A generalised exemplar approach to modeling perception action coupling

Liam Ellis and Richard Bowden
CVSSP, University of Surrey, Guildford, Surrey, GU2 7XH.
{L.Ellis R.Bowden}@Surrey.ac.uk

Abstract

We present a framework for autonomous behaviour in vision based artificial cognitive systems by imitation through coupled percept-action (stimulus and response) exemplars.

Attributed Relational Graphs (ARGs) are used as a symbolic representation of scene information (percepts). A measure of similarity between ARGs is implemented with the use of a graph isomorphism algorithm and is used to hierarchically group the percepts. By hierarchically grouping percept exemplars into progressively more general models coupled to progressively more general Gaussian action models, we attempt to model the percept space and create a direct mapping to associated actions.

The system is built on a simulated shape sorter puzzle that represents a robust vision system.

Spatio temporal hypothesis exploration is performed efficiently in a Bayesian framework using a particle filter to propagate game play over time.

1 Introduction

We present a framework for autonomous behaviour in vision based artificial cognitive systems by imitation through coupled percept-action (stimulus and response) exemplars.

The assumption is made that if a system has, stored in its memory, a symbolic representation of **all** the possible perceptual stimuli (percepts) that it shall ever encounter each coupled with a 'symbolic' action model (response), then it should be capable of responding as required to any given visual stimulus. Of course biological vision systems exhibit many high-order reasoning and context-dependent abstraction capabilities that go beyond any such simple predefined mappings. Nonetheless, our assumption leads us to consider how a practical **estimation** to such a system could be achieved.

The main practical restriction to the above scenario is the limit on the number of percept-action exemplars that can be realistically 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 the nature of the perceptual input is not known a priori. Practical requirements of the system are that it must be capable of searching its entire percept store (visual memory) in order to find a match to the current percept (visual stimulus), and then perform some associated action.

By hierarchically grouping the percepts into progressively more general representations (expressions), and given that the stored percepts adequately cover the percept space, we can structure the stored percepts in such a way as to allow fast searching. Also in generalising the percept representations further at each level of the hierarchy, the system is capable of compensating for the incomplete population of the percept space by performing more general action models given more general percept representations.

This system operates within a simulated shape sorter puzzle environment. The training phase is initiated by the supervisor solving the shape sorter puzzle a number of times. Each time the supervisor takes some action, the system records both the action performed and the associated visual percept. Once the training data has been recorded the grouped percept hierarchy is built.

During the systems on-line state it extracts the current scene information and builds a symbolic representation, Attributed Relational Graphs (ARGs), which can be compared to the stored percepts. A multiple hypotheses forward exploration - back tracking algorithm based on a particle filter is employed to guide the system toward its learned goal.

In order to allow the system to improve its performance at a given task over time, during its own interactions with the world it stores the resulting percept-action pairs. The system is capable of supervising itself over time by rewarding near optimal action sequences. Optimality is defined in terms of a cost function.

Only preliminary experimental results are available at time of press. The influence of some system parameters and algorithms on the systems performance at the given task are however assessed.

2 Background

It has been argued that it is the "purpose of cognitive vision systems not primarily to build models of the geometry of objects or of scenes, but to build up model structures that

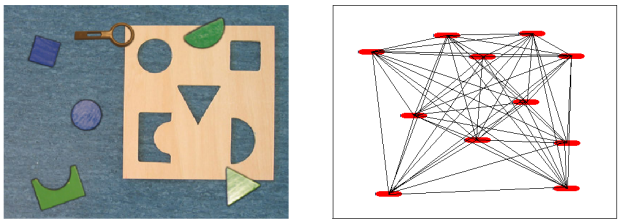


Figure 1: Percepts are represented as Attributed Relational Graphs

relate the percept domain and the action domain"[1]. With this purpose in mind, a system has been developed that attempts to estimate the mapping from perceptions to actions. In the current system, we assume a linear/direct mapping between the two spaces but in future work we intend to generalise to non-linear mappings.

