The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

A neuro-dynamic architecture for one shot learning of objects that uses both bottom-up recognition and top-down prediction

Christian Faubel and Gregor Schöner

Abstract— **Learning to recognize objects from a small number of example views is a difficult problem of robot vision, of particular importance to assistance robots who are taught by human users. Here we present an approach that combines bottom-up recognition of matching patterns and top-down estimation of pose parameters in a recurrent loop that improves on previous efforts to reconcile invariance of recognition under view changes with discrimination among different objects. We demonstrate and evaluate the approach both in a service robotics implementation as well as on the COIL database. The robotic implementation highlights features of our approach that enable real-time pose tracking as well as recognition from views where figure ground segmentation is difficult.**

I. Introduction

Typical scenarios in which human users interact with autonomous robots include service robotics, production assistance, robots in care or clinical contexts as well as entertainment robotics. In such settings, robots operate in natural environments which are not specifically designed for the robots' operation, are not metrically calibrated, and may change over time. Critical to meaningful interaction between human users and robots is the capacity of robots to autonomously generate scene representations. This requires segmenting relevant objects, recognizing these objects and estimating their associated pose parameters.

Object recognition in the context of scene representation differs from the general object recognition problem in computer vision. That general problem is still largely unsolved, especially when objects are embedded in natural environments [1]. Object recognition in the context of scene representation is in some ways simpler than the general object recognition problem, while it entails particular challenges in other respects.

To make things concrete, we consider a scenario with our stationary assistant robot CoRA [2], which is equipped with a seven degree of freedom arm and a pan-tilt stereo vision head. A first simplification of the object recognition problem comes from the fact that the number of relevant objects that fit into a shared workspace of the robot and the users is limited. Furthermore, the viewing conditions are typically simplified by the constraint that the robot's action will only be directed to objects that are within reach of the robot's arm. Because the size of that workspace is restricted by the length of the robot's arm, variations of object appearance through scale and perspective transformations are limited. In addition, contextual information about the layout of the scene is often available, for instance, in the form of calibrated camera geometries describing the relation between a camera-centered and a workspace-centered frame of reference. Finally, scene representations can be continuously updated as actions move objects around or introduce new objects. User feedback is available throughout operation.

On the other hand, specific difficulties arise from such interactive scenarios. Assistance robots are confronted with dynamic scenes. During interaction objects may be moved around and may become occluded for short time intervals by a moving arm of the user or of the robot. The robot may move its head in order to attend to varying parts of the scene. Most dramatically, the number of training views that can be provided to learn a new object is necessarily limited. When users add an object to the scene and provide the system with an associated label identifying the object, they may be willing to give a corrective signal from time to time, but only as long as the number of such teaching signals remains quite small, a few per object. An effective object learning system should be able to generalize from a small number of labelled object views. While this problem of learning object representations from a limited number of training examples is felt most acutely in assistance robotics [3], minimizing the amount of training material required to learn objects is relevant in other settings as well [4].

The difficulty of learning to recognize objects based on a small number of views derives from a fundamental problem of computer vision, the fact that any given object may cast a continous and thus infinite number of different two-dimensional images onto the visual sensor. Identifying an object from a small sample of such views is inherently an ill-posed problem, more so even if the object's pose is to be estimated at the same time. Past efforts to address this problem have often taken inspiration from how the human nervous system seems to effortlessly solve the problem. Multiple feature histogramming approaches [5], [6], [7] generated a degree of shift invariance through spatial pooling of feature representations and by learning the feature contributions that are most invariant for an object with respect to the possible remaining transformations. In order to uncover the invariant features, these approaches require a larger number of training examples, however. Feedforward view-based approaches also achieve shift invariance through a hierarchy of pooling stages [8], [9]. Invariance to rotation is only achieved by increasing the number of training views.

An approach to limit the number of different views is to

This work was supported by the German Federal Ministry of Research and Education (BMBF) through the DESIRE project.

Christian Faubel and Gregor Schöner are with Institut für Neuroinformatik, Ruhr-Universität Bochum, 47800 Bochum, Germany christian.faubel@rub.de, Gregor.Schoener@rub.de

actively estimate the transformation an object has undergone relative to the learned view and to thus place the current view into an object centered reference frame [10], [11]. The difficulty of such correspondence-based approaches is, of course, to uncover the transformation the object has undergone, which is far from trivial. We took inspiration from a recurrent architecture proposed by Arathorn, which solves the correspondence problem efficiently through pattern superposition and the cascading of multiple transformations [12]. We transferred this idea to the context of fast object learning by proposing a neuro-dynamic architecture in which bottom-up information converges on a competitive dynamics that selects the recognized object while top-down information converges on a dynamic neural field that estimates pose parameter values. The bottom-up path is based on feature channels, similar to multiple feature histogramming approaches [5], [6], [7]. We compute the features through pooling, both by summing over receptive fields to sample histograms [7] and by max-pooling operations [13] to generate shape templates. While a degree of shift invariance is used to increase robustness, it is limited by restraining the size of the receptive fields and by attaching them to fixed positions on the input image. Multiple feature channels are fused. The top-down path computes in parallel estimates of the transformations between the current and learned representations of an object. Translational and rotational transformations are cascaded.

