Learning Responses to Visual Stimuli: A Generic Approach

Liam Ellis and Richard Bowden [*]

CVSSP, University of Surrey, Guildford, Surrey. {L.Ellis R.Bowden}@Surrey.ac.uk

Abstract. A general framework for learning to respond appropriately to visual stimulus is presented. By hierarchically clustering percept-action exemplars in the action space, contextually important features and relationships in the perceptual input space are identified and associated with response models of varying generality. Searching the hierarchy for a set of best matching percept models yields a set of action models with likelihoods. By posing the problem as one of cost surface optimisation in a probabilistic framework, a particle filter inspired forward exploration algorithm is employed to select actions from multiple hypotheses that move the system toward a goal state and to escape from local minima. The system is quantitatively and qualitatively evaluated in both a simulated shape sorter puzzle and a real-world autonomous navigation domain.

1 Introduction

The purpose of this work is to develop a system that can learn to respond appropriately to its environment in multiple problem domains. *Cognitivist* approaches, that rely on hard-coded knowledge or engineered rule based systems have shown limited success, especially when the systems are expected to perform in multiple domains or in domains that stray too far from the idealised world envisaged by the engineer. For this reason *emergent* architectures, that model the world through co-determination with their environments, have become a popular paradigm [8] [13]. As shall be made clear, organisation via the response domain is key to the system presented here. Granlund has postulated that related points in the response domain exhibit much greater continuity, simplicity and closeness than related points in the input domain and therefore that the organisation process has to be driven by the response domain signals[9].

Recent neurophysiological research has shown strong support for the existence of a mechanism that obtains a percept-action coupling, known in literature as *direct-matching hypothesis*. The mirror-neuron system provides biological systems with the ability "to recognise actions performed by others by mapping

[*] Part of the work presented here was supported by the the European Union, grant COSPAL (IST-2004-71567) However, this paper does not necessarily represent the opinion of the European Community, and the European Community is not responsible for any use which may be made of its contents. This work is also partly funded by EPSRC through grant EP/P500524/1 to the University of Surrey.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

the observed action on his/her own motor representation of the observed action" [2]. Related to this is the work of Siskind, where an attempt is made to analyse from visual data, the force dynamics of a sequence and hence deduce the action performed[12]. Fitzpatrick et al. have shown that it is possible for an agent to learn to mimic a human supervisor by first observing simple tasks and then, through experimentation, learning to perform the actions that make up those tasks[5]. These approaches deal with exact mimicry, the system presented here aims to extend these capabilities to more general imitation.

Much of the work dedicated to developing cognitive vision systems follows the connectionist methodology[10, 4]. Alternatively, Magee et al.[11] have presented a framework for the learning of object, event and protocol models from audio visual data employing both statistical learning (for objects), and symbolic learning (for sequences of events representing implicit temporal protocols). Despite sharing similar objectives, there is very little similarity between the approach taken by Magee et al. (Inductive Logic Programming), and that which we have taken (probabilistic exploration of hypotheses generated from approximate percept-action mappings).

The system is demonstrated in two problem domains. The first is a simulated shape sorter puzzle used extensively within the COSPAL project [3]. The simulator allows manipulation of the puzzle blocks in horizontal, vertical translation and rotation and provides a representation of the state of the puzzle. The second is an autonomous navigation domain in which a remote controlled car has been fitted with a wireless camera and learns to drive toward a target object.

The first main contribution of this work is a novel hierarchical representation of an exemplar set of point-to-point mappings from the percept to action domains that, akin to classical feature selection, is capable of weighting the importance of perceptual features with respect to the desired actions i.e. identifies which input features should elicit which responses. The second main contribution of this work is a probabilistic action space hypotheses exploration and selection algorithm that utilises a learnt cost function to guide the system toward its learnt goal. By avoiding hard coding to a specific problem domain, this framework is flexible and extendable and thus provides a generic solution to problem solving.

2 Approach

It is assumed that if a system has, stored in its memory, an exemplar mapping from all the possible perceptual stimuli (inputs) that it shall ever encounter and each is coupled with an appropriate action model (response), then it should be capable of responding as required to all visual stimuli. The main practical restriction to this scenario is the limit on the number of exemplars that can realistically be obtained and stored. In most problem domains the complete one-to-one mapping from percept space to action space would require an enormous number of exemplars and in many cases would be impossible as the complete nature of the perceptual input can not be known a priori.

