Embodied Object Recognition using Adaptive Target Observations

Marcus Wallenberg and Per-Erik Forssén

Cognitive Computation Journal manuscript No.
(will be inserted by the editor)

Embodied Object Recognition using Adaptive Target Observations

Marcus Wallenberg · Per-Erik Forssén

Received: date / Accepted: date

Abstract In this paper, we study object recognition in the embodied setting. More specifically, we study the problem of whether the recognition system will benefit from acquiring another observation of the object under study, or whether it is time to give up, and report the observed object as unknown.

We describe the hardware and software of a system that implements recognition and object permanence as two nested perception-action cycles. We have collected three data sets of observation sequences that allow us to perform controlled evaluation of the system behaviour. Our recognition system uses a KNN classifier with bag-of-features prototypes. For this classifier, we have designed and compared three different uncertainty measures for target observation. These measures allow the system to (a) decide whether to continue to observe an object or to move on, and to (b) decide whether the observed object is previously seen or novel. The system is able to successfully reject all novel objects as "unknown", while still recognising most of the previously seen objects.

Keywords Object recognition · Attention · Visual search · Fixation · Object permanence

1 Introduction

Visual object recognition is a useful skill for interactive robots, for direct references; e.g. "Robot, bring me my coffee mug"; and by means of way-points, e.g. "Bring me an apple from the bowl next to the fridge".

Artificial systems often treat recognition as a one shot phenomenon, where methods from image database retrieval [12,7,6,18,10] are fed with the image flow from a bottom-up attention system. This allows a robot to sequentially attend to visually salient targets, and report whether they are recognised or not [23,3]. This is done in [5] using a binary boosted classifier for each known object. In [14] a foveated region is combined with neighbouring regions to form a geometric constellation model. In both systems, object detection is based on a single fixation.

In contrast to this, it is well known that e.g. infants look longer at unpredicted and surprising visual input. This fact is widely used to draw conclusions about infant perception [8]. This motivates us to pose recognition as an active, ongoing process involving *adaptive decision-making* that can be tuned by *feedback learning*. In comparison with recognition in databases, the *embodied recognition* setting brings many advantages, where the most basic one is the option to postpone a decision in uncertain cases.

We have developed a hardware and software platform that exhibits adaptive fixation duration during recognition, see figure 1. Our system is implemented as two nested perception-action (PA) cycles: One for *target observation* and one for *object permanence* (attention and maintenance of a world model). The action component of the PA cycles are currently fairly limited. In the target observation cycle, the action consists of small changes in camera orientation followed by collection of another observation. In the object permanence cycle the actions are switches to novel targets. Future developments will include a cycle for verbal interaction with a user to request different views of an object. Cy-

M. Wallenberg
Linköping University
Tel.: +46 (13) 281329
E-mail: wallenberg@isy.liu.se

P-E. Forssén
Linköping University

cles for sideways motion, and for interaction with the objects are also possible extensions, but these require additional hardware.

We have collected a large database of sequential observations, and use it for controlled evaluation of both the target observation cycle, and of measures for deciding whether the target is recognised or novel.

1.1 Contributions

This paper studies the decision problem of whether to continue to observe a target or to move on. To the best of our knowledge, this is the first solution to this problem that has been applied to a physical robot system.

- We describe the components of an embodied recognition system that treats recognition as an on-line sequential process.
- We present and evaluate three uncertainty measures for object identity.
- We describe how to use these measures to implement an *adaptive target observation*. That is, to decide whether to continue to observe an object or to move on.

The uncertainty measure also allows us to decide whether the observed object is previously seen, or unknown — a prerequisite for learning of new objects on the fly.