To enable continuous, online object recognition in dynamically changing scenes we base the estimation and pattern matching processes on the methods of Dynamic Field Theory [14]. Dynamic Field Theory is a theoretical approach to (embodied) cognition that emphasizes that cognitive systems are embedded in the structured and time-varying environments, to which they are adapted. Dynamic Field Theory has been successfully applied in the robotics domain for lower level perceptual tasks such as obstacle or target representation [15], [16]. More recently, we began to apply this approach to the problem of object recognition [17]. Memory traces in Dynamic Neural Fields defined over low-dimensional feature maps were used to represent objects. This approach allowed for fast learning of relatively invariant features, and achieved good recognition performance for a set of 30 objects with about three learning trials on average.

Here we use Dynamical Neural Fields for the competitive estimation of the transformations of object centered reference frames. This enables the object recognition system to update its pose estimations online, so that the system can be coupled to the video-stream of images captured by the robot to track recognized objects. We use discrete dynamical neurons for the competitive selection of the winning memory pattern during matching. The new architecture makes it possible to employ more highly discriminative feature dimensions, which have a lower level of intrinsic invariance. This increases recognition rates at an even smaller number of learning views than our previous effort. Moreover, the new approach also works when the segmentation of objects is made difficult by close contact between different objects.

II. Method

We compute feature histograms over large receptive fields of the visual input. The input for computing the receptive field histogram comes from associated feature maps based on the responses of locally applied non-linear filters. We extract four different feature maps: a hue color map and three orientation maps based on edge images of opponent color channels: black/white, red/green and blue/yellow (see the center images in Figures 1 and 4 for examples of these feature maps). For each of these modalities separate receptive field histograms are computed. Additionally we generate shape templates through a max-pooling operation on the edge image of the black/white color channel.

A. Localized receptive field histograms

In contrast to standard receptive field histogram approaches [5], [6], [7] we do not take the whole visual input as a single receptive field for computing the histogram. Instead we use receptive fields with spatial Gaussian profiles, which provide more activation at their center and less near their boundaries. Specifically, we compute the localized receptive field histogram $\mathbf{h}$ from the two dimensional feature map $\mathbf{F}$ of size $\hat{X} \times \hat{Y}, \hat{X} = \hat{Y} = 64$. Each discrete point in this map F_{xy} has a feature value $f \in [-1, 0, 1 \ldots f_{\max})$ along the feature dimension that corresponds to the response of the non-linear filter and encodes, for example, the hue value or the orientation of an edge at this position. The resolution along the feature dimension of the histogram is smaller than the corresponding resolution of the feature map. The feature space is subsampled. The range $J = [jr, jr+1, .., (j+1)r)$ depends on the position j in the feature histogram and on the ratio of the histogram's and the feature map's resolution $r = \frac{f^{\max}}{j^{\max}}$.

The Gaussian profile of the receptive field is computed as a discretized Gaussian $G_{x,y}(x^c, y^c, \sigma)$ that has as parameters the receptive field's center coordinates x^c, y^c and its spatial extent σ. The formula to extract a localized receptive field histogram is then as follows:

$$h_j(x_c, y_c) = \sum_{x=1}^{\hat{X}} \sum_{y=1}^{\hat{Y}} \begin{cases} G_{xy}(x_c, y_c, \sigma) & \text{if } F_{xy} \in J \\ 0 & \text{if } F_{xy} \notin J \end{cases} \quad (1)$$

Figure 1 illustrates the activity of a localized receptive field histogram of hue centered on an object.

B. Localized receptive field histograms of hue

To exemplify how pattern matching is combined with the estimation of the shift of an object centered reference frame in a recurrent process, we first focus in this section on the feature dimension "hue". The same basic mechanisms is used for the other feature dimensions discussed below, whose fusion occurs through the dynamics of recognition and estimation (section II-D).

We compute the hue feature map from an hsv-image by assigning the hue value to those pixels that are above a threshold of saturation, while assigning -1 to pixels with saturation below the threshold. The localized receptive field

Fig. 1. Localized receptive field histogram of hue computation for an example object. On the left is the input image and in the middle the hue feature map with a visualization of the receptive field's Gaussian profile: the strength of which is displayed through intensity and its center and width are visualized with the little grey cross and with the grey circle. On the right is shown the receptive field histogram, computed from the hue feature map. Each receptive field bin is colored with the hue value it represents.

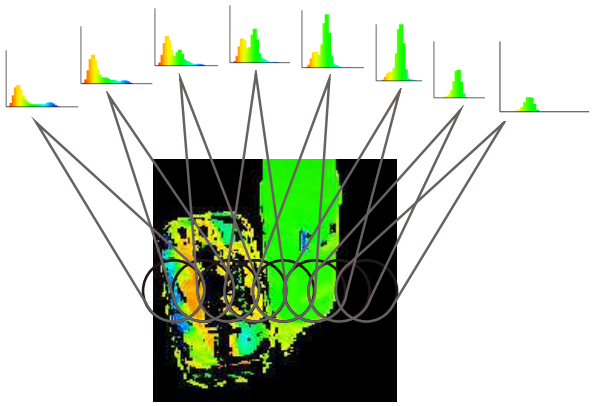

Fig. 2. Grid of localized receptive fields distributed over a feature map that contains two objects. Pictured is a single line of receptive fields and their associated histograms. The receptive fields equally cover the whole image, but in order to avoid an overly cluttered picture only one line is shown.

histogram of hue is invariant to in-plane rotations of objects and partly also invariant to depth rotations. Because we introduce shift variance through the localized receptive field, the shift is the only variance generating variable that has to be estimated in order to achieve an invariant feature representation.