In order for a cognitive system to actively respond to visual stimulus, a mapping between percepts and actions is required. Recent neurophysiological research has shown strong evidence supporting the existence of a mechanism, in both primate and human brains, that obtains this percept-action coupling, known in literature as "direct-matching hypothesis". The mirror-neuron system essentially provides the system (human/primate brain) with the capability "to recognise actions performed by others by mapping the observed action on his/her own motor representation of the observed action" [2].The system presented here has been endowed with these same innate capabilities. Therefore our work differs from that of Siskind, where an attempt is made to analyse from visual data, the force dynamics of a sequence and hence deduce the action performed[3]. Instead, by allowing the system to directly record symbolic representations of the actions performed during the training phase, an exact percept-action coupling can be achieved. As an alternative approach, 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 the tasks[4].

Magee et al[5] have recently presented a framework for learning of object, event and protocol models from audio visual data. Their framework employs both statistical learning methods, for object learning, and symbolic learning for sequences of events representing implicit temporal protocols. As [5] point out this learning of temporal protocols is analogous to grammar learning, in this respect the system presented here shares some goals with that presented by [5]. Further, both systems attempt to achieve this grammar learning through generalising a symbolic data set. There is however very little similarity between the approach taken by [5] (Inductive Logic Programming), and that which we have taken (grouped percept-action exemplars). Where [5] have developed, using Progol, an inference engine to extract temporal protocols, we have employed an *approximation to imitation* approach to learning puzzle grammars.

3 Modelling Perception

This section describes our approach to capturing information from the perceptual domain in order to symbolically represent the scene in an invariant and compact form.

3.1 Capturing scene information

In order to process perceptual data in the symbolic domain a grounded symbolic representation must be obtained.

As the system presented here operates within a simulated environment, many of the common computer vision problems of symbol grounding in terms of feature extraction, classification and object labelling are avoided. Essentially the simulated environment provides a robust computer vision system. That said, exactly what information is needed is of primary concern. The shape sorter puzzle, along with many other potentially autonomous tasks, rely on the ability of the system to identify and classify objects. Further it is essential that the relationships between objects be represented. It is these two elements; scene objects and scene structure that are captured in the representation adopted here.

3.2 Symbolic representation - percepts

Once all the scene information has been extracted it needs to be represented in an invariant and compact form, a percept. In order to group percepts, the representation must allow us to compare, and to find a measure of similarity between two percepts.

Here a scene is represented symbolically as an Attributed Relational Graph. Formally we define Attributed Relational Graphs (ARGs) as a 4-tuple, $g = (V, E, u, v)$ where V and $E (E \subseteq V \times V)$ are the set of nodes (graph vertexes) and edges (links between nodes) respectively. $u : V \longrightarrow A_V$ is a function assigning attributes to nodes, and $v : E \longrightarrow A_E$ is a function assigning attributes to edges. A_V and A_E are the sets of node and edge attributes respectively.

Graph vertexes represent the objects in the scene. Graph edges represent the relational structure of the scene, see figure-1. *Type* attributes are attached to each vertex and dictate the type of object. Graph edge attributes are 3D *relative_position/orientation* vectors that represent both the horizontal and vertical displacement and the relative orientation between the two objects connected by the edge.

As an alternative approach, we intend to make a slight modification of the ARG percept by *emphasising* the importance of contextually important objects. Objects (nodes), and the relationships between them (edges), that are important to the *dynamics* of the modelled scene are therefore considered more important when grouping and matching percepts, see *Percept distance* below.

Bunke et al [7] argued that structural pattern recognition, that uses symbolic data structures such as graphs, is more powerful in terms of representational capabilities than statistical pattern recognition, that uses feature vectors to represent patterns. It is the ability of the structural approach to model structural relationships between pattern features that is of particular interest here. However as [7] point out, the rich set of mathematical tools available in the statistical domain, in particular clustering techniques, are not so readily available in the structural domain. In order to apply statistical clustering techniques in the structural domain, a distance metric between the symbolic representations is required.

