What is Intelligence?: Generalised Serial Problem Solving.

N. A. Thacker, A.J.Lacey and P.Courtney.

Last updated
6 / 11 / 2007

TINA

WWW.TINA-VISION.NET

Imaging Science and Biomedical Engineering Division,
Medical School, University of Manchester,
Stopford Building, Oxford Road,
Manchester, M13 9PT.

Generalised Serial Problem Solving

N. A. Thacker, A.J.Lacey and P.Courtney

Imaging Science and Biomedical Engineering, University of Manchester

Abstract

We have worked to develop modules within the TINA vision system for many years. Though the initial research program explicitly considered the development of an intelligent robotic vision system much of this could can be said to have run into the ground due to fundamental lack of extendibility with the approaches being taken. Increasingly more recent work has focused on improving individual modules, such as the feature detection, tracking and stereo. The aim of this document was to re-assess the possibility of constructing systems which could be considered intelligent, in that they support useful and non-trivial visual competences. To do this we decided to go back to the very definition of what it means to have intelligence and try to better understand the challenge which confronts us. In the process a putative computational structure has been identified which supports the construction of visually guided actions as the result of intelligent reasoning. The ideas presented here may be used as the basis for a proposal for funding, should the right opportunities present themselves. In the meantime, short of actually building an intelligent system, the ideas will be used to constrain approaches we might take in other aspects of computer vision research, such as object recognition and vehicle navigation.

1 Introduction

The area of machine vision has for a long time concentrated on the solution of individual computational tasks, such as the extraction of 3D measurements or object location. Many of these areas are now reaching maturity and it is getting close to a time where these solutions should be combined into practical systems which solve sophisticated control tasks. The ultimate form of such a system would be one which could learn to solve visual control problems, such as automated assembly or visually guided motion. For an example of a high performance generalised automated problem solving system we need look no further than the human mind. Understanding the mind has long been a fascination of both scientists and philosophers the world over. This has resulted in many philosophical arguments which characterise (describe) what we can observe as the internal process, but with little clue as to the computational processes required to support them [1]. In artificial intelligence this quest has been driven by the belief that, whatever it is the mind does, ultimately it can be replicated on a processing architecture other than the brain. Indeed it is our belief that simulation of 'mind-like' systems will be an essential process in our path to understanding it. In the process we hope to understand how to build artificial systems with non-trivial processing capabilities. This document is therefore intended to provide a computational model which generates characteristics which might be interpreted as intelligent. In the process we provide explicit engineering based definitions for the mechanisms and tasks required. It is our belief that with the correct resources and motivation, systems such as this could and will be constructed in the future. The main obstacle to their current existence could be one of political will and a lack of research co-ordination.

2 Modelling the Mind

Let us first put into context what we mean by 'the mind' and how it differs from 'the brain'. The simple answer is that mind is what the brain does (or to be more accurate, the mind is a functional model of what the brain does.) Richard Dawkins, in his book *The Selfish Gene*, describes the mind as;

> "...a user friendly way of experiencing the brain."

To fully appreciate what this means consider the following computer analogy. When a programmer wants to add two numbers together it is most likely they would use a high-level computer language such as 'C'.

---

[1]See for example D.C. Dennett's Consciousness Explained, 1991.

However, in order to actually perform this calculation the computer itself uses an intricate sequence of electrical signals. The syntax of the language is the user friendly way the programmer sees the computer. It provides him with a way of modelling what the computer (and the data being represented) are doing in, from the electrical point of view, an abstracted way. Thus, while we might want to take account of what we believe to be achievable computations in brain tissue, we should aim to construct our description of the mind in a similar abstract way. As an appropriate model of the functional behaviour of the brain. The level of abstraction must be sufficient to enable 'mind-like' behaviour to be reproduced on an artificially engineered processing architecture.

3   What is Intelligence?

Though the task of computer vision is recognised as being 'AI' complete (ie: it requires AI to solve it), we do not see intelligent systems at computer vision conferences. Otherwise, the area of computer science is crammed full with approaches to artificial intelligence. The topic seems to be a motivation for the development of; new languages (PROLOG etc.), formal methods (first order predicate logic [4]), hardware (subsumption architectures) and new algorithmic devices (Fuzzy Logic [7], expert systems [6]). Hopefully, any meaningful insights into the workings of intelligence will explain why these endeavours have failed to spawn a generation of useful robotic helpers.