In order to compensate for incomplete coverage of the percept space, and allow efficient searching of the exemplar store, a hierarchy of clustered exemplars

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

is used to generalise and organise the systems memories/experiences. Each leaf node in the hierarchy represents a single exemplar. More general models of the percept and action representations are formed as the exemplars are clustered together until, at the root node, all the exemplars are represented in a single model that captures the mean and covariance of each parameter for the entire set. Of key importance here is that the clustering is performed in the response domain as opposed to the input domain.

Searching the hierarchy for matches to the current perceptual input yields a set of action models of varying generality each associated with a likelihood (percept match likelihood). For 'familiar' inputs (inputs that closely match in the lower parts of the tree) specific action models are obtained from the hierarchy. In less familiar situations, matching generalised percepts produces general action models resulting in more random, exploratory behaviour. New experiences are added to the hierarchy to refine behaviour in unfamiliar scenarios, this is done offline and requires that the hierarchy be rebuilt. In order to obtain the optimal action from a set of action models obtained from the hierarchy, a forward looking multiple hypotheses exploration algorithm has been developed. This particle filter inspired algorithm utilises both the likelihood values and a cost function. The cost function is a measure of distance between the expected resultant state and the learnt goal state. See Fig. 1 for a system overview.

To bootstrap the system with an initial set of exemplars a teacher first demonstrates the problem by performing the appropriate actions through a user interface to the systems response mechanism (e.g. manipulator or car control). Each action is recorded by the system along with the systems current perceptual state and this forms an exemplar.

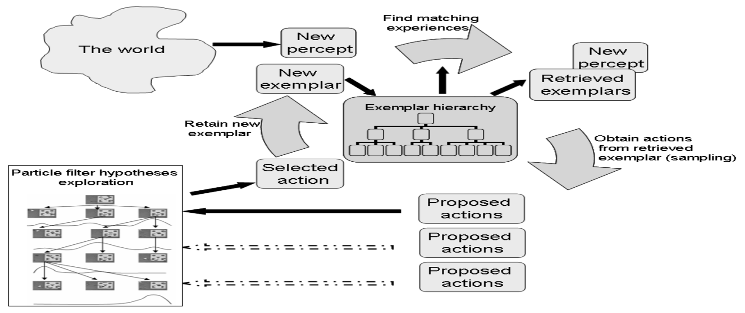


Fig. 1. System overview. The hierarchy provides solutions to the current problem that approximately imitate solutions to previous similar problems. Solutions are probabilistically evaluated and the selected solution is used and recorded by the hierarchy as a new exemplar.

2.1 Representing Actions and Percepts

For the shape sorter puzzle domain the feature set is directly provided by the simulator in the form of object id's and positions (horizontal - x, vertical - y

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

and rotational - θ). For the autonomous navigation domain blob features (or homogeneity features) are computed directly from the image. The blob feature extraction algorithm used, detailed in [6], provides a collection of blobs for each image where each blob, representing a homogeneous region in the image, is parametrised by six moments (describing shape) and the RGB encoding of the mean colour of the detected region. These parameters are concatenated to form a single nine element feature vector $[M_{00},M_{01},...,R,G,B]$ (see Fig. 2). Percepts are then represented by concatenating all object vectors with fixed ordering. For the puzzle domain the first three elements of the percept vector always describe the $[x,y,\theta]$ components of the circular object and elements four through to six are for the square object etc. For the car domain only two blobs are represented, one being the target object and one being a contextually unimportant (distractor) object.

For the puzzle domain an action is represented by a three element vector that determines the horizontal, vertical and rotational movement of the manipulator $\boldsymbol{a} = [\delta\text{x},\delta\text{y},\delta\theta]$. For the navigation domain an action is a two element vector encoding the drive and turn parameters needed to control the vehicle.