4 Perceptual Grouping

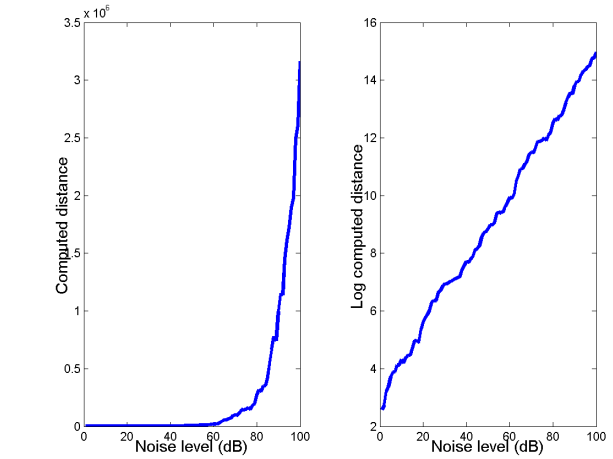
This section describes how we group our percepts into a hierarchical structure and how higher levels in the hierarchy represent more general percepts.

4.1 Percept distance

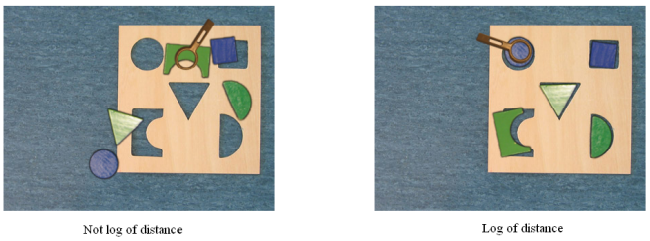
In order to group the percepts we need some way to measure/compute the similarity or distance between two Attributed Relational Graphs.

There are a number of proposed algorithms for the computation of an ARG distance metric;[8],[9],[10],[12]. Cordella et al[13] at the Artificial Vision Group (University of Naples) have developed the VF graph matching algorithm and have provided the graph matching database and VFlib [1], a class library for testing graph matching algorithms. The VF algorithm uses a Contextual Transformational Model for the inexact matching of ARGs.

In our work we have used VFlib and the VF graph/subgraph isomorphism algorithm in order to compute a distance between ARGs/percepts. As our graphs currently always have the same objects in the scene each labelled with distinctive labels that are consistent across graphs, there is always an exact isomorphism between any two graphs. In order to compute the distance between two graphs we therefore compute the cost of the isomorphism/match found. The



(a)



(b)

Figure 2: (a) Distance between original and noisy scene as noise level increases, (left) without taking log and (right) with taking log. (b) Two system are built using ARG distance and log ARG distance methods. When presented with a scene from the hierarchy with 10dB of AWGN added, the log distance system is more successful at the given task.

distance (cost of match) was initially computed by summing the Euclidean distance between the *relative_position* vectors (graph edge attributes) of each matching graph edge. Although this approach did give some measure of similarity (temporally local percept pairs in a sequence are awarded greater similarity than temporally distant pairs), it was found that the distance values obtained increased exponentially when AWGN (Additive White Gaussian Noise) is applied to scene object positions. This non-linearity of distance function is resolved by taking the log of the euclidean distance value. This gives us,

$$dist(G_1, G_2) = \log \left| A_E^{G_1} - A_E^{G_2} \right| = \log \sqrt{\sum_{i=1}^{d} \left| A_{E_i}^{G_1} - A_{E_i}^{G_2} \right|^2} \quad (1)$$

[1] Graph Database: http://amalfi.dis.unina.it/graph/

where $A_E^{G_k}$ is the edge attribute set of graph G_k and d is the dimension of the edge attribute, in this case 3.

Figure-2 (a) illustrates the effect of taking the log of the distance function. Figure-2 (b) shows the improvement in performance of the system when the log distance function is used. See section 7 Experimental Results for details of this experiment.

As mentioned in *Symbolic representation - percepts* above, contextually important nodes and edges can be made to have a greater influence on the graph distance.