In the light of all these attempts, the question is often asked of whether a computer will ever have real intelligence /understanding /consciousness, but the answer to such a question is complicated by the fact that standard (dictionary) definitions for these concepts are cyclic or rhetorical. They are relational rather than mechanistic, and can form no basis for the design of a system. Thus, much as we accept that the freedom to redefine our own terms reduces the semantic purity of any achievements (we can achieve anything we like by redefining it [1]), in order to make more progress we **need** to replace these with more solid ones.

One entertaining (and quite apt) definition is ;

> " Intelligence is what we use when we don't know what to do. "

This definition is humorous in its clear avoidance of the issues. Yet it is echoed by animal behaviorists, who define intelligence as the ability to develop skills outside those normally required in the daily environment. Such definitions show us that human intelligence is at one extreme of a spectrum of capabilities illustrated in the natural world. However, while such definitions are useful for categorisation, they do not (as made more obvious by the first version) define the essentials of the intelligent process, leaving us with little clue of how the process might proceed.

The most common definition of intelligence in this respect, is that suggested by Alan Turing. The 'Turing Test' defines an intelligent system as one which cannot be distinguished from a human by another human. This definition breaks our links with the rest of the animal kingdom, placing intelligence solely in the brain of homo-sapiens. We need to remember that the reason we are interested in intelligence in the first place is that we believe that an intelligent system is capable of solving useful problems in the real world. Unfortunately, many of the classic systems which score well on the basis of Turing's criteria (PARRY, ELIZA) are nothing more than clever fakery and get us no closer to a useful mechanism. We therefore have to be careful about using such a definition to drive the goals of research. We may hope that the correct definition would allow us to construct systems which could start with poor (but useful) performance and be refined and extended until a human-like capability is ultimately generated. Alan Turing's definition may well have unintentionally seen to it that the area of artificial intelligence has very few candidates on this path. In fact from this point of view our first definition may have been more useful (and more respectful of dolphins and apes).

3.1   Engineering Intelligence.

An informative definition of intelligence would be mechanistic, rather than categorical or relational. Unfortunately, this requires knowledge of the computational process, and we have very little understanding of the way our brains work when we solve problems. However, we are all vaguely aware that in order to solve a problem we gather together facts which restrict a solution and devise possible candidates. Often

a solution will just appear to pop out of the process and at that moment we are able to evaluate it and see immediately if (and why) it is likely to work. In order to have arrived at such a solution there is every possibility that multiple alternatives need to be considered at the same time, and that only one action is selected for ultimate execution. Such an approach has been referred to by Dennett as the "multiple drafts" model [3]. Unfortunately, this idea lacks any computational model and if we are not careful the practical issues involved in building a working system will stumble on the problem of explaining how it is we can select between alternative actions. Indeed, it is easy to see how this selection could itself be the "intelligence" which does all of the difficult work, so that this approach gets us no closer to building a useful system. Like many ideas in this area (such as the higher level systems in Brook's sub-sumption architecture) the difficult part of the system has been moved into a box and labelled in a way which only looks (naively) like it should be simple to solve.

Introspection of our own problem solving tells us that the solution we identify is generally a series of connected actions, which if executed in the correct sequence will result in achieving our goal. It is this property of the solution which we will now use to help define a mechanism for problem solving and intelligence. For example; consider these engineering driven definitions for mental behaviour based on a system containing a sequential solver. We start by defining some basic terms;

- state : a measurably distinct set of internal and external circumstances.

- representation : a derived description of the environment sufficient to identify specific circumstances (*states*).

- manipulation : the modification of *representations* by the system in order to generate a different *state* (including physical actions).

With these definitions we are already assuming that the system will be capable of representing the world and forming a useful grammar of *manipulations* with this *representation*. The key definition being that a manipulation will result in another *state* (from the set of all possible *states*) and not just a random *representation*. We are now in a position to define the following;

- reasoning : a process which identifies particular sequences of *manipulations* required (or likely) to achieve a particular *state*.

- intelligence : the ability to learn from available sensor data without external supervision and modify decision processes (*reasoning* or *manipulation*) in order to improve the probability of achieving a future *state*.

- understanding: a set of learned *manipulations* which enable *reasoning* on a particular set of *representations*.

With the above definitions we can consider 'thinking' as the processes of the *manipulation* of the extrinsic (sensory originated) and intrinsic (consciously formed) *representations* in order to form new *representations*. Notice that this particular definition of *intelligence* includes aspects of direct exploration and learning for those circumstances where the system does not know what else to do. It also deliberately includes a probabilistic definition of prediction which fore-warns of an inherently statistical approach to decision making and system modification (learning). It would be useful to be able to describe this process mathematically and the need for *manipulation* implies something more than conditional notation (ie: $P(A|B)$ used in probability theory). Such notation is capable of only describing correlations between two observed states (A and B). In order to specify a unique *manipulation* M we need $P(A|M|B)$ (ie: the probability that state A will result form state B following action M). This is more like the methods used to describe state transitions in statistical physics (the probabilistic equivalent of Turing's universal processor). In addition, the best course of action may be one which maximises the (frequentist) probability of achieving the desired result while at the same time attending to a set of non-commensurate costs, such as time and resource. These observations suggest that we will not construct intelligent systems by simply turning the handle with Bayes Theorem.