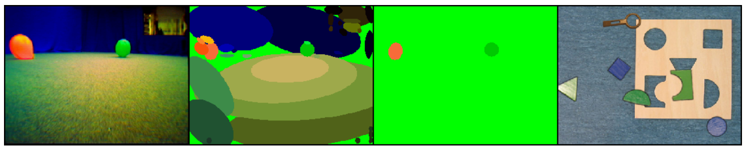


Fig. 2. Percept representation: (a) Image from car camera, (b) output from blob detection algorithm, (c) blobs represented, (d) puzzle domain image

2.2 Hierarchical Organisation

Fundamental to the approach is the use of past experiences in solving new problems. Therefore the system requires a memory of previous experiences, this is the role of the exemplar hierarchy. There are three further factors motivating the development of the hierarchical structure. The first is the need to efficiently locate the most applicable exemplars to the current situation. A binary search tree is well suited for this as it provides a mechanism to discard large areas of the search space early on in the search procedure. Second is the requirement to generalise from experiences already stored in the memory. Figure 3 illustrates this with an example from the autonomous navigation task. The input percept does not match any of the existing percepts and so none of the existing action models will be adequate. However the variance on the position of the percept model that generalises the 'go left' and 'go forward' exemplars is sufficient to allow a match. The selected action model will (if sampled sufficiently - see Sect. 2.3) yield an action that will drive the car toward the target. The third motivation is to provide a mechanism for weighting the importance of perceptual features (see Fig. 3). By clustering the exemplars in the action parameter space,

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

those features that bare no influence on the task are likely to have high variance and therefore contribute less to the matching criteria.

The percept-action hierarchy consists of one root node with two children, each child also having two children and so on right down to the leaf nodes. All the nodes in the hierarchy contain two multivariate Gaussian models, one to represent the action model and one the percept model. The major difference between this and conventional binary search trees is that the data is not clustered in the same parameter space as is used for searching. Instead, the action parameter space is hierarchically partitioned and this partitioning is used in the percept space where the search for a matching percept occurs. The action models are generative (they generate samples) and the percept models are discriminative (they are used to classify incoming percept models).

To build the hierarchy, the action space is first partitioned using a KD-tree decomposition [1, 7], a recursive subdivision of the parameter space into disjoint hyperrectangular regions. The root node is then constructed by building multivariate Gaussians of both the entire action parameter set and the entire percept parameter set. The KD-tree partitioning is then used to recursively build the entire data structure. Multivariate Gaussian models are obtained by collecting all the (action or percept) vectors into a D by N cluster matrix (D is the dimension of the parameter space, N is the number of elements in the cluster), computing the covariance matrix and performing an Eigen decomposition. The Gaussians are parametrised by their mean, Eigen values and Eigen Vectors.

During online operation, the hierarchy is searched for a set of matching percept models. This is achieved by first computing the Mahalanobis distance between the input percept vector and the multivariate Gaussian percept model of the root node. If this distance is less than a fixed threshold, τ (see Sect. 3 for a discussion on selecting τ) then the children are tested recursively in a depth first search. All models that pass the threshold test are returned for evaluation by the probabilistic exploration algorithm with their distances normalised. Returning all the models in this way ensures that both general and specific models can be evaluated.

2.3 Probabilistic Exploration of Action Hypotheses Space

As the hierarchy attempts to generalise the action space and its mapping to percepts it is unlikely that a single random variable sampled from an action model will provide a purposeful action. Therefore multiple samples must be tested. In order to judge the effectiveness of any given sampled action, a generic performance metric is required. The cost of performing an action, given the current state of the world, is defined as the distance between two percepts. One being the result of performing the action on the world, the other being the learnt *goal state*. The goal state is the mean of all solutions (the last state in all training sequences) and the distance function is a weighted euclidean distance (each dimension of the percept space is divided by the variance of that dimension across all end states). This ensures invariance to features that vary across the

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

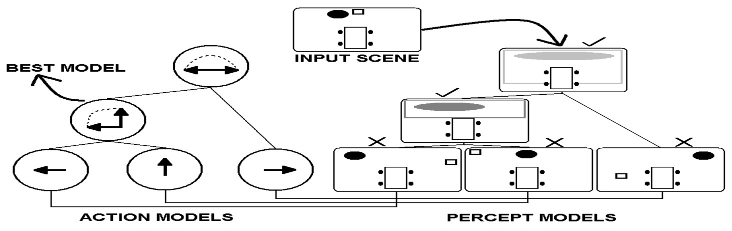


Fig. 3. An illustration of generalising and feature importance weighting capabilities of the percept-action hierarchy. The car has been trained to follow the circular target object, the square object is not important to the task. The input percept is first compared to the most general percept model and as it fits the matching criteria it is compared to both child models. It fits only the left child and is subsequently compared to this models children, both of which it fails to match. Therefore the best matching model is one that generalises over two examples. The associated general action model is thus selected as the most applicable to the current situation.

solutions. When deciding which action to select we have two guiding factors: match likelihood and action cost.