This method of computing a distance could be implemented without the use of a graph matching algorithm but it is intended that this system shall be applied to a less robust vision system that may not provide consistent object labels across scenes. In these situations the use of an inexact graph matching algorithm is highly applicable.

The *distance* between two puzzle scenes is often not obvious to specify. Although the method employed here has shown some good results, it is an engineered measure and can not be trusted completely in its ability to measure similarity in a way relevant to the problem context. In future systems it is intended that the distance metric be learned from training data[14]. This work is being carried out concurrently.

4.2 Hierarchical percept grouping

Due to the complexity of the distance computation and the size of the data set, the groups are formed greedily i.e. non optimally.

During grouping, a distance matrix is incrementally populated and used as a look up table to avoid recomputing ARG distances. This is only necessary with large graphs.

At the bottom level of the percept hierarchy, all the percepts in the system memory are present. At the next level up the hierarchy, lev_2, we group these percepts together if they are within the current levels Maximum allowable Distance (MD), i.e. $clust(G_1, G_2)$ if $dist(G_1, G_2) \leq lev_2.MD$. Then the median of each group/cluster is computed and is used to represent the group members at the current level. The median graph is computed by finding the graph that has the minimum sum of distances to all other cluster members. At each successive level the levels MD is increased, the medians from the lower level are clustered and finally the new level medians are computed. Currently the value of MD at each level is engineered, the values are selected through percept distance histogram analysis to roughly ensure that the level MD values divide the distance histogram into approximately even areas. In future work a more principled method of setting level generality threshold would be desirable.

As mentioned in the introduction, each percept is coupled to an action model or response. For the percepts in the bottom level of the hierarchy i.e. ungrouped percepts, this model is a one-to-one mapping to an action vector. However for groups of percepts, formed through the perceptual grouping procedure described above, the coupled action group is modelled as a multivariate Gaussian probabilistic distribution. Section 5 below gives the details of the action models used.

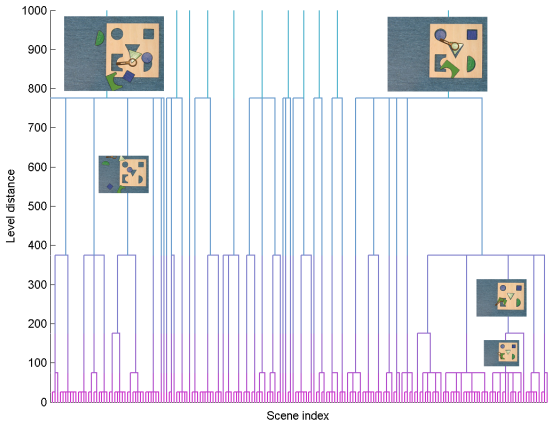


Figure 3: Percepts are clustered into a hierarchy.

5 Modelling Actions

The lowest level actions that the simulator system is capable of performing are 'pick up object', 'put down object' and 'move gripper'. Within the context of the problem the system needs only to perform these actions in a fixed format: A:= move - pick up - move - put down. Since such a compound action may generate multiple intermediate percepts, *key scenes* corresponding to the beginning of a sequence are extracted and coupled to the action model. It is these key scenes, represented as ARGs, that form the data set of the systems visual memory. By temporally segmenting the continuous perceptual input in accordance with the beginning of our compound action sequences we can discard all the perceptual data not extracted as key scenes.

Another motivation for forming compound action sequences is that a fixed length vector can be used to represent actions. V_A = (δ_{x1},δ_{y1},δ_{x2},δ_{y2},δ_θ). The first and second elements of the action vector represent the initial movement prior to 'pick up'. As 'pick up' and 'put down' need no parameterisation being implicit to the action, they are not represented. The last three elements of the action vector are therefore left to parameterise the post 'pick up' move operation with a rotation.