The above definitions provide an immediate interpretation of familiar thought processes in terms of achievable engineering solutions. The definitions may appear quite restricted in comparison to the broader use of English language, but no-one could deny that systems which could be endowed with such characteristics would indeed have useful capabilities. This is the main criteria by which we would like our definitions to be judged. However, as we will show below, we can now also go on to try to identify the role of more abstract mental characteristics.

3.2 Engineering Consciousness

Our everyday experiences lead us to divide the mind into two parts; the conscious and the subconscious. The subconscious mind can be thought of as representing the low-level processing within the brain. It models the direct interaction of brain with body. This part of the mind is responsible for filtering and *representing* data received from the senses and also decomposing into actions the high level information received from the conscious mind. The subconscious mind performs the automatic processes which we find difficult, if not impossible, to verbalise; co-ordination of leg and foot muscles when walking; the synchronisation of our lips and tongue when we speak; the calculations necessary to intercept and catch a moving ball.

This description immediately brings to mind the idea of a system which is attending to a series of tasks (some of them delegated to sub-systems) while at the same time monitoring progress of several factors so that alternative problems can be addressed should they become important. Clearly, we can generally only perform one conscious task at once, so we need a way of brokering between alternatives, simular to the way interrupt driven operations are selected in a multi-tasking computer. So, based upon the assumption that there will be such things as consciousness and sub-consciousness, we will now attempt to describe a few terms couched in the terminology of a scheduled system with interrupts, in relation to our earlier definitions.

- sub-conscious (1).

  The use of frequently updated *representations* to generate an interrupt mechanism to stop current task processing and attend to other circumstances.

- sub-conscious (2).

  The use of previous *reasoning* or knowledge given at construction for semi-automatic (sub-conscious(1)) execution.

- consciousness.

  The process of continually monitoring the incoming sensor data (and therefore changing *representation*) to ensure that a planned *manipulation* task is being executed correctly.

- concentration.

  The ability to attend to a task with a relative minimum of *sub-conscious(1)* interrupts.

- self-awareness.

  The ability to *understand* the systems own *sub-conscious(2)* factors.

- self-consciousness.

  The ability to predict the consequences of *manipulations* on the system as well as the environment.

Finally, and really for fun, we can attempt to define concepts generally attributed to character. In particular we can define;

> personality : a set of behaviours generated by commonly selected *manipulations* used during *reasoning*, generated either at construction or during learning.

eg: selfishness; (undue) use of *self-awareness* to influence *reasoning*.

and impatience; an ongoing evaluation of the rate of task completion which may spur further *reasoning* to identify quicker solutions.

Notice that these definitions are suggestive of approaches which may form the basis of quite useful algorithmic devices. Clearly, many other common English words corresponding to behaviour could be considered as clues as to what practical mechanisms are likely to be of value in a working system, but the existing list of examples should be enough to make the basic point. This is that an interrupt driven serial problem solver could be engineered to display aspects of 'personality'. The approach is therefore not inconsistent with the Turing test.