During learning, the Gaussian weight matrix $\mathbf{G}$ is centered on the object and the localized receptive fields are computed by sampling the feature map (see also Figure 1).

TABLE I

AN OVERVIEW OF THE NOTATION WE USE.

receptive field histograms vectors	$\mathbf{h}$
backward pathway vector/matrix	$\mathbf{b}$, $\mathbf{B}$
forward pathway vector/matrix	$\mathbf{f}$, $\mathbf{F}$
weight vector/matrix	$\mathbf{w}$, $\mathbf{W}$
similarity/correlation vectors/matrices	$\mathbf{c}$, $\mathbf{C}$
second transformation for a cascade	$\mathbf{b}'$, $\mathbf{f}'$
feature dimension superscript	$\mathbf{b}^{\text{hue}}$, $\mathbf{b}^{\text{y}}$, $\mathbf{b}^{\text{u}}$, $\mathbf{b}^{\text{v}}$, $\mathbf{B}^{\text{shape}}$
transformation superscripts shift/rotation	$\mathbf{w}^{\text{s}}$, $\mathbf{w}^{\text{r}}$
transformation variables shift	 x, y; number of shifts X, Y

The extracted localized receptive field histogram, $\mathbf{h}^{\text{hue}}(x^c, y^c)$, is then stored in a memory vector, $\mathbf{m}_l^{\text{hue}}$, associated with object, l, which is an element of the complete memory buffer, $\mathbf{M}^{\text{hue}}$ (see Table I for the notation we use). For an illustration see also the top right of the schema in Figure 3.

During recognition, the input image is subsampled by a grid of localized receptive fields equally distributed over the image at $X \times Y$ locations ($X = Y = 16$). For each receptive field we obtain a feature histogram $\mathbf{h}^{\text{hue}}(x, y)$ that is used for matching (see Figure 2 and the lower left of the schema on Figure 3).

We exploit the ordering property of pattern superposition [12] and design the processes of estimating the shift of the reference frame (and analogously the process of pattern matching) as a recurrent computation. This means that all shifted vectors of input are superposed (and similarly all pattern memories are superposed). The contribution of each shifted input vector is weighted with a dynamical activation variable that reflects how strongly the associated shift value matches the current estimate of shift (and similarly, each memory vector is weighted with an activation variable that reflects how strongly the associated memory currently matches the input). These weights evolve in time so that the superpositions converge (see schema in Figure 3).

Initially all weights for building the sum of the memory patterns $\mathbf{w}_p$ are set to be equal. Instead of matching with each object at each location we match a weighted sum of memory patterns at each location. Of course this does not tell us where a specific object is but it gives us a spatial similarity with the weighted memory pattern. We use this spatial similarity matrix $\mathbf{C}^{\text{s,hue}}$ to estimate where the object centered reference frame might have shifted. This pathway is the backward pathway (right part on Figure 3) and the weighted sum of the memory patterns is the vector $\mathbf{b}^{\text{hue}}$:

$$\mathbf{b}^{\text{hue}} = \sum_{l}^{L} w_l^p \mathbf{m}_l^{\text{hue}} \tag{2}$$

The shift of the object centered reference frame is initially also unknown as all patterns extracted at all locations have equal a priori probability to best match one of the memory items. Therefore, all elements of the weight vector $\mathbf{W}^{\text{s}}$ are initially set to equal values.

$$\mathbf{f}^{\text{hue}} = \sum_{x,y}^{X,Y} W_{x,y}^{\text{s}} \mathbf{h}^{\text{hue}}(x, y) \tag{3}$$

Equation 3 constitutes the forward pathway (left part of Figure 3). The forward vector, $\mathbf{f}^{\text{hue}}$, formed this way is then compared with each memory item. This gives us a similarity vector $\mathbf{c}^{\text{p,hue}}$ which we use to estimate which object is present in the input image:

$$c_l^{\text{p,hue}} = -\|\mathbf{f}^{\text{hue}} - \mathbf{m}_l\| \tag{4}$$

At the level of the pattern match we get a pattern similarity measure by comparing each memory item with the weighted sum of transformed inputs and at the level of the shift of the object centered reference frame we get a spatial similarity

