Map-Seeking Circuits in Visual Cognition: A Computational Mechanism for Biological and Machine Vision*. Stanford, CA, USA: Stanford University Press, 2002.

[13] B. Gregoire and R. Maher, "Map seeking circuits: a novel method of detecting auditory events using iterative template mapping," in *Digital Signal Processing Workshop, 12th - Signal Processing Education Workshop, 4th*, Sept 2006, pp. 511–515.

[14] D. Arathorn, "Recognition under transformation using superposition ordering property," *Electronics Letters*, vol. 37, no. 3, pp. 164–166, Feb 2001.

[15] A. G. Hashmi, "Cortical columns: A non-von neumann computational abstraction," Ph.D. dissertation, University of Wisconsin, Madison, 2011.

[16] D. W. Arathorn, "Introduction to map seeking circuits." [Online]. Available: http://giclab.com/index.html

[17] D. Hubel and T. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, 1962.

[18] T. Kurita, K. Hotta, and T. Mishima, "Scale and rotation invariant recognition method using higher-order local autocorrelation features of log-polar image," in *Proceedings of the Third Asian Conference on Computer Vision-Volume II*, ser. ACCV '98. London, UK, UK: Springer-Verlag, 1997, pp. 89–96. [Online]. Available: http://dl.acm.org/citation.cfm?id=647291.722645

[19] D. Maurer and T. Lewis, "Visual acuity and spatial contrast sensitivity: normal development and underlying mechanisms," in *The Handbook of Developmental Cognitive Neuroscience*, C. Nelson and M. Luciana, Eds. Cambridge, MA: MIT Press, 2001, pp. 237–51.

[20] Mathworks, "Matlab computer vision toolbox." [Online]. Available: http://www.mathworks.com/products/computer-vision/

[21] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders, "The amsterdam library of object images," *International Journal of Computer Vision*, vol. 61, no. 1, pp. 103–112, 2005.

[22] Mathworks, "Detecting a cell using image segmentation." [Online]. Available: http://www.mathworks.com/help/images/examples/detecting-a-cell-using-image-segmentation.html

[23] Adobe, "Adobe photoshop manual." [Online]. Available: http://helpx.adobe.com/photoshop.html