4   The Minimum Intelligence Architecture (MIA)

The above description of intelligence perhaps lacks the philosophical musings generally associated with introspective analysis, but has the advantage of providing tangible definitions. From these definitions, a system capable of learning to plan and monitor sequences of actions may be very close to the naive implication of "artificial intelligence". Indeed many existing systems could already be said to embody one or more of these factors.
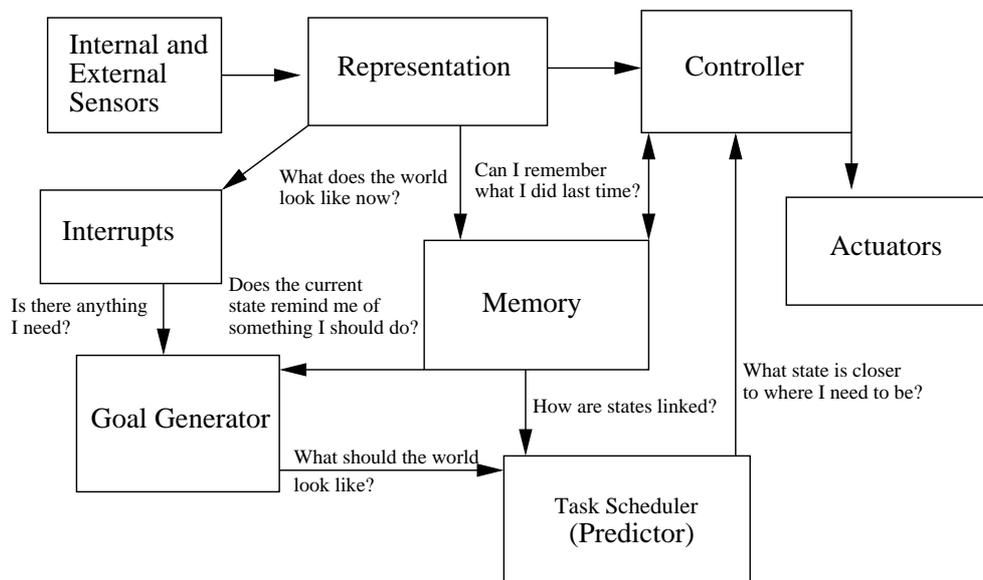

Figure 1: Minimum Intelligence Architecture.

The figure shown in figure 1 could be said to satisfy these criteria if the modules fulfilled the following functions.

- sensors: examples include auditory and vision.

- representation: examples include speech, object recognition, location, 3D measurement.

- interrupt generator: take the current *representation* and generate an interrupt when it matches specific *states* which have been specified a-priori.

- memory: previous examples of sensory *representation* and associations stored in a manner suitable for planning tasks.

- goal generator: take the current analysis of environment and system and select target *states*.

- task scheduler: generate a *reasoned* list of actions and allowable interrupts storing generic information for future use.

- controller: take the current list of tasks and execute them using feedback from the current *representation*. Feed execution timing back to the task scheduler.

- actuators: the physical robot may require; wheels, robot arms, steerable vision system.

The first thing to say is that this architecture (which looks a lot like what people now call a system of autonomous agents) avoids the problem inherent in many approaches to understanding intelligence, of somehow hiding the difficult parts in a complex and poorly defined module. It avoids a central control (or 'meaner' in a Cartesian Theatre as Dennett would say), and does not imply specialised knowledge in one module in order to deal with the inadequacies of later stages (as much of Brook's work on layered

architectures would appear to imply). We do have a 'goal generator' but this needs to be nothing more complicated than identifying hunger or thirst.

Of course, specifying the modules is a long way from generating a working system and means nothing until we can at least postulate a straw man solution. However, much of the architecture looks very familiar and much like a conventional operating system with some sensors and robotic manipulators, none of which has much mystery. The presence of interrupts and control devices implies that this system is inherently parallel, with regular serial signals along the identified arrows. The complicated part of the architecture is in the task scheduler and centres on the way in which arbitrary concepts can be stored in a framework which permits problem solving. This requires careful thought.

The brain is understood to be a parallel processing system and yet our own experiences of problem solving and perception are based on a sequential understanding of the world. The reason for this could well be that our brains parallelism is exploited mainly as a content addressable memory, ie: it is the fact that multiple memories can be accessed simultaneously with the most useful response being elicited rather than there being a parallel problem solver. Individual problems may well be solved serially in a goal directed manner.

4.1   Serial Problem Solving on a Known State Map

In order to perform planning we require a map of the state space. Such maps have immediate uses for tasks such as clustering, see for example colour segmentation [2]. These maps can be generated using the data available in a single scene. Planning tasks generally require information to be accumulated via a more sophisticated mechanism. In previous work we have attempted to solve the most straightforward planning problem, that of path planning in a restricted environment using visual information [5, 8]. This basic scheme requires the ability to generate a visual map of an environment via exploration. The exploration process generates not only *state* vectors of visually distinct locations but also the *manipulations* required to move from *state* to *state*. The basic planning process is illustrated in figure 2 and is an instance of the fire-burning algorithm (on a free-form self-generated topology), and is similar to the A^* algorithm. The basic idea is a follows:

1. The current *state* (ie: visual location) and target *states* are identified.

2. Starting from the current *state* connections representing allowable *manipulations* are followed to nearby *states*. The cost of arriving at this location is computed (as for example the minimum iteration number to arrive at that *state*)

3. The process is iterated by following connections to newly reachable *states* ($T = 1 - 6$ in figure 2).

4. The process ceases when we reach one or more target *states*.

5. The map is then traversed in reverse from the target selecting the minimum cost *state* at each step (dotted curve).

6. The next suggested physical *manipulation* is the minimum cost link from the start which is executed in a closed visual control loop using the computed visual *representation*.