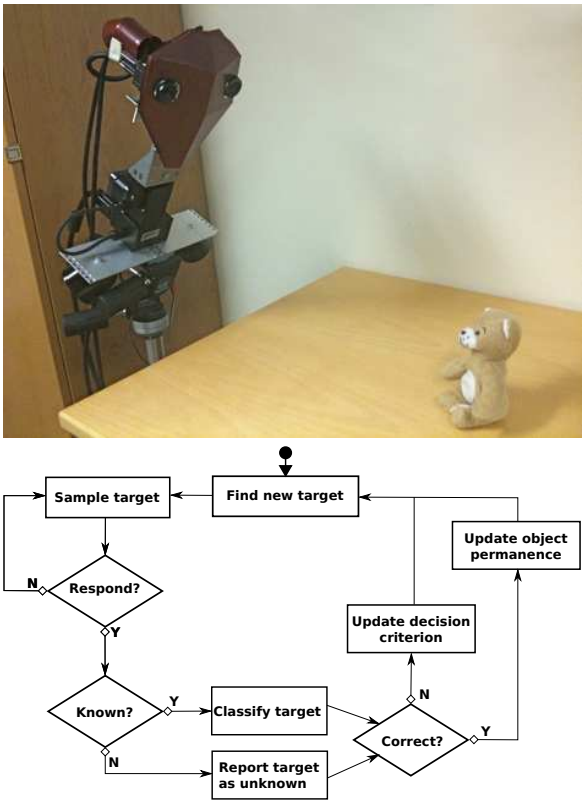

Fig. 1 Top: Robot platform. **Bottom:** system behaviour. Cycles control target observation (top left) and object permanence (main cycle).

1.2 System Overview

Our system consists of hardware for image acquisition and movement, as well as software for adaptive recognition, object permanence, and user communication. The robot platform has a dual camera mount atop a rigid aluminium frame. The frame is in turn mounted on a fast pan-tilt unit (see figure 1). The robot also has an on-board speaker system for communication, and is controlled by a desktop computer via FireWire and USB.

The platform is designed to test ideas inspired by the human visual system.

1.3 Visual Search

Examining the entire visual field in high resolution is in most cases intractable in both artificial and biological systems. Therefore, a visual attention mechanism is usually applied to a low-resolution version of the visual field and used to guide a high-resolution "spotlight".

Human visual search is performed using a *peripheral-foveal system*, using low-acuity peripheral vision for guidance and high-acuity foveal vision for recognition. Foveal vision is directed by fast eye movements known as *saccades*, which orient the eye toward regions of interest in the visual field [15].

We implement visual search with a static, bottom-up saliency measure called *incremental coding length* (ICL) [9] computed on the system's peripheral view. It is combined with an inhibition mechanism that suppresses visual saliency in regions containing previous object observations. Details of the attention system can be found in [24]. The visual search behaviour is illustrated in figure 2.

1.4 Object Permanence and Inhibition of Return

Object permanence is a skill that allows the system to remember the locations of previously seen objects, even when they are not currently in view. In humans, object permanence is typically learnt before the age of two.

We implement object permanence by storing previous object sightings as (non-metric) 3D point clouds of matched feature locations and their associated class labels. Object permanence allows us to implement an object based inhibition of return (IOR) mechanism, simi-

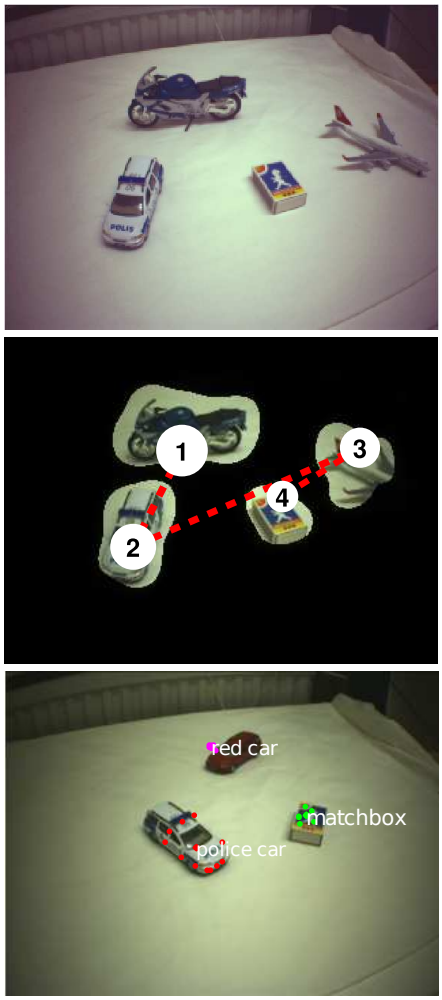

Fig. 2 Top: Peripheral view. **Centre:** resulting (thresholded) saliency map and resulting visual search pattern. **Bottom:** object reprojection scheme.

lar to the one in [14]. We implement IOR by projecting the convex hull of the stored point cloud to the image plane and suppressing the corresponding region in the saliency map. Performing IOR on the object level, rather than using a purely visual-field-based representation, allows us to also suppress previously attended objects that have temporarily been out of view. The object reprojection scheme is visualised in figure 2.

1.5 Visual Feature Extraction

Pattern matching in the human brain is very rapid. Foveate recognition is mainly feed-forward and happens in less than 150ms [21]. The *standard model* [16] of the processing hierarchy starts with only a few feature types (oriented bars and edges) and a high spatial specificity. For each layer the number of feature types increases, and the spatial specificity is reduced. At the top level we have a very high number of feature types, and no spatial information. In this sense, the top level is analogous to *bag-of-features* (BoF) matching [19], which we use here.