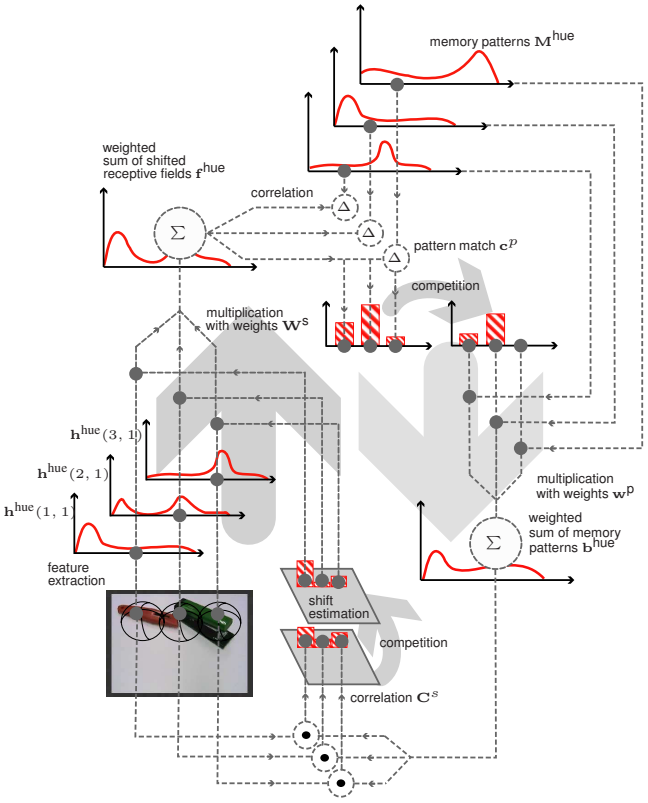

Fig. 3. Recognition with localized receptive fields of hue. The schema shows the recurrent process of shift estimation and pattern recognition. On the left side the forward pathway is sketched, the localized receptive fields are shifted over the image to build the forward vector $\mathbf{f}^{\text{hue}}$ to be matched with each memory pattern. On the right side the backward path is depicted where the vector $\mathbf{b}^{\text{hue}}$ is constituted through the weighted sum of memory patterns.

measure by comparing the extracted input with the weighted sum of all memory patterns:

$$C^{\text{s,hue}}_{xy} = \mathbf{b}^{\text{hue}} \cdot \mathbf{h}^{\text{hue}}(x,y) \tag{5}$$

Just using the result of these correlations, $\mathbf{C}^{\text{s,hue}}$ and $\mathbf{c}^{p,\text{hue}}$, as weights, $\mathbf{W}^s = \mathbf{C}^{s,\text{hue}}$, and $\mathbf{w}^p = \mathbf{c}^{p,\text{hue}}$, for building the next weighted sum in each pathway may already lead to convergence. To ensure that the system converges to single solutions, a competitive process is needed. Arathorn uses a competitive function that preserves the magnitude of the largest competitor and diminishes all others [12]. This function works well when the input is not changing while the recurrent recognition/estimation process runs. In contrast, working on a video stream of images, in which input may change and fluctuate during the recurrent computation requires stability against fluctuating input, not provided by Arathorn's competitive function. Instead of Arathorn's function we therefore use Dynamic Neural Fields for the robust estimation of the possible transformation of object centered reference frames. Similarly, we use assemblies of dynamic neurons for the competitive process at the level of the pattern match. At both levels, self excitation provides stability.

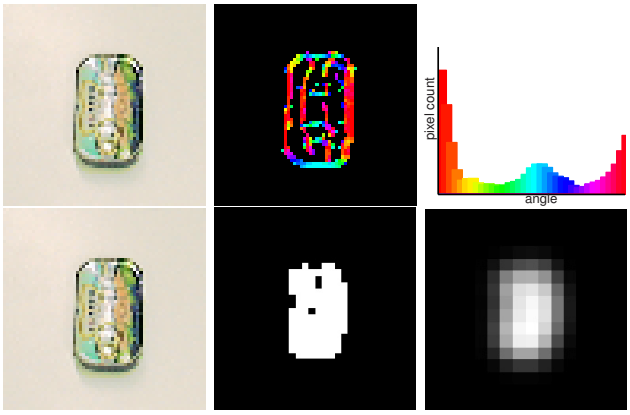

Fig. 4. The top of the Figure shows the orientation map and the associated localized receptive field histogram. The bottom part shows the shape description obtained through maximum pooling.

C. Other feature channels

In addition to the hue feature map, we compute three other feature maps from the responses of steering filters [18] applied to the three color channels of a YCrCb color space. These feature maps contain orientation information and, just like for the hue feature map, we extract localized receptive field histograms from these maps. In addition to these four types of localized receptive field histograms, we also compute a shape description through a maximum pooling of the energy from the steering filter responses. See Figure 4 for examples of these additional feature channels.

D. Cascading of transformations

Both the localized receptive fields of oriented edges and the shape description vary with object rotation and support the estimate of rotation. For the localized receptive field histograms of oriented edges, object rotation induces a phase shift. Computing rotated responses thus corresponds to shifting the histograms. For the shape description we compute rotated versions by applying a rotation matrix to the shape image.

The estimation of shift and of rotation of an object can be cascaded [12]. First the shift transformation is estimated. The weighted sum of the forward pathway $\mathbf{f}^{\text{y,u,v}}$ and $\mathbf{F}^{\text{shape}}$ is used as input to the rotation transformation. Here, analogously to the shift transformation, weighted sums of rotated patterns $\mathbf{f}'^{,\text{y,u,v}}$ and $\mathbf{F}'^{,\text{shape}}$ are formed that are then used for the pattern match. In the backward pathway, the estimated rotation is conversely applied to back-transform the weighted sum of memory patterns $\mathbf{b}'^{,\text{y,u,v}}$ and $\mathbf{B}'^{,\text{y,u,v}}$. These back-transformed versions are then used as weighted sums of memory patterns for the shift estimation.

E. System fusion, transformation estimation modules and pattern matching

The role of the transformation estimation modules and the pattern matching module is to accelerate this process through selection, to force them to converge to single solutions and to stabilize the selected estimates and winning patterns. We

use Dynamic Neural Fields for the estimation of the spatial shift and of the rotation. For the pattern matching module, we use assemblies of Amari type dynamic neurons. These all have a two layer architecture in common. The first layer receives the input and has relatively weak lateral interactions so that its output resembles the input with weak suppression of outliers. The output of the first layer feeds into the second layer which is more strongly interaction driven and has stronger competition. This makes that only a a single peak of activation survives (for the estimation field) and only a single neuron remains active (for the pattern matching system, see also Figure 5).