The whole process is analogous to the dynamic programming algorithm, and up to stage 5 is almost certainly identical to the GPS route planners now used by many motorists. However, without a GPS, our current *state* location must be identified using noisy sensor data, thereby forcing operation of these algorithms to be statistical. In particular probability theory is required to assess the effects of sensor error on the *representation* and take correct account of this in the identification of the current *state* and the *manipulation* between *states*. Such a process is equivalent to a pattern recognition task, in particular the methods of Hidden Markov Models.

The algorithm can be said to be optimising a function of the form

$$Q \;=\; \sum_j C(S_i|M_j|S_k)$$

Where a chain of *manipulations* M_j are determined which link the current *state* to any targets. If necessary, the cost function can be related to probability of success (eg: $C(S_i|M_j|S_k) \;=\; -log(P(S_i|M_j|S_k))$).

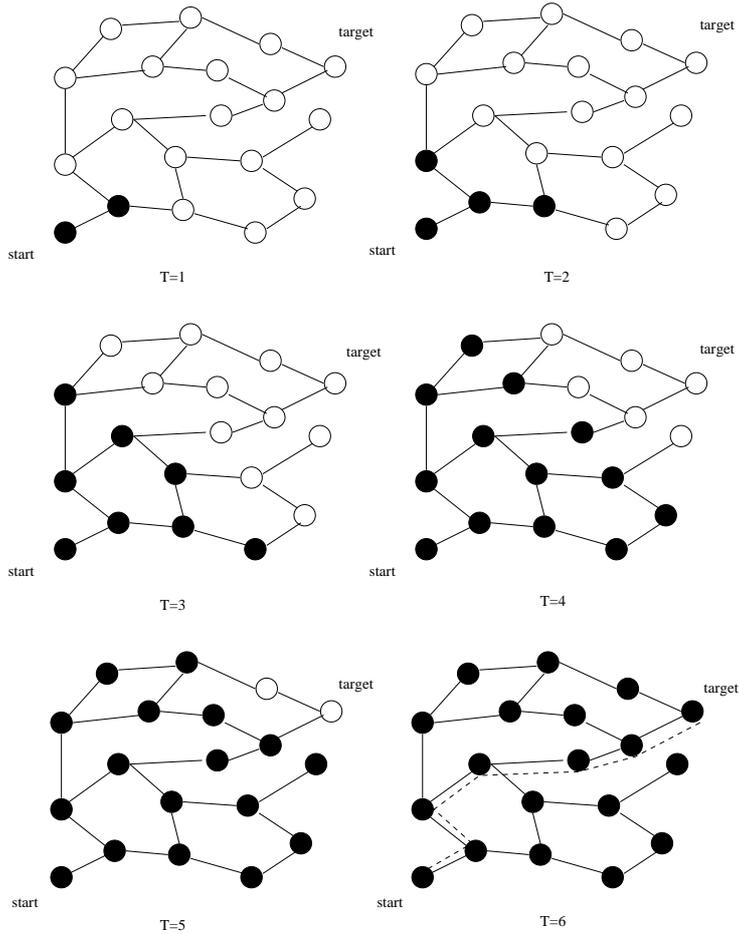The key features of this algorithm are as follows;

Figure 2: Route finding through a state space.

- The ability to specify multiple target *states* means that we do not miss the opportunity of identifying several equivalent goals in one pass of the algorithm.

- The process can execute on any representable topology (and dimensionality).

- The minimum cost path is solved in linear time for a fine grain parallel architecture (re-introducing the role of parallelism for practical efficiency).

- Multiple costs can be used in order to plan for a variety of circumstances (shortest route, quickest time etc).

- Re-planning after each physical *manipulation* eliminates problems due to errors in the map construction (ie: ambiguities).

We have used this technique in our solution for map building and path planning, though it could also be used for 3D object manipulation [2]. At a generic level, such an algorithm is flexible enough to store *state-to-state* (location to location) transition rules and sequences of actions to carry them out. However, the map requires generation via an exploration process to identify, link and calibrate all visually distinct locations in the world. This poses a potential problem for cases where the space of reachable *states* is not directly explorable.

---

[2]Providing a computational model for time varying mental rotation tasks such as those described by Shepard and Metzler 1971.