By searching the percept-action store and maintaining multiple hypotheses over a number of time steps, we can overcome local minima and plateaus in our evaluation function. Then by backtracking from the 'best' solution, an appropriate action for the current timestep is identified. This results in operation similar to a traditional tree search procedure except that only branches carrying a high likelihood of success are explored therefore reducing complexity over traditional AI approaches.

The first stage in the multiple hypotheses forward exploration algorithm is to initialise the state of a particle filter: S_0={ θ_0^n,A_0^n,π_0^n,ι_0^n,n=1...N}. Where, for each of the n hypotheses, θ_t^n is the state of the world at iteration t. A_t^n is the action to be performed on θ_t^n. π_t^n is the posterior probability of hypotheses S_t^n and also the prior probability of hypotheses $S_{t+1}^{\iota_{t+1}^n}$, where ι_t^n is the index to the 'parent' particle, S_t^n . N is the total number of particles employed.

At t=0 the set of world states, θ_0^n, are the same for all n i.e. the current state of the world. In order to construct S_0, a set of initial action hypotheses, A_0, must be generated.

Searching the percept hierarchy for the M best matches to the current scene yields M bivariate Gaussian action models, Λ, and M match likelihood values that coarsely approximate the probability of action model Λ given current world state.

$$\rho(\Lambda_m \mid \theta = \theta_0) \approx 1 - \frac{dist(\theta_o, \phi_m)}{\sum_{i=1}^{M} dist(\theta_o, \phi_i)} \qquad (1)$$

where ϕ_m for m = 1 ... M is the set of matched scenes.

Next, N random samples, A_0^n (n = 1 ... N), are taken from the action models. The number of samples taken from model Λ_m being determined by N * $\rho(\Lambda_m \mid \theta = \theta_0)$.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

The Final step in initialising the particle filter is to construct π_0. This is done by setting $\pi_0^n = \rho(\Lambda_m \mid \theta = \theta_0)$ where A_0^n was sampled from model Λ_m. π_0^n must then be normalised to ensure $\sum_{n=1}^{N} \pi_0^n = 1$.

Once initialised, the particle filter must iteratively construct $S_t = \{\theta_t^n$, A_t^n, π_t^n, ι_t^n, n=1...N$\}$ from, $S_{t-1} = \{\theta_{t-1}^n$, A_{t-1}^n, π_{t-1}^n, ι_{t-1}^n, n=1...N$\}$.

Repeat for T iterations i.e. t = 1 ... T.

1. **Perform actions for each hypotheses**: Generate θ_t by performing A_{t-1} on θ_{t-1}.
2. **Compute hypotheses weights** ω_t^n: ω_t^n = 1 - $\frac{dist(\theta_t^n, \Gamma)}{\sum_{i=1}^{N} dist(\theta_t^i, \Gamma)}$, where Γ is the learnt goal state.
3. **Multiply hypotheses prior by weight to obtain posterior**: $\pi_t^n = \pi_{t-1}^n$ * ω_t^n
4. **Normalise, sort and discard hypotheses**: Sort hypotheses according to posterior and discard worst to ensure just N remain. Normalise posteriors $\pi_t^n = \frac{\pi_t^n}{\sum_{i=1}^{N} \pi_t^i}$
5. **Find percept match and sample from distributions**: A best match to the current world θ_t^n is found. The associated action model Λ is sampled to provide A_{t+1}. The number of times Λ is sampled is determined by N * π_t^n. For each new particle formed, ι_{t+1}^{np} is assigned the index value for the parent hypotheses.
6. **Propagate posteriors to next iteration priors**: $\pi_{t+1}^{np} = \pi_t^{\iota_{t+1}^{np}}$
7. **End of iteration**: S_{t+1} is now constructed so move to next iteration.

Once T iterations have been performed, the optimal world state is

$$opt(S_T) = \arg\max_{\forall n}(\pi_T^n) \tag{2}$$

The optimal action at t = 0 is then obtained by backtracking from opt(S_T) through the particle filter states, using ι to indicate particle parents.

The hierarchy can be viewed as a fast search data structure used for scene matching in the visual database (although it is not claimed that organising the percepts according to action clustering improves performance over clustering in the search space). It can also be viewed as a binary classification tree where decisions are made based on a thresholded weighted metric (Mahalanobis distance) where the weighting reflects the importance placed on various perceptual parameters. It is the organising and importance weighting of the perceptual space achieved by projecting the action space clustering into the percept space that is considered the hierarchies most significant value. This, coupled with the implicit construction of a general to specific set of action models, gives the system the power to model the complexity of many input-output processes within one integrated structure.