Now that an action can be modelled as a vector, the actions coupled to a percept group can be represented as a

Gaussian. The actions associated with a percept group are collected together into a 5 by N matrix, where N is the number of percepts in the group. By computing the zero mean covariance matrix and performing an Eigen value decomposition, a multivariate Gaussian probability distribution is obtained.

6 Perception-Action Cycle

Many of the activities that cognitive systems must perform require perceptually guided action e.g. walking or catching. It is also true to say that much of what is perceived by a cognitive system is the result of actions performed by that system e.g. moving to a new space in the environment or picking up an object to examine it. Here, the perception-action cycle simply describes a model of behaviour whereby perception influences action and action influences perception.

6.1 System overview

The system extracts a symbolic representation, percept, from the current sensory input. It then finds a set of closest matching percepts in the percept hierarchy. Each match is associated with an action model whose variance depends on the level in the hierarchy at which the match was made.

In initial experiments the system simply selected the best matching percept and performed the associated action - MLE (see Section 7). This approach worked in some cases where the input scenes were present in the percept hierarchy. In some cases, given an already seen perceptual scene, it was capable of entirely solving the puzzle. However, given any input the system had not seen (did not have in its percept store), the system randomly performs actions i.e. it does not generalise well. In many cases even with seen data the system would find itself 'stuck' in an action loop.

To allow the system to deal with perceptual stimuli not already encountered in a more guided fashion, we introduced a cost function. We also introduced a multiple hypotheses forward exploration algorithm to optimise action selection. See section 6.2.

6.2 Cost function, probabilistic framework and multiple hypotheses search strategy

The systems inability to solve puzzles from unseen percepts can be attributed to number of factors. Firstly the current system only has 1000 percept-action couples in its store, this is unlikely to provide an adequate coverage of either percept or action spaces i.e. the spaces are under populated. Secondly the limitations of the distance measure currently used, as mentioned above, will hinder the systems performance.

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 a certain set of action parameters given the current state of the world is defined by the distance between two percepts. One being the result of performing the action on the world, the other being a learned *solved state*. Although no formal justification of this evaluation function has been made, experimental evidence does suggest it to be of some value i.e. the function decreases as the solution is approached.

When it comes to deciding what action to take we have two guiding factors: similarity of a percept match and cost of performing an action. Posing this in a probabilistic framework allows us to combine the factors in a principled manner. Also the potential for including other probabilistic factors such as a prior on particular regions of the percept space or action transition probabilities becomes available. To do this we use a particle filter framework.

By searching the percept-action store maintaining multiple hypotheses over a number of time steps, we can overcome local minima and plateaus in our evaluation function. By backtracking we can then obtain the optimal action at the current step.

The first stage in the multiple hypotheses forward exploration algorithm is to initialise the state of a particle filter, S, at t = 0.

$$S_0 = \{\theta_0^n, A_0^n, \pi_0^n, \iota_0^n, n = 1...N\} \tag{2}$$

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.

$$\rho(\Lambda_m \mid \theta = \theta_0) \approx 1 - \frac{dist(\theta_o, \phi_m)}{\sum_{i=1}^{M} dist(\theta_o, \phi_i)} \tag{3}$$

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 * $\rho(\Lambda_m \mid \theta = \theta_0)$.

The Final step in initialising the particle filter is to construct π_0. This is done by setting π_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

$$S_t = \{\theta_t^n, A_t^n, \pi_t^n, \iota_t^n, n = 1...N\} \tag{4}$$

from,

$$S_{t-1} = \{\theta_{t-1}^n, A_{t-1}^n, \pi_{t-1}^n, \iota_{t-1}^n, n = 1...N\} \tag{5}$$

for t = 1 ... T.

This is achieved by performing T iterations on the following steps.

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.

$$\omega_t^n = 1 - \frac{dist(\theta_t^n, \Gamma)}{\sum_{i=1}^{N} dist(\theta_t^i, \Gamma)} \tag{6}$$

   where Γ is the learned solved state.