4.2   Generalised Serial Problem Solver

The problem of planning and controlling physical actions using visual representations can be regarded as an example of *visual intelligence*. Examples of such systems already exist in the computer vision and robotics literature. In relation to our previous discussion on the architecture for an intelligent system, two issues need to be addressed in order to extend the map-builder approach to a system capable of a form of intelligence which we might recognise as a high level thought process. The first of these is relatively simple. In order to deal with an interrupt scheme (to support *conscious/ subconscious processes*) connections between *states* need to be coded with suitable interrupt levels, so that important tasks are allowed to complete without interruption. Secondly, in order to approach a more generic solution to *reasoning* and problem solving we need to be able to apply the fire-front scheme to problems where exploration in the *state* space is not directly possible. What is required is the equivalent of a mental exploration (ie: *imagination*). This step takes us away from the capabilities of existing systems and into a more speculative area.

Mental exploration would be possible if we could frame the problem as a *state to state* transition governed by a limited set of transformation instructions, for which we could learn to generalise the transition rules based on a limited set of example data. The nearest algorithmic candidate to solve this problem in the current literature is an artificial neural network trained as a *state* transition predictor (Figure 3). The *state* map could then be constructed in a thought experiment in which a range of map locations were hypothesised and linked. Solution of the problem could then proceed as for the visual path planning case. Clearly, any tested *manipulation* could also be used to provide new data to improve generalisation and regenerate the map, until a valid solution were found.
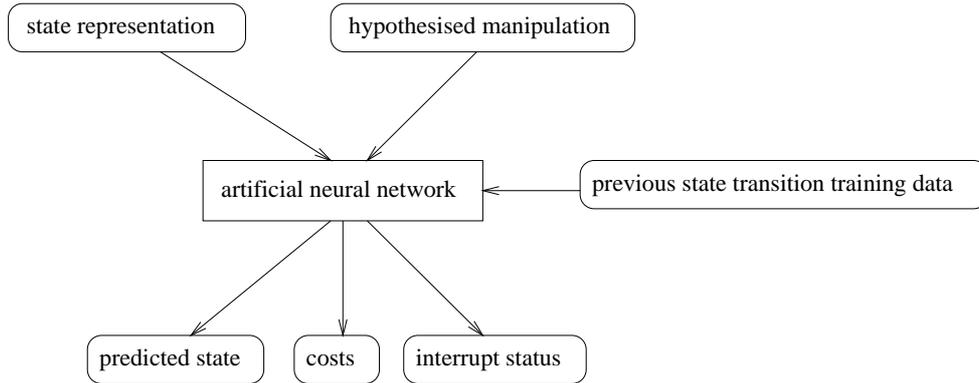

Figure 3: State Transition Predictor.

The process would look like the following:

1. Use the result of previous attempted *manipulations* to train a *state* transition predictor.

2. Use the *state* transition predictor to grow new *states* from the current set, comparing them with known locations to ensure that they are distinguishable and storing information such as costs and interrupt levels.

3. Iterate the process of *state* generation until a target location is reached with some required cost.

4. Having generated a map, retrace the minimum cost solution as in the serial problem solver algorithm.

Notice that this process fits directly our earlier definition of *thinking* as well as *reasoning*. If all of this can be achieved using a *representation* which can be closely related to sensory input, tasks such as robotic manipulation could then be achieved in a standard feed-back control loop.

A nice feature of such a solution is that the *state* map extends naturally with extra nodes should new control mechanisms become available to reach previously unattainable *states*. The mechanism for learning generalisations is actually unimportant and does not even need to work particularly well as eventually the hypothesis and test process could be forced to search every available route. Though clearly the sooner a

reasonably good generalisation is achieved, the sooner the *state* map would compute a valid minimum cost path. This observation has implications for the way that the mental map is extended towards the required solution and would be controlled by a series of rules generally referred to in the AI literature as heuristics and in this case may be as simple as comparing the current and required *state*. For a previously unseen problem we can expect to have no valid heuristics so for now it is convenient to ignore this issue until more is known about how they could be practically integrated into the working system. The important problem is then how accurately the *state* transition predictor can achieve its task. This can be simply stated as the requirement to have the best possible generalisation to unseen *states* from the available data. The solution to this problem must be expected to make use of a theoretical understanding of quantitative (ie: frequentist) probability. Practical examples of such problems are nicely illustrated in conventional IQ tests. In the statistical literature this is often referred to as the "model selection problem", while in the area of artificial neural networks it is called the "bias-variance dilemma". Inappropriate generation of a *state* which does not come from the set of achievable *states* would invalidate our initial definitions for an intelligent process, effectively reducing the system to one which could only solve problems by guesswork. Thus we see that this could well be the main problem in building a learning/reasoning (intelligent) system.