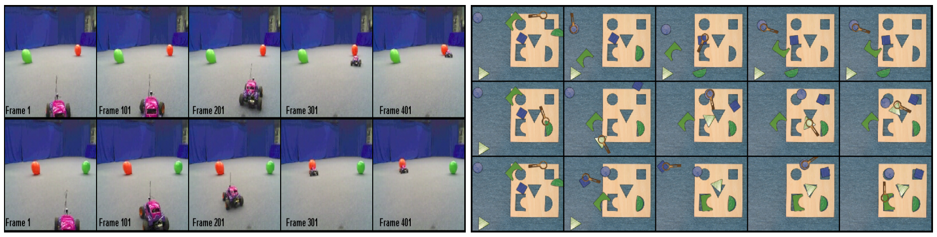
Furthermore, the use of the match likelihood allows the system to move between guided random exploration to goal directed behaviour depending upon the confidence of memory recall.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

3 Experimental Setup and Results

A GUI is used to allow the human teacher to drive the car. The GUI displays the image from the wireless camera attached to the car and allows the user to click on a control grid where the x and y coordinates determine the magnitude of the drive and turn parameters. Each time a move is made the image from the camera is recorded and stored along with the control parameters. The teacher's objective was to drive the car toward the red target, if the red object was not present in the image then the car was turned until the object could be seen. Twenty sequences (each with around 15 actions) were recorded and used to build the hierarchy (304 exemplars in the hierarchy). When building the hierarchy the images were processed to obtain the eighteen dimensional percept representation (2*9 element moment–RGB vector).

Figure 4(a) shows every 100th frame from two sequences showing the car driving itself. In both these tests the car is initially pointing at the green distractor. Twenty tests were made in total with different start conditions for each test. In most cases (15 out of 20) the car drove directly to the target. In some cases (2 out of twenty) the car over steered and had to turn a full 360 degrees in order to bring the target back into view - this was autonomous. In the remaining three tests the car lost sight of the target and drove in circles never managing to bring the target back into view. It is not possible to demonstrate the hypotheses exploration algorithm within this domain as a world model is required for actions to be evaluated.



(a) (b)

Fig. 4. (a) Every 100th frame from two videos showing the car driving toward the target. (b) Scene 1, 5, 10, 15 and 24 for no cost system (top), 1 iteration system (middle), 4 iteration system (bottom).

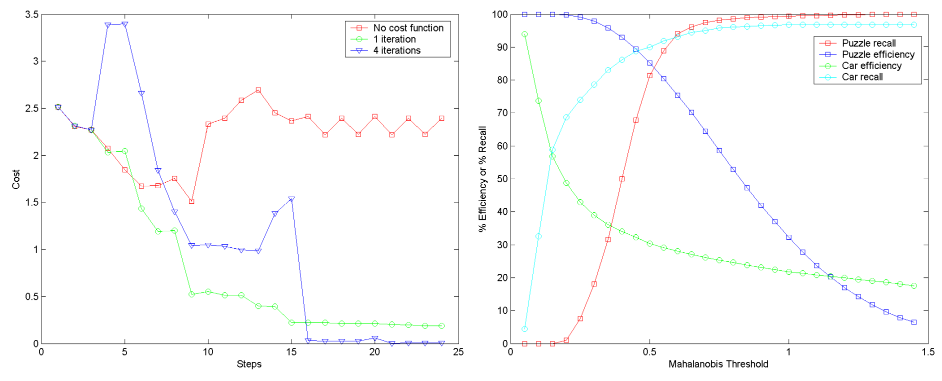
49 puzzle solutions were provided by a teacher each with a slightly different end state. From each end state 10 training sequences were obtained by randomly moving the blocks out of the holes and then reversing the sequence giving 490 training sequences and a total of 2940 exemplars. These exemplars were then used to build the hierarchy.

Figures 4(b) and 5(a) show the effect the probabilistic forward exploration algorithm had when applied in the puzzle domain. The same start scene was input to the system with three configurations: The first configuration does not

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

use the cost function, action selection is determined solely by percept match likelihood and action samples are not evaluated. The second uses the cost function to evaluate initial matches but does not propagate the hypotheses forward in time. The third uses the full forward exploration algorithm performing four iterations of the particle filter and maintaining 20 hypotheses (particles). It can be seen that the use of the cost function improves the performance of the system. Furthermore, by propagating the hypotheses forward a number of time steps and thus allowing intermediate actions to move up the cost surface and into more familiar regions of the percept space, the system is eventually able to achieve the goal state.