A simple peak detection mechanism at the level of the second layer gates the read-out of either the first or second layer to the output of the estimation or pattern match layer. As long as no peak has yet emerged in the second layer, the first layer is read out. Once a peak has formed, the second layer is used for read-out. The rationale behind this architecture is that at the beginning of the pattern match and transformation estimation, it is useful that the weighted sums are still relatively unselective, so that many transformations and many memories have a vote. This avoids that the system is driven by an accidental dominance of individual features early in the process. Once the recurrent converges, it is nevertheless desirable to switch into a decision making mode, in which a single solution is selected.

The system is based on five different feature dimensions which need to be fused. Fusion is done in the two kinds of modules at several levels. Fusion takes place at the input level of the different transformation estimation modules. The contributions of different feature dimensions are weighted and summed up to form the combined input to the transformation estimation. Fusion also takes place at the output level of these estimation modules, because each output is applied as a weight factors to all feature dimensions. Similarly the different feature dimensions are fused in the pattern matching module. The feature dimensions are weighted and summed at the input level to the pattern matching module.

1) Spatial shift and rotation estimation: The spatial sift and the rotation estimation module only differ in their dimensionality, their parameter settings and the inputs they receive. Because they operate identically otherwise, we only explain the shift estimation module in more detail. The first layer Dynamic Neural Field for the shift estimation receives as input the correlations computed for each feature dimension (hue, black/white edges, blue/yellow edges, red/green edges, shape). To fuse these different dimensions, the correlations are first made mean-free (subtracting the average correlation across all possible shifts), multiplied with a constant weight factor and summed:

$$\begin{aligned}\mathbf{S}^s &= \gamma^{\text{s,hue}}\mathbf{C}^{\text{s,hue,norm}} + \gamma^{\text{s,y}}\mathbf{C}^{\text{s,y,norm}} \\ &+ \gamma^{\text{s,u}}\mathbf{C}^{\text{s,u,norm}} + \gamma^{\text{s,v}}\mathbf{C}^{\text{s,v,norm}} \\ &+ \gamma^{\text{s,shape}}\mathbf{C}^{\text{s,shape,norm}}\end{aligned} \tag{6}$$

The weights $\gamma^{\text{s,hue}} = 20, \gamma^{\text{s,y}} = 100, \gamma^{\text{s,u}} = 0, \gamma^{\text{s,v}} = 0, \gamma^{\text{s,shape}} = 10$ were empirically determined to reflect the frequency with with correlations occur in the different channels. For example, for most objects there are more pixels with a color value than with an edge value so that the color input is usually stronger than the edge input. The color edges where not used because for the purpose of shift estimation they are redundant with black and white edges.

Conceptually, the Dynamic Neural Field is a continuous representation of the parameter to be estimated, although the numerical implementation discretely samples the parameter. In this sense, the input $\mathbf{S}^s$ is a continuous function over space and time, $s_s(x, y, t)$. For the spatial shift, the activity at a certain location at a moment in time, $u(x, y, t)$, represents the estimate of the shift of an object with respect to the object centered reference frame. The field equation of the first layer field is:

$$\begin{aligned}\tau_1 \dot{u}_{s,1}(x,y,t) &= -u_{s,1}(x,y,t) + h_1 + s_s(x,y,t) \\ &+ \int\int W_{s,1}(x-x', y-y') \\ &\quad \sigma(u_{s,1}(x',y',t))dx'dy'\end{aligned} \tag{7}$$

The second field receives the output from the first layer field as input. It has different interaction parameters but is governed by an equation of the same form. It is more strongly interaction dominated: its kernel, $W_{s,2}$, has stronger excitatory and inhibitory weights than the kernel of the first layer field, $W_{s,1}$ (see also Figure 5 for an illustration). We smooth the output from the first layer with a Gaussian kernel before we use it as input to the second layer. Because the second layer receives the output of the first layer, no peaks form in the second layer before there is suprathreshold activation in the first layer. The timescale τ_2 of the second layer is slower than the timescale of the first layer τ_1, so that it takes more time for a peak to build up there. Until a peak has formed in the second layer, the output of the first layer is taken as output of the shift estimation module. The output, $\mathbf{\Sigma}^{s,i}$, is written in discrete notation to index its role as discrete weights, $\mathbf{W^S}$, for the summed feature vectors, $\mathbf{f}, \mathbf{F}$, and corresponds to the sigmoided activation, $\sigma(u_{s,i}(x, y, t))$, for each layer with index i.

$$\mathbf{\Sigma}^{s,i} = \sigma(u_{s,i}(x,y,t)) \tag{8}$$

When a peak in the second layer is detected with a peak detector, $\sigma(u_{\text{p},s,2}) > 0$, the sigmoided activation of the second layer is used as output of the module, otherwise the output of the first layer is used.

$$\tau \dot{u}_{\text{p},s,2} = -u_{\text{p},s,2} + h + \int \sigma(u_{s,2}) \tag{9}$$

$$\mathbf{W}^{s,\text{pre}} = (1 - \sigma(u_{\text{p},s,2}))\mathbf{\Sigma}^{s,1} + \sigma(u_{\text{p},s,2})\mathbf{\Sigma}^{s,2} \tag{10}$$

$$\mathbf{W}^s = \frac{\mathbf{W}^{s,\text{pre}}}{\|\mathbf{W}^{s,\text{pre}}\|_1} \tag{11}$$