Having completed the planning process, the most important information which should be stored for the future would be the minimum set of *manipulations* and *state* transition predictors required to solve that category of problem. There may also need to be modifications to the heuristics governing the map generation process. Frequently recurring sequences might be stored as additional *manipulations* and suitably trained *state* transition predictors, to reduce the number of individual steps required in future planning processes and facilitating hierarchal problem solving. Indeed, the construction of hierarchal maps with nested solutions is the only identifiable process within this framework for support of high level reasoning capabilities. The full solution would need storing only for a frequently executed task, and in that case may be considered as placing it in memory for automatic (*sub-conscious (2)*) execution. In many cases the deterministic regeneration of the solution should not be regarded as a large overhead.

4.3   Practicalities of Constructing a Working System

The focus of this work should be in the construction of a generalised serial problem solver for visual tasks. The aim being to construct a system which can take a start and target *states* and identify a serial list of actions required to link them. The method should be designed to work with noisy *representations*. Clearly, not all information is required in order to represent the salient information in an image. In fact the only information we are interested in is that data which causes a change in *representation* subject to a *manipulation*. It therefore seems natural to focus the attention of a vision system on the aspects of the *representation* that change, ie: image differences. In addition, initial exploration of the problem space will need to be achieved in a controlled manner and this may well require the ability to undo the effects of individual *manipulations*. This mechanism will certainly be required in order to avoid dead-end solutions. Investigation of all of the issues involved will require some experience with example problems. Starting with simple visual problem solving tasks and simple control options would probably be a useful way of investigating the important characteristics of the approach with minimum additional complexity overhead. Before work can proceed further a set of example problems with allowable *manipulations* and ways of representing the data need to be identified.

We have said that the mechanisms for *state* storage and manipulation have to be statistical and while this is true it does not narrow the possibilities very much. In this circumstance it is necessary to go back to the brain itself for inspiration. Assuming that we may have a chance of building a system if we use the data processing mechanisms available in the brain then the statistical methods we will need are those derived by encoding *representations* as frequencies (ie: axonal pulses). It is no accident that our previous work in this area on the CLAM architecture provides us with exactly the set of statistical manipulations we need to construct mental exploration maps. Though CLAM cannot be used for the *state* transition prediction (it is simply an associative memory and therefore no generalisation capability) it can be used to generate and store the map. The concept of storing all statistically distinguishable patterns fits nicely with the subsequent use of the map in a feedback control system, as the stored patterns would guarantee the ability to follow a control path at the limit of the information available. With some minor modification it could also store *manipulations* and interrupt levels. Some thought is needed as to whether the ability to separate supervised from un-supervised learning can be utilised, or the ability to classify sets of *representations*.

Our work on the Bhattacharyya measure and B-Fitting addresses directly the issue of model selection. Specifically, a quantitative prediction of the model is made in the form of a probability distribution, which is then compared to distribution for example data. Clearly more work is still needed to construct the required network training algorithm, but the principle has been shown to be capable of selecting the most appropriate model for predictive tasks. As simpler models generally have fewer parameters and predict more compact distributions, this approach can be considered as a statistical version of Occam's Razor. In order to make the statistical assumptions behind this approach valid (uniform errors on target values) the *state* predictor simply needs to be trained on the square root of the *representation* vector. In addition our work on Pairwise Geometric Histograms (PGHs) provides a mechanism for representing shape in the required manner. Use of this representation alone would allow immediate control of the out of plane rotation and imaged size of an object. I would also suggest that we develop a representation for 3D representation based on disparity with the same statistical characteristics.

As we are not expecting to make explicit use of logical constructs (problems are to be solved using probabilities rather than simple logic) so we will not benefit from specialist AI languages (PROLOG). The amount of vector algebra is also unimportant so languages which simplify the syntax for linear algebra will not be of much value either (MATLAB or AIDA). The programming challenge involves the flexible and efficient *manipulation* of dynamically generated resources which may also require local storage for future use. This is best approached using pointer based algorithms such as supported in the C language rather than LISP (which would not cope well with the numerical aspects of the problem). This requirement also clashes directly with the standard use of C++ which would be likely to hinder both flexibility, speed, parallelisation and debugging. I can also foresee no added benefit from an overtly object oriented approach at this time. Efficient and flexible use of the host operating system is also likely to be of great value in the construction of a full system, in particular direct access to interrupt mechanisms and hardware. This would be facilitated by the use of an open architecture such as Linux. These comments are of course in keeping with our experiences of the development of the TINA machine vision system. Much of this work could be done most easily in simulation which will require the reliable integration of the POV ray software used on the map-building project.