3. Multiply hypotheses prior by weight to obtain posterior.

$$\pi_t^n = \pi_t^n * \omega_t^n \tag{7}$$

4. Normalise, sort and discard hypotheses. Sort hypotheses according to posterior and discard worst to ensure just N remain. Normalise posteriors

$$\pi_t^n = \frac{\pi_t^n}{\pi_t^n * \omega_t^n} \tag{8}$$

5. Find world state match and sample 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.

$$\pi_{t+1}^{np} = \pi_t^{\iota_{t+1}^{np}} \tag{9}$$

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

$$opt(S_T) = \arg\max_{\forall n}(\pi_T^n) \tag{10}$$

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.

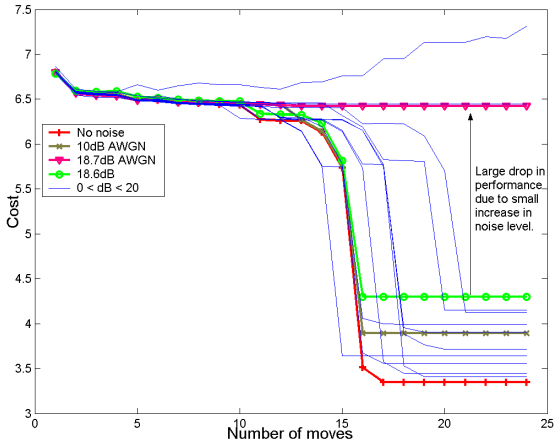


Figure 4: Cost plots for sequences of play with different levels of noise applied to (already seen) input.

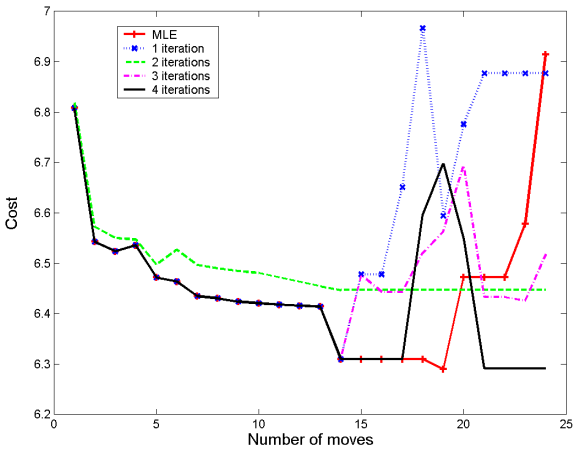
7 Experimental Results

Figure-2 (a) shows the results of an experiment designed to examine the nature of the developed distance function. A *seed* percept is taken from the hierarchy and new percepts are generated by applying increasing levels of AWGN (Additive White Gaussian Noise) the the percepts object locations. The distance from each of the noisy scenes to the seed scene is computed and plotted both before and after taking logs. Figure-2 (b) shows the resulting puzzle state at a given task for two versions of the system. One system that uses the log distance (to group and search for percepts as well as to provide cost and likelihood values) and another system where the log is not taken. The original input scene in this case is a scene from the hierarchy with 10dB AWGN applied.