Figure 5(b) shows plots for the recall rate and efficiency of the hierarchies. The hierarchies are tested (at various thresholds) with the same data as they are built from and if the exact match is found then the recall count is incremented. Efficiency is defined as 1-(number of nodes searched/total number of nodes)*100 i.e. 0% is an exhaustive search. The best thresholds will be those that produce high recall rates (not rejecting close matches) whilst maintaining high efficiency rates (not comparing to every model). The thresholds chosen for the two domains are 0.5 for the puzzle domain and 0.3 for the car domain. It is worth noting that the efficiency scores are affected by the size of the training set (304 for the car and 2940 for the puzzle). It is also important to note that, even though both general and specific models are important, both the recall and efficiency scores only reflect the performance on exact matches whereas many of the more general models matched are also positive results.



(a) (b)

Fig. 5. (a) Cost function plots for the system when: not using cost function, using cost function to evaluate initial matches (1 iteration) and using cost function in forward exploration algorithm (4 iterations). (b) Threshold vs. recall rate and efficiency for both car and puzzle domain hierarchies. Recall rate is the percentage of correct exact matches and efficiency is 1-(number of nodes searched/total number of nodes)*100 i.e. high efficiency means few nodes searched and low efficiency means many nodes are searched.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

4 Conclusion and Discussion

The results demonstrate that the system is capable of learning to respond appropriately in two quite different problem domains without the need for task specific knowledge. Furthermore, it is shown that the hierarchy allows for generalisation of experiences that allow it to perform in unseen scenarios. It has also been shown that, by allowing forward exploration of multiple action hypotheses, the system is able to overcome local minima in the cost surface and achieve better convergence to the goal state.

In future work the system will be extended to allow learning in environments with multiple objects with no fixed correspondences by developing a robust feature correspondence algorithm. Environments requiring responses to dynamic events i.e. moving objects will also be dealt with in future work. Another avenue of future work will look at ways of replacing the need for a world model for evaluating proposed actions.

References

1. J. L. Bentley. K-d trees for semidynamic point sets. In *6th Ann. ACM Sympos. Comput. Geom. 187 - 197, 1990.*
2. Buccino, G., Binkofski, F., and Riggio L. 2004. The mirror neuron system and action recognition. In *Brain and Language, 370-6, 2004.*
3. http://www.cospal.org/
4. Michael Felsberg and Per-Erik Forssen and Anders Moe and Gosta Granlund. A COSPAL Subsystem: Solving a Shape-Sorter Puzzle In *AAAI Fall Symposium: From Reactive to Anticipatory Cognitive Embedded Systems, 65-69, 2005.*
5. P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini. 2003. Learning About Objects Through Action: Initial Steps Towards Artificial Cognition. In *ICRA'03 vol.3,3140-3145,2003.*
6. P.-E. Forssen and G. Granlund. Blob Detection in Vector Fields using a Clustering Pyramid. In *Technical Report LiTH-ISY-R-2477.*
7. J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. In *ACM TOMS vol.3 issue 3, 209-226, 1977.*
8. T. van Gelder and R. F. Port. Its about time: An overview of the dynamical approach to cognition. In *R. F. Port and T. van Gelder, editors, Mind as Motion Explorations in the Dynamics of Cognition. MIT Press, 1995.*
9. G. H. Granlund. The complexity of vision. In *Signal Processing, vol. 74, 101-126, 1999.*
10. G. Granlund. Organization of Architectures for Cognitive Vision Systems. In *Proceedings of Workshop on Cognitive Vision, 2003.*
11. C. J. Needham, P. E. Santos, D. R. Magee, V. Devin, D. C. Hogg, and A. G. Cohn. Protocols from Perceptual Observations. In*Artificial Intelligence, 167(1-2), 103-136, 2005.*
12. J. M. Siskind. Reconstructing force-dynamic models from video sequences. In *Artificial Intelligence archive, vol. 151, 91-154, 2003.*
13. E. Thelen and L. B. Smith. A Dynamic Systems Approach to the Development of Cognition and Action. *Cognitive Psychology. MIT Press, 1996*

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de