

Interactive Online Learning¹

G. Heidemann, H. Bekel, I. Bax, and H. Ritter

*Neuroinformatics Group, Bielefeld University, P.O. Box 10 01 31, D-33501 Bielefeld, Germany
e-mail: {gheidema, hbekel, ibax, helge}@techfak.uni-bielefeld.de*

Abstract—We present a computer vision system for object recognition, which is integrated in an augmented reality setup. The system can be trained online to recognize objects in an intuitive way. The augmented reality gear allows interaction using hand gestures for the control of displayed “virtual menus.” The underlying neural recognition system combines feature extraction and classification. Its three-stage architecture facilitates fast adaptation: in a fast training (FT) mode, only the last stage is adapted, whereas complete training (CT) rebuilds the system from scratch. Using FT, online acquired views can be added at once to the classifier, the system being operational after a delay of less than a second, though still with reduced classification performance. In parallel, a new classifier is trained (CT) and loaded to the system when ready

INTRODUCTION

The view-based approach to computer vision has the great advantage that visual knowledge can be conveniently acquired from samples (e.g., [1]). However, in most applications, the acquisition and processing of training views is limited to a one-time training phase. Here, we propose a system that is capable of “anytime” online learning; i.e., novel views can be acquired and added to the neural classifier whenever either a new object should be added or classification of an already-known object should be improved. To realize online learning in a real-time application, two key abilities must be implemented: (i) the classifier must allow very fast reconfiguration to the new view database to keep the system running, if necessary, at the cost of temporarily reduced classification performance; (ii) an intuitive, natural user interface has to be provided.

First, we will describe the context of the system and the learning of novel objects from the point of view of a user. Then, the underlying technique of both the neural classifier and the user interface will be outlined. The final sections give an overview of the results and future prospects.

AUGMENTED REALITY SCENARIO

The work described is part of an experimental augmented reality system intended to be a “personal assistant.” As a sample task, an office scenario was chosen. The user wears augmented reality gear (AR gear): two cameras and a display are mounted to a helmet; the cameras view the area in front of the user (Fig. 1). The camera image is displayed by a “see-through loop”; in addition, augmentations of the system can be overlaid on the real image.

¹ This article was submitted by the authors in English.

The task of the system is the interactive memorization and retrieval of objects relevant to office tasks, such as pencils, sharpeners, or tape. Once an object has been learned and given a class label by the user, the system should keep track of the object and be able to answer queries such as “Where is the pencil?” by display of augmentations highlighting the object. Here, we concentrate on the first part of the task, i.e., acquisition and classification of objects.

USER INTERFACE FOR ONLINE LEARNING

Interaction with the system is carried out by virtual buttons and menus overlaid on the real image (Fig. 2). Since the AR gear is intended to be a mobile system, conventional input devices such as a keyboard or mouse



Fig. 1. User wearing the augmented reality gear, pointing at an object of the office scenario.

Received October 25, 2004



Fig. 2. User interface for interactive online learning. The user is pointing at a falsely classified object, which is highlighted as a feedback. To the right, virtual menu buttons are being displayed.

are not feasible. Instead, input by gesture and speech is used. A button visible in the display can be pressed either gesturally by moving a finger onto the button and making a “pressing” movement or by speech input. Both modalities complement each other, a particularly useful feature since visual input can be disturbed by changes of illumination and speech input, by noise. In case one modality fails, buttons can still be pressed using the other one. By this means, parameters, e.g., of the underlying skin color segmentation for fingertip recognition, can be readapted online.

A basic functionality is the display of object classification results (Fig. 2). The user thus sees which objects are misclassified or still completely unknown. In this case, online learning is activated using the menus. To reference the object in question, the user now points at the object, and, as a feedback, the system displays the pointing direction and the referenced object by an overlay (cone and object highlighting in Fig. 2). For online training, the system now makes images of the object. The user turns the object to different positions to provide a variety of views and triggers image acquisition again by pressing a virtual button.

After sample acquisition, a class label must be given by speech input. After a short training time of less than one second, a first version of the newly trained classifier is loaded and the system is operational again. Invisible to the user, in a parallel thread, a better version of the classifier is trained over about one minute. When ready, the improved classifier is loaded automatically. The underlying neural methods are described in the next section.

IMAGE PROCESSING ARCHITECTURE

We restrict the description of the processing architecture to the basic vision functionalities: object loca-

tion and classification and pointing gesture recognition. A complete outline including functionalities such as the virtual menu control is outside the scope of this paper.