5   Conclusions

The main conclusions of this document should probably be taken as follows. Artificial intelligence according to the common meaning of the phrase can probably never be conclusively demonstrated (at least to the most skeptical) due to the lack of clarity of definition. Other definitions are therefore necessary, though that proposed by Alan Turing could be said to be of little practical value, even if this has been the primary motivation for much work (eg: sub-sumption, PARRY, ELIZA).

Taking the opportunity to define our own terms we can specify an engineering solution for intelligence which may be not only achievable but useful. Practical use of the solutions in a real environment will probably require interrupt schemes which could be said to parallel the role of the sub-conscious. The architecture we suggest for this task we have called MIA (Minimum Intelligence Architecture). Notice that in contrast to many common theories of intelligence this approach does not necessarily involve a primary role for language, though we have not said that language applications are excluded. It therefore does not provide an immediate explanation for the internal narrative that many are often aware of being generated during their daily lives. Rather it illustrates that many intelligent actions can be developed without this process. Much of the architecture of the system can already be said to be relatively familiar to us. The system would appear to require a software approach which is consistent with our existing work in TINA and therefore suggests a natural extension.

Key in the design of an intelligent system would be a general serial problem solver. We expect that this would be used to evaluate expressions of the form;

$$Q \; = \; \sum_j C(S_i|M_j|S_k)$$

The sequence of *state* transition terms $(S_i|M_j|S_k)$ in this expression are directly comparable to Turing's ideas of the universal processor. This mathematical description could be taken as a computational definition for artificial intelligence, and this (rather than mimicking human behaviour) may even have been Turing's original intention. The analysis presented in this document differs from the idea of a

universal processor in one main respect, unlike a chess playing program for example, the various *states* are not easily specified, but must be generated from a continuum. These *states* and the associated *manipulations* must also be learned and stored in a way which permits feedback control in the presence of sensor noise and representational uncertainty. Solutions to artificial intelligence have no general utility in the real world without these additional requirements. This observation explains why , PROLOG, first order predicate logic and expert systems were not going to provide an answer to "What is Intelligence?". The area of 'autonomiuse agents' will also fail to generate intelligent systems unless attention is paid to these issues.

In this document we have explained how the idea of 'state' maps can be used for a variety of tasks of increasing complexity, clustering, planning and ultimately high level reasoning. The increasing sophistication of these problems may be a clue to the evolutionary development of high level processes in the human brain. Notice, this approach is not based upon a logical analysis of a task, as exemplified by formal logic. Indeed, in the inevitable presence of contradictory data found in a learning system, humans seem to use logic only to construct an argument from the subset of data which supports a favoured hypothesis, perhaps to justify an action or behaviour to another person. Here, an approach has been suggested for how a serial problem solver may be constructed and its required characteristics. Fundamental research issues which need to be addressed, in terms of building a working system with recognisable aspects of high level intelligence, are those of *representation* and optimal generalisation from restricted data (ie: model selection). These are seen as challenging but not necessarily insurmountable in relatively short research timescales. Our one guide is that the process seems to be computable using a parallel network of sparsely connected units which encode data as frequencies (ie: histograms). This raises the question of why such systems have not already been constructed. It could be that solutions require a massive set of analog temporal processors. Another observation is that there are sufficient distractions in the associated research areas of computer vision and artificial intelligence to provide a lack of coherent will (or focus) to address such issues. When such a focus is provided, either by necessity or funding, we see no reason why such systems could not be simulated. Construction of a parallel architecture which realises the intrinsic potential of these ideas may prove to be more of a challenge.

References

[1] M. Boden, Artificial Intelligence and Natural Man., Oxford University Book Set, Harvester press Ltd., ISBN 0-85527-435-2, 1977.

[2] P.A.Bromiley, N.A.Thacker and P.Courtney, Colour Image Segmentation by Non-Parametric Density Estimation in Colour Space. Proc. BMVC 2001, 283-292, 2001.

[3] D.C.Dennet, Consciousness Explained, Penguin Books, 1993.

[4] M. Ginsberg, Essentials of Artifical Intelligence, Morgan Kaufmann Pub., 1993.

[5] A.J.Harris, N.A.Thacker and A.J.Lacey, Computer Vision Algorithms for Autonomous Robot Map Building and Path Planning, proc. IEEE Intel. Trans. Sys. Conf, ITSC'97, Nov 1997.

[6] L. Johnson, E.T. Keravnou, Expert Systems Technology, Abacus Press, 1985.

[7] D. McNeill, P.Freiberger, Fuzzy Logic, Touchstone, 1994.

[8] N.A.Thacker and A.J.Harris, A Feature Representation for Map Building and Path Planning. proc. BMVC, 800- 809, Southampton, 1998.