Bags-of-features are constructed from local invariant features such as SIFT [11] and MSER [13]. For each feature, a local image patch is converted into a *descriptor vector* that is robust to illumination changes and geometric perturbations. The descriptors are then quantised into visual words, and the entire image is represented as a *bag of features*: a histogram of visual word occurrences. The choice of features is in itself not crucial, as BoF-like methods have been used with a variety of feature combinations [18,19,22].

The term *visual word* stems from the analogy to document analysis [19]. A vocabulary of visual words is defined by first computing descriptor vectors for all features in a representative set of images. These are then clustered. In this paper, we use a fixed vocabulary of $k = 8\,000$ visual words, computed from approximately 10^5 SIFT descriptor vectors by k-means clustering [2]. The decision to use SIFT features is motivated by their popularity in image retrieval systems, and the vocabulary size is chosen for performance reasons.

2 Prototype Construction and Classification

2.1 Prototype Construction

The aim of prototype construction is to create a set of templates $\{\boldsymbol{p}_j\}$, with associated class labels $\{c_j\}$, where $c_j \in \mathcal{C}$, and $\mathcal{C} = \{C_1, \ldots, C_l\}$. These are created from a set of N labelled training samples $\{\boldsymbol{t}_n, c_n\}$, where $\boldsymbol{t_n}$ are visual word histograms of training images and $c_n \in \mathcal{C}$ are the corresponding class labels. This prototype set should be robust to errors in feature detection, yet compact enough to be handled efficiently. In this section, we describe four methods of creating these prototypes, which we later evaluate.

The first two methods use only the labels of training data to create the prototypes, while the remaining use clustering techniques.

A weighting scheme based on the *inverse document frequency* (IDF) [17] is used. Each bin k in the visual word histogram is assigned a weight w_k calculated from the training samples $\{\boldsymbol{t}_n\}$, as

$$w_k = -\ln\left(\frac{N}{|\{n : t_{nk} > 0, n \in [1, N]\}|}\right). \quad (1)$$

When the prototype vectors have been calculated (using one of the methods described below) a matching

matrix $\boldsymbol{P}$ is calculated as

$$\boldsymbol{P} = [\hat{\boldsymbol{p}}_{\boldsymbol{w}1}, \ldots, \hat{\boldsymbol{p}}_{\boldsymbol{w}j}], \quad \text{where} \quad \hat{\boldsymbol{p}}_{\boldsymbol{w}j} = \frac{\boldsymbol{W}\boldsymbol{p}_j}{||\boldsymbol{W}\boldsymbol{p}_j||} \tag{2}$$

and $\boldsymbol{W}$ is a diagonal matrix containing the weights w_k.

2.1.1 Class mean and single-sample prototypes.

The simplest prototype construction option is to simply sum all training samples $\boldsymbol{t}_n$ with class label C_l into one prototype vector $\boldsymbol{p}_j$ and then associate this vector with the corresponding class label C_l by setting $c_j = C_l$.

The opposite extreme of prototype construction (here called *single-sample* prototypes) regards each training sample $\boldsymbol{t}_n$ as a unique instance, and assigns one $\boldsymbol{p}_k$ to each training sample.

2.1.2 k-nearest-neighbour-clustered prototypes.

To retain prototype specificity, while keeping the number of prototypes low it is useful to determine which samples in the training data can be merged into a single prototype. One method is to perform within-class clustering among the training samples $\boldsymbol{t}_n$. We use the Tanimoto coefficient [20], see (5) to calculate an $N \times N$ adjacency matrix $\boldsymbol{A}$ as

$$\boldsymbol{A} = [\hat{\boldsymbol{t}}_{\boldsymbol{w}1}, \ldots, \hat{\boldsymbol{t}}_{\boldsymbol{w}N}]^T [\hat{\boldsymbol{t}}_{\boldsymbol{w}1}, \ldots, \hat{\boldsymbol{t}}_{\boldsymbol{w}N}], \tag{3}$$

$$\text{where} \quad \hat{\boldsymbol{t}}_{\boldsymbol{w}k} = \frac{\boldsymbol{W}\boldsymbol{t}_k}{||\boldsymbol{W}\boldsymbol{t}_k||} \quad \text{for} \quad k = 1, \ldots, N. \tag{4}$$

When the adjacency matrix has been calculated, each training sample is symbolically linked to its k nearest in-class neighbours, and all resulting unique clusters are averaged into prototype vectors $\boldsymbol{p}_j$. This results in a set of prototypes with fewer elements than the training set, where each prototype in computed from the average of a subset of the training data.