The attentional subsystem has two branches: (i) a goal-specific module performs skin color segmentation for the fast location of hands pointing at objects and fingertips pointing at buttons; (ii) domain-independent mechanisms search for “conspicuous” regions that might indicate the presence of objects. For the latter, three complementary methods are used. The overall “interestingness” of regions is judged by their informational content as proposed in [2]. Since artificial objects often exhibit symmetries, the symmetry detector of [3] is implemented as a second mechanism. Finally, to detect edges and corners, the method of [4] is applied, which is particularly useful for human-made objects. The resulting feature maps are integrated to obtain an overall saliency map indicating the most likely object locations by the adaptive weighting scheme proposed in [5].

Corresponding to the two branches of ROI detection, two classifiers are applied for (i) classification of skin-colored locations into the categories “pointing hand”/“pointing fingertip”/“other” plus direction detection for pointing hands and (ii) classification of possible object locations into one of the trained classes or a garbage class. The underlying classification system is outlined in the next section.

NEURAL CLASSIFICATION

The neural architecture applied for the representation of objects relies on three processing stages, which implement local principal component analysis (LPCA) for feature extraction in combination with a bank of neural classifiers (Fig. 3). The architecture is called VPL, short for vector quantization, PCA, LLM, as will become clear in the following. The main advantage of the VPL classifier for the current purpose is that adaptation of the last stage can be carried out very fast using existing features represented in the first two stages. These features can be trained offline in a slower process.

At each location found by the attentional subsystem, a window of predefined size is extracted and regarded as a vector. During object learning, windows are sampled at the location indicated by a user’s pointing gesture. Internally, each shot of an object is artificially distorted and scaled to obtain a set of several views. Thus, a broader variety of appearances can be covered by a single shot.

The set of all windows (old and new) is the training database. In its first stage (“V”), the VPL classifier is trained unsupervised by vector quantization carried out on the raw windows by the algorithm proposed in [6]. Thus, a first partitioning of the raw data is achieved. The second stage (“P”) attaches to each of the reference vectors a single-layer feedforward neural net for the

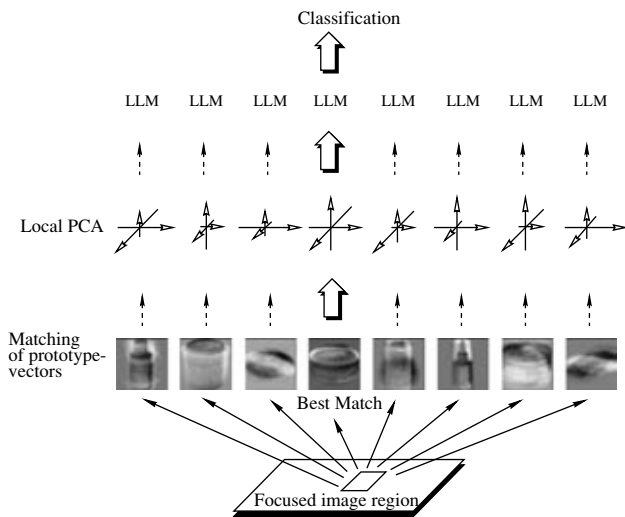


Fig. 3. VPL architecture for classification. In the first stage (V, bottom row), for the focused input window, the best match reference vector is sought. Reference vectors are visualized as prototypic images. This selection decides for the local PCA coordinate system to be used (P, middle row). The projection onto the local PCs serves as a feature vector, which is in the third stage (L, top row), classified by the corresponding LLM net.

successive calculation of the principal components after the rule of [7]. Training of the P stage means that, for all images that fall into the Voronoi tessellation cell of a certain reference vector, the principal components (PCs) are calculated. Thus, the V and P stages implement a simple version of LPCA. More elaborate versions of LPCA adapt locations of the reference vectors and PC computation iteratively, guided by the achieved reconstruction error [8]. However, for fast adaptation to a large dataset in a high dimension, separated training is much faster, with only a minor loss of quality in the achieved representation.

The third stage (“L”) is trained supervised from the labeled windows. To each of the principal component analysis (PCA) nets, a neural classifier of the local linear map (LLM) type is attached (see [9] for a detailed description). Each LLM is responsible only for the classification of the samples in the Voronoi cell of its reference vector.