2) Pattern matching module: The pattern matching module is realized through two layers of discrete dynamic neurons. Within each layer, the number of neurons corresponds to the number of objects the system can recognize. An individual neuron's activity indexes the presence of an associated

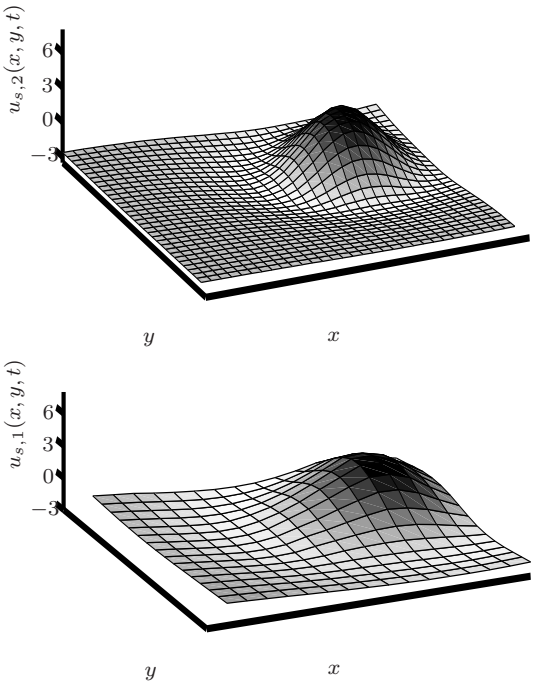

Fig. 5. The two layer shift estimation module. The bottom of the Figure shows the broader activation $u_{s,1}$ of the first layer, that projects its output to the second layer. The second layer's activation $u_{s,2}$ is figured on top of the first layer.

object. Like in the two layered architecture of the estimation fields, the first layer is input dominated allowing for multiple neurons being suprathreshold, whereas the second layer is interaction dominated so that only a single neuron can have suprathreshold activity. Both layers are initially set to a resting level, h_1, so that even in the presence of input their neural activity is *off*. Only when a peak is detected on the level of the second layer of the rotation estimation, $\sigma(u_{\text{peak},r,2}) > 0$, are these neurons released to a higher resting level, $h_{1,\text{boost}}$, where activity can pass threshold.

We remove the mean and fuse the inputs from the different dimensions by summing them up with different weights:

$$\begin{aligned}\mathbf{s}^{p,1} &= \gamma^{p,\text{hue}}\mathbf{c}^{p,\text{hue,norm}} + \gamma^{p,y}\mathbf{c}^{p,y,\text{norm}} \quad (12)\\ &+ \gamma^{p,u}\mathbf{c}^{p,u,\text{norm}} + \gamma^{p,v}\mathbf{c}^{p,v,\text{norm}} + \gamma^{p,\text{shape}}\mathbf{c}^{p,\text{shape,norm}}\end{aligned}$$

In contrast to the shift estimate, here the color edge distributions also contribute to recognition,. The weights are chosen as follows: $\gamma^{p,\text{hue}} = 27, \gamma^{p,y} = 32, \gamma^{p,u} = 28, \gamma^{p,v} = 28, \gamma^{p,\text{shape}} = 10$

The first layer receives this weighted sum, $\mathbf{s}^{p,1}$, as input and feeds its activity to the second layer where it serves as input. Again, time is conceptually continuous. Because the equation for the second layer only differs in parameter values and in the source of input, we show only the equation for the first layer.

$$\begin{aligned}\tau_1\dot{u}_l^{p,1}(t) &= -u_l^{p,1}(t) + h_1 - \gamma_{\text{inh}}\sum_{l'=1,l'\neq l}^{L}\sigma(u_{l'}^{p,1}(t))\\ &+ \gamma_{\text{exc}}\sigma(u_l^{p,1}(t)) + s_l^{p,1} \quad (13)\\ &+ \sigma(u_{\text{peak,r,2}})h_{1,\text{boost}} \quad (14)\end{aligned}$$

When reading out the weights for the calculation of the weighted memory vectors, we check whether the second layer is suprathreshold. If this is the case, the second layer's output is used. Otherwise, the first layer's output is used.

$$\begin{aligned}\mathbf{w}^{p,\text{pre}} &= (1-\sigma(\max(\mathbf{u}^{p,2})))\sigma(\mathbf{u}^{p,1}) \quad (15)\\ &+ \sigma(\max(\mathbf{u}^{p,2}))\sigma(\mathbf{u}^{p,2}) \quad (16)\\ \mathbf{w}^{p} &= \frac{\mathbf{w}^{p,\text{pre}}}{\|\mathbf{w}^{p,\text{pre}}\|_1} \quad (17)\end{aligned}$$

F. Backprojection and segmentation

Reliable segmentation in a purely feed forward way is very difficult to implement. Segmentation becomes much easier when object knowledge is available. In the current framework, object knowledge is integrated through the backward pathway. If the estimated rotation and shift are inversely applied to the shape description, the result, $\mathbf{B}^{\text{back}}$, can be used to filter out those parts of the input image that do not belong to the object. Once the system has converged to a decision for an object, the associated shape is back-projected, which improves pose estimation as it effectively cuts away input information incompatible with the recognized object.