2.1.3 k-nearest-neighbour-averaged prototypes.

Results from image retrieval using BoF methods indicate that the matching process in some cases benefits from averaging of adjacent training samples [22]. We perform this using adjacency calculated as in (3). Each training sample $\boldsymbol{t}_n$ is then averaged with its k nearest in-class neighbours to form a prototype vector $\boldsymbol{p}_j$. This results in a set of prototypes with the same number of elements as the training set, where each prototype is computed from the average of a training sample and adjacent neighbours with the same class label.

2.2 Classification of Novel Images

All classification methods we use can be described as various cases of *k-nearest neighbours* (KNN) classification [2]. Regardless of how the matching matrix $\boldsymbol{P}$ (2) is constructed, our classification procedure remains the same.

The similarity measure used is the Tanimoto coefficient [20]. The similarity, s_j, of query vector $\boldsymbol{q}$ and prototype vector $\boldsymbol{p}_j$ is calculated as

$$s_j = \frac{\boldsymbol{q}^T \boldsymbol{W}^2 \boldsymbol{p}_j}{||\boldsymbol{W}\boldsymbol{q}|| \cdot ||\boldsymbol{W}\boldsymbol{p}_j||}. \tag{5}$$

However, since we already have the pre-weighted and normalised matching matrix $\boldsymbol{P}$, we can apply the same operations to the query vector and then calculate all similarities as

$$\boldsymbol{s} = \hat{\boldsymbol{q}}_w^T \boldsymbol{P}, \text{ where } \hat{\boldsymbol{q}}_{\boldsymbol{w}} = \frac{\boldsymbol{W}\boldsymbol{q}}{||\boldsymbol{W}\boldsymbol{q}||}. \tag{6}$$

Once all similarities have been calculated, they are sorted in descending order, and KNN voting (with $k = J$) is carried out according to

$$v_l = |\{j : c_j = C_l, j \in [1, J]\}| \ . \tag{7}$$

Here, v_l denotes the number of votes received by class l and $\boldsymbol{v} = [v_1, \ldots, v_l]$.

For all prototype construction methods except the single-sample prototypes, we use $J = 1$ (corresponding to *nearest neighbour* (NN) classification). For class mean prototypes, this is necessary since every class is represented by a single prototype vector. In the clustered methods we wish to study performance as k increases, and therefore set $J = 1$. For single-sample prototypes, we use $J = 1, 2, \ldots, 50$.

3 Uncertainty Measures for Target Observation

The purpose of a target uncertainty measure is twofold. Firstly, the system must determine whether it is done examining a target. Secondly, the system must decide whether the target is known or unknown. The actual difference in appearance needed to distinguish a known object from an unknown one cannot be defined in a straight-forward manner, since it is dependent on (a) the specific content of the training set (the appearance of previous objects), (b) the specificity of the visual vocabulary and (c) how different the appearance of the object is during learning and a subsequent observation. It is therefore a property which must be tuned to the specific conditions under which observation is performed.

Once enough observations of a target have been gathered and it is considered known, it will be classified by the system (see figure 1). Our aim is to use the similarities from (6), and the resulting class ranking to assign such an uncertainty to the target identity.

The uncertainty measures we propose aim to describe the risk of misclassification, by comparing the two classes ranked first and second in a classification. If, for instance, a sample has two equal contenders among the class labels, this should result in complete uncertainty as there is no way to say which one is correct. In such cases, a decision must be postponed, until more powerful methods can be applied. The uncertainty measure should also produce predictable values for unknown objects as well as confining these values to an easily identified range for reliable identification of these occurrences. We evaluate three uncertainty measures based on different utilisations of similarity and class ranking. These are described below.

3.1 Uncertainty Measures

Using the similarities from (6) and the resulting class ranking (7), we define three uncertainty measures,

$$h_v = \frac{v_B}{v_A}, \tag{8}$$

$$h_s = \frac{s_B}{s_A} \quad \text{and} \tag{9}$$

$$h_d = \frac{1 - \sqrt{s_B}}{1 - \sqrt{s_A}} \tag{10}$$

Here, s_A and s_B denote the average of the K highest similarities in the two classes ranked first and second respectively (a typical value is $K = 7$). In the same manner v_A and v_B denote the number of votes received by these classes. Thus, the first measure, h_v, is the quotient of the votes received by these classes (note that this is only applicable when using KNN). The second measure, h_s, is the quotient of the similarities of the two strongest competitors in each of these classes. The final measure, h_d, can be seen as the quotient of the dissimilarities of these two classes to the query vector. The purpose of the square-root in h_d is to expand the range of values corresponding to correct classifications. We evaluate decisions based on thresholding of these three uncertainty measures.