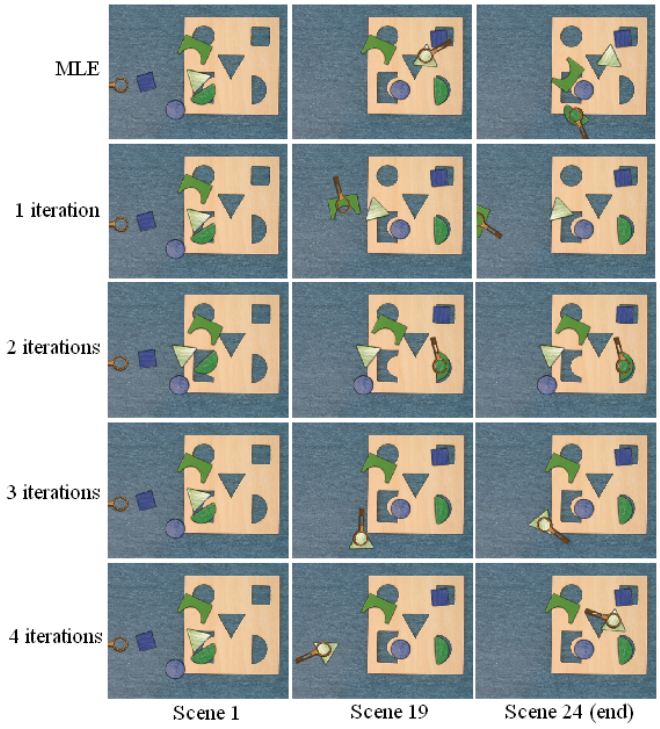
Figure-4 shows the results of an experiment designed to determine the noise level at which the MLE version of the system (no multiple hypotheses search strategy - just match likelihood) will fail on a given sequence. From a percept present in the hierarchy, a set of new initial puzzle states are generated by applying AWGN (from 1dB to 20dB) to the object positions of the percept. In general as the noise level increases the system performance (in terms of the cost function) decreases. As can be seen from Figure-4 there is a point at which system performance drops greatly (the cost plots fail to reach low minima). It is at this noise level (the breaking point of the MLE system on this sequence), that the multiple hypotheses forward exploration strategy system is tested. The reason for this is that this noise level is the lowest level at which any improvement can be observed in system performance due to the exploration strategy. For the sequence selected the noise level found was 18.65dB.

In the following experiments the multiple hypotheses exploration algorithm has a total particle population, N = 40, so 40 hypotheses are maintained at each iteration. The number of matches found when initialising the particle filter is M = 10.

The system was run, with the same initial puzzle state (generated by applying 18.65dB noise to object positions of percept from the hierarchy), with 4 different depths of hypotheses exploration. All the runs were stopped after 24 moves. The costs for the resulting sequences are plotted in Figure-5 (a) along with the cost of the MLE sequence. It can be seen that the sequence with the optimal end cost is the sequence generated by the system that applies 4 iterations of the multiple hypotheses forward exploration algorithm while the least optimal sequence comes from the MLE system. Figure-5 (b) shows scenes from each of the sequences in (a) after the 1st, 19th and 24th moves.



(a)



(b)

Figure 5: (a) Cost plots of sequences of play when 18.65dB AWGN applied to input scene. The number of iterations refers to depth of the hypotheses exploration. (b) Scenes from the sequences represented in (a) at step 1, 19 and 24.

8 Conclusion, discussion and Future Work

When exact matches to perceptual stimuli are present in the percept hierarchy the system can solve the puzzle. Furthermore, when no exact match to the perceptual stimuli is present in the hierarchy the system is capable of identifying the most similar percept and, within a certain range of similarity, completing the puzzle - MLE (Figure-2 (b)). This suggests that it is able to generalise. By combining match likelihood and action cost values in a Bayesian framework, the multiple hypotheses forward exploration strategy appears to allow the system to further extend its generalising capabilities and its ability to solve the puzzle.

During the sequences shown in Figure-5, the MLE system moves one of the blocks (half circle) out of it matching hole. This action selection is based purely on similarity of percept match and has no regard for the problem context - provided by the cost function. The multiple hypotheses system with a forward exploration depth of 4 is shown in Figure-5 to perform actions that also result in a higher cost value (the highest over all hypotheses at step 19), however in this case the system selects the hypotheses based on the fact that there is a lower minima on the cost function once the local maxima has been overcome. In fact the system moves the triangle well away from the board and in so doing changes the puzzle state such that a percept-action couple is found that brings the triangle over the correct hole.

The results presented here come from preliminary experimental data. Clearly in order to fully evaluate the systems performance a great number of tests must be run. This is partly due to the random nature of the sampling process and partly due to the number of system variables involved. For example the number of particles/hypotheses that the system

maintains at any time and the effect on performance is to be investigated further. The number of initial matches that are found when initialising the particle filter may also effect performance.