III. RESULTS

A. Assessment of the baseline performance

To first assess the baseline performance of our new system we use the setting of our previous system [17]. Each object is trained in a single view, centered and in a canonical orientation with the longer of its major axes aligned vertically For testing, we vary the object pose to three different locations and three different orientations each. Thus, each object is tested in nine different poses. In total, the test of thirty objects at nine poses results in 270 tests. With a single training view, the system reaches an overall recognition performance of 90 percent in this test scenario.

B. Assessment of the baseline performance on the COIL-Database

Although we are more interested in realistic robotic scenarios, we tested our system on the COIL database [19]. We considered only the first 30 objects and achieved a recognition rate of 85 per cent with two training views and of 94 per cent with four training views. Each additional view was treated as an independent object. If any of the these matched closest, it was counted as a successful recognition. This performance may be compared to the results of [8] who achieved 80.1 per cent recognition rate for two views and 89.9 per cent for four training views on the first 30 objects of the COIL database.

Fig. 6. Two objects that touch in the image. The images show examples of test scenes.

C. Assessment of performance with two items that touch in the image

In order to assess the performance of the system for more complex scenes we tested it in situations in which two objects touch in the scene (see Figure 6). In such cases, a simple segmentation based, for instance, on knowledge about the background, will not be sufficient to segment the scene into two objects. We did not test all possible combination of two objects from the set of 30 objects, but chose an arbitrary drawing of object pairs from the object sets such that each object was contained in at least one test. Currently, in the absence of an active process of scene representation, which of the two touching objects is recognized is a reproducible function of the quality of the match. The recognition rate for that first recognized object was 76 percent over all trials. The associated pose estimate was correct for 72 percent of trials. This number includes those trials on which recognition was incorrect. Pose estimation is correct on 96 percent of all trials on which recognition was correctly achieved.

D. Tracking and distractor robustness

Once the system has recognized an object, this decision is stabilized by the winner-takes-all connectivity of the pattern matching module. The decision for a specific object structures the input the system receives through the backward pathway, and renders those parts of the image which have a match with the specific object more salient than others. This supports robustness against distractors. The stability property of the localized peaks in the estimation fields representing the pose of the object also supports robustness against distractors. On the other hand the shift and rotation estimation fields are adjusted so that they can still track changing input. These two properties — distractor robustness and tracking — are illustrated in Figure 7: the operator's hand acts as a distractor the system must be stabilized against, but his hand also moves the object and the system has to track the object. This example also gives an idea how scene representation and the ability to track objects can support the recognition process. Tracking objects over time can for example render the problem of occluding objects easier, if those occlusions happen over time.

IV. DISCUSSION

The present system achieves high recognition rates of everyday objects that are presented at varying poses. This was tested by varying position, rotation and the viewing angle systematically. Importantly this performance was achieved

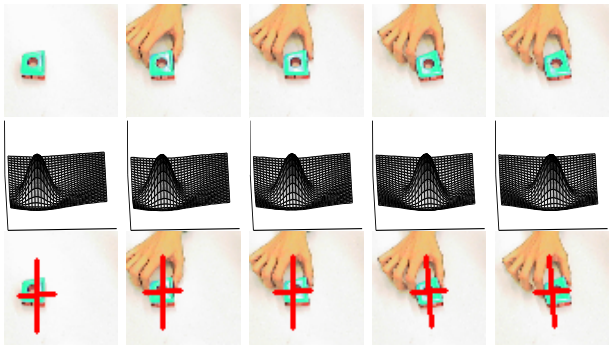

Fig. 7. Tracking and distractor robustness. Top row shows the input image, in the middle the field activity of the spatial decision field is depicted, and in the bottom row the pose estimation results is overlaid on the input image.

on the basis of only a single training view. Even when segmentation is made more difficult by two objects touching recognition rates remain reasonable. On the COIL database our system achieves high performance rates based on learning only two or four views per object.

A. Comparison to view-based feedforward models of object recognition

View based feedforward models [8], [9] usually need a larger number of training views in order to achieve invariant representations. Most feed-forward object recognition systems focus on the recognition portion of the problem of object perception, determining object identity. Position and pose information about the object is usually lost along the feedforward stream in order to achieve position invariant recognition. If two or more objects enter the feedforward stream, recognition of one of these objects may be successful and a binding problem may be avoided [20]. Feedforward models do not account, however, for how object pose information may be linked to object identity. If pose parameters must be associated with the object's identity, the problem of binding these two object properties together arises in feedforward models. In human vision, pose information is readily associated with object identity [21]. Moreover, using object recognition in robotic scenarios makes pose information critical as it enables robots to plan grasping or manipulatory movements.

B. Comparison to map-seeking circuits

The principles of our architecture are similar to those underlying Arathorn's proposal of a map seeking circuit [12]. Differences arise at three levels. The first difference is our choice of features. Instead of relying on edge image templates we sample local distributions like the hue color or the orientation of edges and generate shape templates. Through this choice of features we achieve a higher degree of invariance with respect to those transformations that we are not simultaneously estimating. At the tilt angle with which we are looking at the scene, an in-plane rotation of the objects with respect to the table plane deviates significantly from a rotation in the image plane. While Arathorn solves this problem with a complete three dimensional model of

a target object that requires estimating also the in-depth rotation, it remains unclear how an object recognition system could acquire such a three dimensional model on its own.

A second difference lies in our treatment of feature fusion. Arathorn developed his approach for a single feature modaility, edge images. We use multiple feature channels which we fuse by weighting the different dimensions and linearly combining them within the computational loop between pose estimation and pattern matching.