4 Results

4.1 Training and Evaluation Data

Training and evaluation data was captured for nine object classes. Examples from these classes, and their associated class labels are shown in figure 3.

Training data was obtained by placing an object on a white sheet in front of the robot. For each object pose, the robot was allowed to examine the object and collect five *observations* (high-resolution foveal view pairs), each with a slight offset in gaze direction. In the data sets, a collection of observations of the same pose constitute a *view*. The object pose was then changed and the procedure repeated until 40 such views had been gathered. This was done for each of the nine training objects, resulting in 200 observations per object.

In order to obtain predictable and reproducible results, the same training set was used in all classification trials. An "easy" evaluation set was gathered in the same way as the training data, under similar conditions (similar object distance, background and illumination). A "hard" evaluation set was also gathered. In this set, background, illumination and distance to the object were varied using a desktop turntable to produce more challenging data, where background and distance to the object varied. The illumination was also changed several times during data collection between sunlight, incandescent and fluorescent lighting. Examples of images from these sets are shown in figure 3. All data sets contain 40 views of each object, each containing five observations (resulting in 3600 images per data set).

4.2 Prototype Construction and Classification

In order to select a prototype construction and classification method, the methods from sections 2.1 and 2.2 were evaluated on the "easy" and "hard" data sets. The *correct classification rate* (CCR) obtained using each observation separately is shown in figure 4. Performance when using all five available observations in each view was generally higher, but similarly distributed. Since the combination of single-sample prototypes and KNN classification provides a consistently high CCR, it is the method used in the following experiments.

4.3 Sequential Recognition and Uncertainty Measures

The sequential addition of several observations within a single view (object pose) is motivated by the fact that

Fig. 3 **Top:** examples of the nine training objects. **Bottom:** examples of training and evaluation images for object #1.

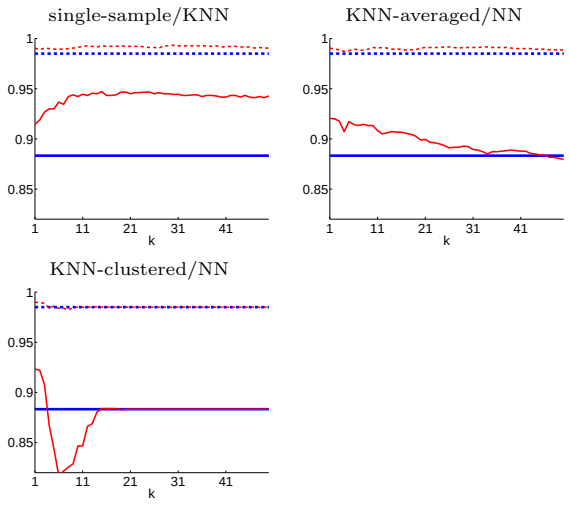

Fig. 4 Single-observation CCR on both evaluation sets. Dashed lines show results on "easy" set, solid lines show results on "hard" set. Results obtained using class mean prototypes (constant across k, blue) are shown for reference.

feature extraction is not completely reliable due to subtle variations in illumination and image noise across the sequence of observations. The detection of a prototypical visual word distribution is thus aided by the addition of more observations even when captured from the same camera position and orientation. Note also that noisy features are automatically suppressed by using multiple observations, since they are unlikely to occur in several consecutive observations. Our experiments also seem to indicate that small changes in camera orientation increase the likelihood of convergence of known objects to a prototypical visual word distribution by increasing the chance that features seen in training are detected.

4.3.1 Uncertainty distribution.

In order to estimate the distribution of uncertainty for "accepted and correctly classified" and "rejected and misclassified" samples, we calculated the three uncertainty measures and the resulting decisions using a sequential evaluation procedure (described further below). Normalised kernel density estimates of the resulting uncertainty distributions are shown in figure 5, left column. As the figure shows, h_d provides the strongest concentration of uncertainty for "rejected and misclassified" samples, which suggests it is the most suitable for easy and reliable rejection of unknown objects.

4.3.2 Convergence to decision.

We illustrate the convergence behaviour of the uncertainty measures using two sequences of 100 observations each. One sequence contained foveal views of a known object and the other contained foveal views of an unknown object. We studied the behaviour of each of the uncertainty measures when adding more observations to the bag of features.