To obtain the classification of an input window, first, the best match reference vector is sought; then, the window vector is projected to the local PCs to obtain a feature vector. The final LLM classifies the feature vector into one of the object classes.

EVALUATION

The key question is how well the VPL classifier can perform fast training. We distinguish two training modes: “Fast Training” (FT) leaves the V and P stages untouched; i.e., feature extraction remains the same. Only the LLM nets are readapted to the new dataset. In

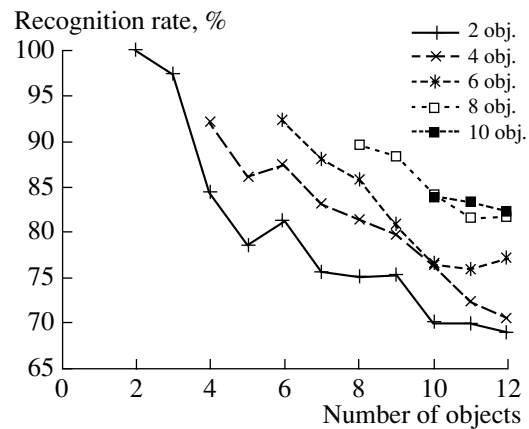


Fig. 4. Classification results for online learning (see text).

contrast, “Complete Training” (CT) trains the V, P, and L stages from scratch. FT can be carried out fast enough to make the system operational in less than a second after the acquisition of novel views, whereas CT computes a completely new VPL in the background.

Figure 4 shows the percentage of correctly classified objects for a VPL trained by CT for two to ten objects (the leftmost point of each curve). Then, more objects were “stuffed” into the VPL by mere FT. The graphs can be interpreted as follows. In all cases, recognition rates decrease when additional objects are acquired by FT. However, the decrease becomes smaller the more objects were initially trained by CT; i.e., a VPL trained to a large set of objects (CT) can handle an object added by FT much better than a VPL trained to only a small set of objects. This effect is caused by the LPCA representation (V and P stages): a representation of a large variety of views is much more likely to also capture a novel object than a highly specialized, narrow representation.

The example presented in Fig. 4 employs a VPL with only three reference vectors in the V stage, eight local PCs, and 20 nodes for the LLM networks. Classification performance improves smoothly with increasing dimensions of the networks until saturation is reached.

CONCLUSIONS

We have presented a system for interactive learning of objects that is based on a flexible neural architecture and provides a natural user interface. In the future, we will enhance the capability of object retrieval by integration of self-localization methods in combination with visual scene memory.

ACKNOWLEDGMENTS

This work was supported within the project VAMPIRE, which is part of the IST program (grant no. IST-2001-34401).

REFERENCES

1. H. Murase and S. K. Nayar, "Visual Learning and Recognition of 3D Objects from Appearance," *Int. J. Comput. Vis.* **14**, 5–24 (1995).
2. T. Kalinke and W. Von Seelen, "Entropie als Mass des lokalen Informationsgehalts in Bildern zur Realisierung einer Aufmerksamkeitssteuerung," *Mustererkennung 1996* (Springer Verlag, 1996), pp. 627–634.
3. D. Reissfeld, H. Wolfson, and Y. Yeshurun, "Context-free Attentional Operators: The Generalized Symmetry Transform," *Int. J. Comput. Vis.* **14**, 119–130 (1995).
4. C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Proc. 4th Alvey Vision Conf.* (1988), pp. 147–151.
5. R. Rae, M. Fislage, and H. Ritter, "Visuelle Aufmerksamkeitssteuerung zur Unterstützung gestikbasierter Mensch–Maschine Interaktion," *KI—Künstliche Intelligenz.*, No. 1, 18–24 (1999).
6. G. Heidemann and H. Ritter, "Efficient Vector Quantization Using the WTA-Rule with Activity Equalization," *Neural Process. Lett.* **13** (1), 17–30 (2001).
7. T. D. Sanger, "Optimal Unsupervised Learning in a Single-Layer Feed-Forward Neural Network," *Neural Networks* **2**, 459–473 (1989).
8. N. Kambhatla and T. K. Leen, "Dimension Reduction by Local Principal Component Analysis," *Neural Comput.* **9** (7), 1493–1516 (1997).
9. H. J. Ritter, T. M. Martinetz, and K. J. Schulten, *Neuronale Netze* (Addison-Wesley, 1992).