This system demonstrates an ability to solve shape sorter puzzles without any hard coded rules (currently only with puzzle states similar to states that it has seen before). It is believed that this approach could be employed to aid unconstrained automation, endowing human like attributes of adaptability into systems. There are any number of environments where no amount of hard coded rules account for all eventualities and it is these environments where the framework presented may provide some benefit.

In future work, a number of improvements to the system are to be included. As mentioned already, we intend to weight contextually important nodes and edges when computing similarity values. Initial testing has shown that this may lead to more meaningful action models formed as a result of perceptual grouping. It is also expected that by increasing the size of the exemplar set, the ability of the system to generalise will increase. Further, both the similarity measure and the cost function are expected to become more relevant to the context problem when the learned distance function is applied[14]. This will improve the Maximum A Posteriori (MAP) estimate made at the end of each particle filter iteration. Another way to improve the MAP estimates is to include more a priori information such as action transition probabilities. Finally, it is thought that some problems will require a more complex mapping (than one-to-one) between percepts and actions. This will also be addressed in future work.

Acknowledgements

Part of the work presented here was supported by the the European Union, grant COSPAL (IST-2004-71567)[1] This work is also partly funded by EPSRC through grant EP/P500524/1 to the University of Surrey.

References

[1] Granlund, G. 2003. Organization of Architectures for Cognitive Vision Systems. In *Proceedings of Workshop on Cognitive Vision*.

[2] Buccino, G.; Binkofski, F.; and Riggio L. 2004. The mirror neuron system and action recognition. In *Brain and Language, volume 89, issue 2, 370-376*.

[3] J. M. Siskind 2003. Reconstructing force-dynamic models from video sequences. In *Artificial Intelligence archive, Volume 151 , Issue 1-2 91 - 154*.

[1]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.

[4] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini. 2003. Learning About Objects Through Action: Initial Steps Towards Artificial Cognition. In *2003 IEEE International Conference on Robotics and Automation (ICRA)*.

[5] Magee D., Needham C.J., Santos P., Cohn A.G. and Hogg D.C. 2004.Autonomous learning for a cognitive agent using continuous models and inductive logic programming from audio-visual input. In *Proc. AAAI Workshop on Anchoring Symbols to Sensor Data, 17-24*.

[6] Nock R. and Nielsen F. 2004. Statistical Region Merging. In *IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 26, Issue 11, 1452- 1458*

[7] Bunke, H., Gnter, S., Jiang, X. 2001. Towards bridging the gap between statistical and structural pattern recognition: two new concepts in graph matching. In *S. Singh, N. Murshed, W. Kropatsch (eds.): Advances in Pattern Recognition - ICAPR 2001, Springer Verlag, LNCS 2013, 2001, 1 - 11*

[8] Bunke, H., Shearer, K. 1998. A graph distance metric based on the maximal common subgraph. In *Pattern Recognition Letters archive Volume 19 , Issue 3-4 , 255-259*

[9] Wallis W., Shoubridge P., Kraetz M., Ray D. 2001. Graph distances using graph union. In *Pattern Recognition Letters, volume 22 , Issue 6-7, 701-704*

[10] Fernndez M.L., Valiente G. 2001. A graph distance metric combining maximum common subgraph and minimum common supergraph. In *Source Pattern Recognition Letters archive, Volume 22, Issue 6-7, 753 - 758*

[11] Bunke H. 1997. On a relation between graph edit distance and maximum common subgraph. In *Pattern Recognition Letters, volume 18, 689-694*

[12] Robles-Kelly A., Hancock E. 2005. Graph Edit Distance from Spectral Seriation. In *IEEE transactions on Pattern Analysis and Machine Intelligence, volume 27*

[13] Cordella L., Foggia P., Sansone C., Vento M. 1996. An efficient algorithm for the inexact matching of arg graphs using a contextual transformational model. In *Proceedings of the International Conference on Pattern Recognition, volume 3, 180-184*

[14] Ong E., Bowden R. 2005. Learning multi-kernel distance functions using relative comparisons. To appear *Pattern Recognition Journal, Available online*.