The system was set to signal it was ready to decide if a target was known or unknown when the change in uncertainty was less than 10^{-3} when averaged over over the last ten views. The results are shown in figure 5. As the figure shows, the uncertainty measure h_d exhibits faster convergence for the unknown object. This suggests that using h_d as uncertainty measure also leads to faster decisions. The rate of change in uncertainty can be interpreted as a measure of "hesitation", corresponding to an estimate of how much information can be expected to be gained from acquiring an additional observation of an object. In the embodied setting, when a potentially unlimited number of views are available, this is the way in which we want to set fixation duration. However, when studying the effects of different

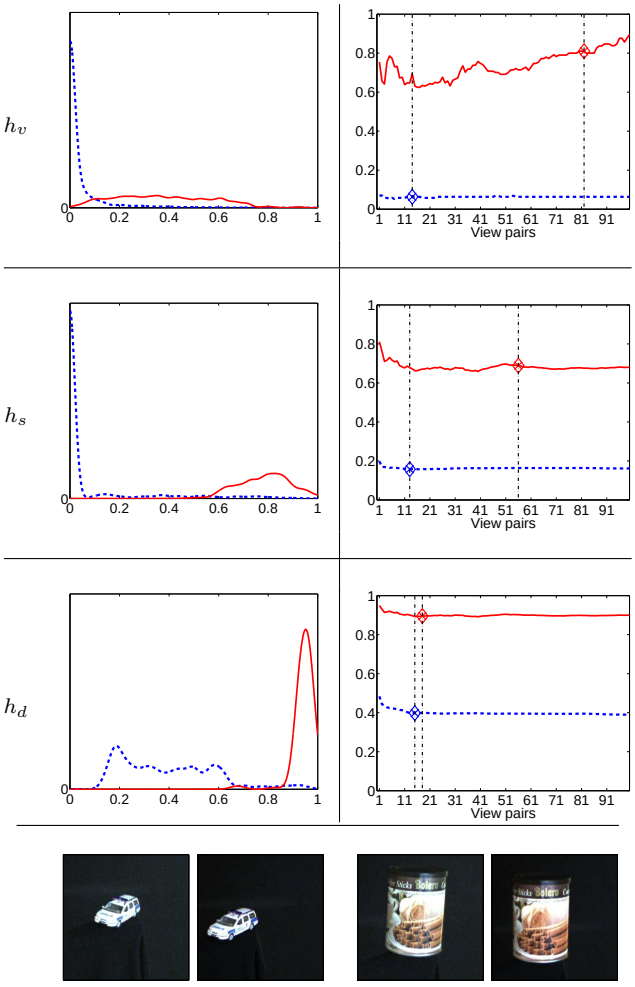

Fig. 5 Left column: normalised kernel density estimate of uncertainty measures (with $\sigma = 2 \cdot 10^{-2}$) over all sets. Dashed lines show accepted and correctly classified samples, solid lines show rejected and misclassified samples.
Right column: uncertainty convergence over 100 observations of a known (dashed blue) and an unknown (solid red) object. Diamonds show the frame where the decision was made.
Bottom: Left: an observation of the known object. Right: an observation of the unknown object.

thresholds on the evaluation data sets, having only five observations of a specific object pose available, we instead resort to terminating the fixation when uncertainty falls below a threshold (as described below).

4.3.3 Decision quality.

In the embodied setting it is important that the system not incorrectly associates a novel object with a previously seen one, as this would contaminate the prototype set. In order to allow autonomous learning of novel objects we thus cannot tolerate false recognitions.

The quality of the known/unknown decision mechanism depends on two factors:

1. We desire the decision process to reject potentially unknown objects, rather than attempt to assign an identity which may be incorrect.
2. At the same time we also desire the decision process to accept as many of the correctly classified samples as possible.

This trade-off can be analysed using the *precision* and *recall* measures [4]. Note however that we do not use precision and recall in the normal sense, where a classifier performance is evaluated by moving the decision threshold. Here we are instead evaluating a complete *decision-making behaviour*. A change in the threshold may now cause the system to observe more or fewer images. A good threshold setting would be just before the precision drops below one, as this would give us the highest possible recall with zero false recognitions.

To see how well the decision process can reject unknown objects, four out of the nine training objects were removed from the training set before evaluation of uncertainty measures. As ground truth, we labelled a decision as correct if the classification was successful and as incorrect otherwise.

Precision, recall and the resulting *mean frames until decision* (MFD) were calculated in a sequential evaluation process as follows:

1. Acquire a frame pair and run the classifier.
2. Calculate the uncertainty.
3. If uncertainty $<$ threshold, accept classification.
4. If uncertainty $\geq$ threshold, acquire new frame, add features and repeat.
5. If no more frames are available, stop and reject classification.