The third major difference lies in our use of Dynamic Neural Fields for the estimation of the image transformations and, similarly, of competitive neuronal dynamics for the pattern matching process. This difference is associated with a more fundamental difference in outlook. While the competitive mechanism of Arathorn's map-seeking algorithm is designed for open-loop computation in which image input is not updated during the computational cycle, we have shown how Dynamic Neural Fields make it possible for the estimation and recognition processes to be continuously updated by new sensory information in closed loop.

C. Computational time

Although the structure of the numerical algorithm would enable implementation on parallel hardware, we have realized our software architecture for standard multi core i386 processors. On a Core-Duo 2.1 Ghz with Intel's MKL and IPP-libraries, the computations of the features and of the correlations that are used as inputs to the dynamic neural fields take about 300 ms. The dynamic fields are iterated in a parallel thread. Their execution takes 80 ms. It may take up to 8 seconds to establish a stable and time invariant decision about object pose and identity. At this time, all estimates are in an attractor state. The transient solutions typically indicate the correct decision already after a small number of iterations of the dynamic fields (e.g., 20 cycles or less than 2 seconds), at which time the neuron with the strongest activation matches the neuron that will ultimately win the competition. Thus, there is no need to await full relaxation before reading out the results. A related feature is the fact that even during recognition the system is continuously updating its input and can thus track time-varying input during recognition.

V. CONCLUSION

We have presented a pervasively neuro-dynamic architecture for object recognition that supports scene representation for an autonomous robot. We reach competitive performance on a standard benchmark of object recognition. More importantly, we show that our approach also works in more difficult conditions, where segmentation of objects is not trivial. Furthermore our system has important stability properties, enabling the tracking of objects and their pose parameters. This is especially relevant in robotic scenarios where the robots own movement as well as object movements induced by human users require continuous online updating of the scene.

REFERENCES

[1] N. Pinto, D. D. Cox, and J. J. Dicarlo, "Why is real-world visual object recognition hard?" *PLoS Computational Biology*, vol. 4, no. 1, pp. e27+, January 2008. [Online]. Available: http://dx.doi.org/10.1371/journal.pcbi.0040027

[2] I. Iossifidis, C. Theis, C. Grote, C. Faubel, and G. Schöner, "Anthropomorphism as a pervasive design concept for a robotic assistant," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 4, 2003, pp. 3465–3472vol.3.

[3] J. Steil, F. Röthling, R. Haschke, and H. Ritter, "Situated robot learning for multi-modal instruction and imitation of grasping," *Robotics and Autonomous Systems*, vol. 47, pp. 129–141, 2004.

[4] L. Fei-Fei, R. Fergus, and P. Perona, "A bayesian approach to unsupervised one-shot learning of object categories," in *Proceedings of the Ninth IEEE International Conference on Computer Vision 2003*, 2003.

[5] M. Swain and D. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, pp. 11–32, 1991.

[6] B. Schiele and J. Crowley, "Object recognition using multidimensional receptive field histograms," 1996. [Online]. Available: citeseer.comp.nus.edu.sg/35791.html

[7] B. W. Mel, "Seemore: Combining color, shape, and texture histogramming in a neurally inspired approach to visual object recognition," *Neural Computation*, vol. 9, pp. 777–804, 1997.

[8] H. Wersing and E. Koerner, "Learning optimized features for hierarchical modelling of invariant object recognition," *Neural Computation*, 2003.

[9] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 994–1000. [Online]. Available: http://portal.acm.org/citation.cfm?id=1068508.1069194

[10] S. E. Palmer, "The psychology of perceptual organization: a transformational approach," *Human and machine vision*, pp. 269–339, 1983.

[11] B. A. Olshausen, C. H. Anderson, and D. C. Van Essen, "A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information," *J. Neurosci.*, vol. 13, no. 11, pp. 4700–4719, November 1993. [Online]. Available: http://www.jneurosci.org/cgi/content/abstract/13/11/4700

[12] D. W. Arathorn, "Map-seeking circuits in visual cognition: A computational mechanism for biological and machine vision," Ph.D. dissertation, Stanford, CA, USA, 2002.

[13] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 2000.

[14] G. Schöner, "Dynamical systems approaches to cognition," in *Cambridge Handbook of Computational Cognitive Modeling*, R. Sun, Ed. Cambridge, UK: Cambridge University Press, 2008, pp. 101–126.

[15] C. Engels and G. Schöner, "Dynamic fields endow behavior-based robots with representations," *Robotics and autonomous systems*, vol. 14, pp. 55–77, 1995.

[16] E. Bicho, P. Mallet, and G. Schöner, "Target representation on an autonomous vehicle with low-level sensors," *The International Journal of Robotics Research*, vol. 19, pp. 424–447, 2000.

[17] C. Faubel and G. Schöner, "Learning to recognize objects on the fly: a neurally based dynamic field approach," *Neural Networks*, vol. 21, pp. 562–576, 2008.

[18] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, 1991. [Online]. Available: http://citeseer.ist.psu.edu/freeman91design.html

[19] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-100)," Department of Computer Science, Columbia University, Tech. Rep., February 1996.

[20] M. Riesenhuber and T. Poggio, "Are cortical models really bound by the "binding problem"?" *Neuron*, vol. 24, pp. 87–99, 1999.

[21] J. M. Henderson and A. Hollingworth, "High-level scene perception," *Annual Reviews of Psychology*, vol. 50, pp. 243–271, 1999.