This was carried out using each of the three uncertainty measures. This was done for the "Easy" and "Hard" data sets, as well as for the training set using *leave-one-out cross-validation* (LOOCV) [1]. The thresholds were initially set to zero (complete rejection) and gradually relaxed toward complete acceptance.

The results of this evaluation are shown in figure 6. For h_s and h_d we have used averages of $K = 7$ similarities in these experiments. As can be seen in the figure, h_v fails to achieve full precision for any threshold value (maximum precision in LOOCV is 0.966 for a threshold value of 0.01). The lowest possible non-zero value of h_v is determined by the number of neighbours J used in classification as $h_{v,min} = \frac{1}{J-1}$, which requires that $J > 100$ to obtain values smaller than 0.01. This leads to an undesirable decrease in classification performance and we therefore typically use $J < 100$. This means that basing decisions on h_v equates to requiring $h_v = 0$ and also invariably leads to errors, since not even $h_v = 0$

LOOCV		"Easy"	"Hard"
h_v	Precision	*0.980*	*0.973*
	Recall	*0.970*	*0.725*
	MFD	*2.9*	*3.5*
h_s	Precision	1.000	**1.000**
	Recall	0.940	0.413
	MFD	2.9	4.2
h_d	Precision	1.000	0.978
	Recall	**1.000**	**0.684**
	MFD	**2.8**	**3.6**

Table 1 Precision, recall, and *mean frames until decision* (MFD) using a decision threshold determined from LOOCV. Italics show best possible results obtained using h_v, which never reaches full precision.

OPTIMAL		"Easy"	"Hard"
h_s	Recall	0.940	0.592
	MFD	2.9	3.8
h_d	Recall	**1.000**	**0.653**
	MFD	**2.8**	**3.7**

Table 2 Recall and MFD for optimal threshold setting. (Highest possible recall with precision 1).

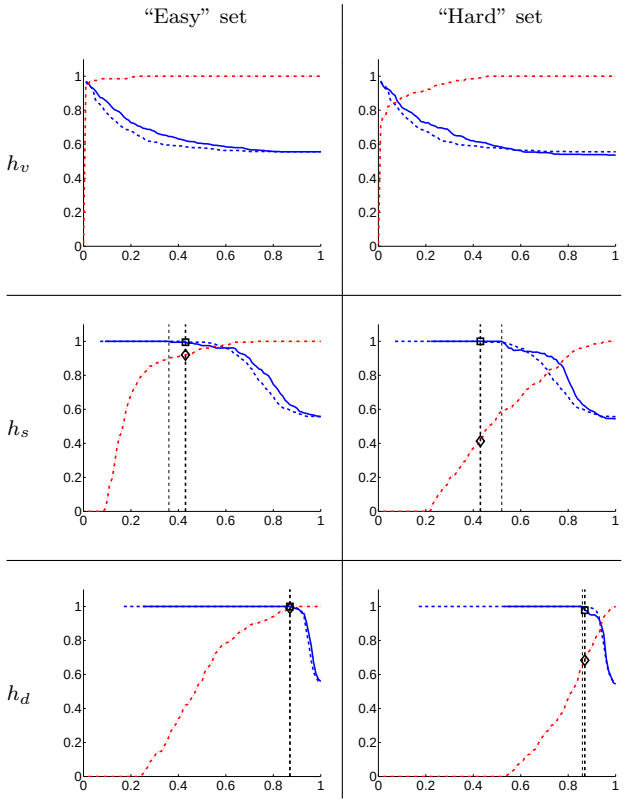

Fig. 6 Left column: Recall and precision using the three uncertainty measures on the "Easy" data set.
Right column: Recall and precision using the three uncertainty measures on the "Hard" data set.
Recall curves (dash-dotted red) start at zero and are increasing. Precision curves (blue) start near one and approach the ratio of known to unknown samples in evaluation.
For precision, we show two curves: Dashed blue curves show precision obtained in LOOCV on training set, solid blue lines show precision in evaluation. Thick vertical dashed lines show thresholds that give maximum recall while maintaining full precision in LOOCV. Square and diamond markings show actual precision and recall when using these thresholds (shown in table 1). Thin vertical lines show optimal threshold values determined ex post (shown in table 2).

results in full precision. A threshold therefore cannot be set in the desired fashion described above.

In contrast, both h_s and h_d attain full precision at significant recall, and which allows a useful threshold to be selected. The uncertainty measure h_d has the most consistent behaviour on the different data sets, which suggests that an adaptive threshold initialised using training data could generalise rather well to evaluation data of varying difficulty. As can be seen in tables 1 and 2, h_d exhibits both higher recall and shorter decision times than h_s, and also achieves full recall on the "Easy" set while maintaining full precision (which h_s does not). Note, however, that the threshold from LOOCV using h_d on the "Hard" data set gives rise to a small number of falsely accepted incorrect classifications. If we look at the lower right plot in figure 6, we see that an adjustment of the threshold by a mere 0.01 would have solved this problem. A slightly more restrictive strategy for selecting the threshold would thus be recommended. E.g. one could chose the threshold to be some safe margin below the last full precision point.

5 Conclusions and Future Work

We have presented and evaluated a strategy for active and sequential visual object recognition in the embodied setting. Our strategy uses bag-of-features-based image matching to adaptively decide on when to classify an object, and whether to trust that classification or not. As this allows both classification and rejection of an object as unknown, it is a more realistic model of recognition than direct classification. Our decision criteria result in more frames until decision when faced with difficult or unknown object views. Thus, they mirror the increased hesitation displayed by the human visual system when faced with unpredicted and surprising visual input. The increased decision time for unknown objects (see figure 5) indicates that the uncertainty measure used makes a premature and erroneous decision less likely than waiting and responding with greater certainty. It also seems that the rejection criterion learnt from LOOCV generalises well to novel data and can be used for initialisation of an adaptive threshold.

Future work includes replacing the decision threshold with adaptive decision-making. We will also investigate more sophisticated view-planning strategies, and the incorporation of new kinds of image features and object geometry into the recognition strategy. Another possible development is an object observation strategy that incorporates interaction with the user. This would allow the system to ask the user to show more views of an object before deciding on its identity. Such a strategy could make use of dynamic visual attention and object tracking to incorporate multiple views of an object, without the system itself being able to manipulate it.

Acknowledgement

This work was supported by Linköping University, and the Swedish Research Council through a grant for the project *Embodied Visual Object Recognition*.

References

1. C.G. Atkeson. Using locally weighted regression for robot learning. In *ICRA*, pages 958–963, Sacramento, CA, 1991.
2. C. M. Bishop. *Neural Networks for Pattern Recognition.* OU Press, 1995.
3. Mårten Björkman and Jan-Olof Eklundh. Vision in the real world: Finding, attending and recognizing objects. *International Journal of Imaging Systems and Technology*, 5(16):189–209, 2006.
4. Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In *ICML06*, pages 233–240, 2006.
5. Stephen Gould et al. Peripheral-foveal vision for real-time object recognition and tracking in video. In *IJCAI*, 2007.
6. Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE TPAMI*, 28(4):594–611, 2006.
7. R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from Google's image search. In *ICCV*, volume 2, pages 1816–1823, 2005.
8. Marshall Haith. Who put the cog in infant cognition? Is rich interpretation too costly? *Infant Behaviour & Development*, 21(2):167–179, 1998.
9. Xiaodi Hou and Liquing Zhang. Dynamic visual attention: Searching for coding length increments. In *NIPS*, 2008.
10. Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, volume I of *LNCS*, pages 304–317, 2008.
11. David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.
12. David Nistér and Henrik Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, volume 2, pages 2161–2168, 2006.
13. Stepán Obdrzálek and Jirí Matas. Object recognition using local affine frames on distinguished regions. In *BMVC*, pages 113–122, 2002.
14. Francesco Orabona, Giorgio Metta, and Giulio Sandini. Object-based visual attention: a model for a behaving robot. In *CVPR05 Workshop APCV*, 2005.
15. Stephen E. Palmer. *Vision Science, Photons to Phenomenology.* MIT Press, 1999.
16. Maximilian Riesehuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025, 1999.
17. Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513–523, 1988.
18. S. Savarese and Li Fei-Fei. 3D generic object categorization, localization and pose estimation. In *ICCV*, 2007.
19. Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003.
20. Taffee T. Tanimoto. IBM internal report. 1957.
21. Simon Thorpe, Denis Fize, and Catherine Marlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.
22. Panu Turcot and David G. Lowe. Better matching with fewer features. In *ICCV Workshop (WS-LAVD)*, 2009.
23. Ales Ude, Chris Gaskett, and Gordon Cheng. Support vector machines and gabor kernels for object recognition on a humanoid with active foveated vision. In *IEEE Conference on Intelligent Robots and Systems (IROS'04)*, 2004.
24. Marcus Wallenberg and Per-Erik Forssén. A research platform for embodied visual object recognition. Technical report, SSBA'10 Symposium on Image Analysis, 2010.