From: AAAI-94 Proceedings. Copyright © 1994, AAAI (www.aaai.org). All rights reserved.

Results on Controlling Action with Projective Visualization

Marc Goodman*

Cognitive Systems, Inc.
234 Church Street
New Haven, CT 06510

Computer Science Department
Brandeis University
Waltham, MA 02254

Abstract

A projective visualizer learns to simulate events in the external world through observation of the world. These simulations are used to evaluate potential actions on the basis of their probable outcomes. Results are given that indicate, 1). the error rate for projective visualization is sub-linear as the system projects farther into the future, 2). the error rate is inversely proportional to the number of cases, 3). a simple domain model can be used to reduce the effect of compounding error, and 4). projection can be used to increase the performance of an agent, even when this projection is imperfect.

Introduction

Projective visualization is a technique for controlling action. A system built with projective visualization learns to project situations into the future based on past experience. Potential actions are selected and rejected based on an evaluation of the projected state. Previous work (Goodman 1993) presents the basic model for projective visualization and results indicating that a limited form of projection can be used to build a system that performs as well as a reactive system, based on observation of the reactive system. It also offers preliminary results on the accuracy of projection for increasing windows of projection.

Projective visualization starts with a concrete representation of an observed process or activity. These observations are organized as a set of temporally linked cases. The system induces a large number of decision trees, using an algorithm akin to ID3 (Quinlan 1986), CART (Brieman *et al.* 1984), or Automatic Interaction Detection (Hartigan 1975). Each decision tree is responsible for projecting an individual feature of a situation one step forward into the future. During projection, a new, hypothetical case is created using these decision trees, and this hypothetical case serves as the basis for further projection.

*Thanks to David Waltz and Richard Alterman for useful discussion. This work was supported in part by ARPA under contract no. DAAH01-92-C-R376.

Other work on the direct application of previous experience to prediction has been presented (Rissland & Ashley 1988; Klein, Whitaker, & King 1988). Both of these works use a very restricted notion of prediction, however. It is assumed that there is only one value to be predicted and that that value is directly associated with the retrieved case (i.e. there is no notion of a projection window, or of projecting an entire situation. Rather, prediction in this sense is akin to a classification task). Prediction in Battle Planning has also been treated as a classification task (Goodman 1989).

The issues addressed in this paper are closely related to issues of simulation. Simulation has been framed in terms of a qualitative model of a domain (Kuipers 1986). The SIMGEN system (Forbus & Falkenhainer 1990) uses both qualitative and quantitative information to drive simulation of a physical systems, and introduces the notion of self-explanatory simulations. Both of these techniques require significant knowledge of the domain, in the form of qualitative relationships. One of the issues addressed by this paper is to what extent such models can be used to guide the induction process.

This paper extends previous results (Goodman 1993). This paper presents details on the inductive algorithm used to generate the projectors used by the system and evaluates their performance. This paper also compares the accuracy of projection with and without *interpretations*, and with and without a simple domain model to guide induction. Finally, the effect of projecting farther and farther into the future on the ability of the system to act is evaluated.

The Target Domain

The target domain discussed in this paper is a simulation of personal combat between two gladiators armed with axes and shields, based on the video game *The Bilestoad.* The perceptual system of the agent in the combat simulation includes a visual system, a pain sense, a kinesthetic sense, a haptic sense, a sense of joint stress, and awareness of motor control signals which control the agent's body.

The personal combat simulation is a good domain for evaluating projection because: 1). a large amount of data can be automatically created to test the limits of the system, 2). the domain is inherently real-time, focusing on very small changes in the state of the world at each time step, 3). the system must project relatively large distances into the future to evaluate the effectiveness of its actions, and 4). there are relatively large number of different actions that the agent can take at each step, which allows this work to examine both the accuracy of projection as well as the effects of projection on performance.

How Projective Visualization Works

Projective visualization manipulates a representation of a process or activity called a *case*. A case is defined as a snapshot of the state of the world, along with temporal links to previous and next cases. In effect, a case is like a single frame of a motion picture. In the personal combat domain, cases are gathered every $\frac{1}{6}$th of a second. Throughout this work, this paper will use the following terms:

Field: A case is a collection of fields, each field defining a particular value in the case. For example, in the personal combat domain, there is a field for the amount of damage to the left elbow of the agent, a field for the angle of the agent's left elbow, and so on. **Match fields** are used to build a set of inductive indices for projecting individual features of a situation. Different sets of indices may have different match fields. **Outcome fields** are used as a classifications or predictions by the system. Different sets of indices will have different sets of outcome fields. For example, one set of indices may be useful for projecting the angle of the agent's elbow, while another set of indices is useful for projecting whether the agent will continue walking.

Field value: A field value of a case is the value of a particular field for that case. For example, the field value for the angle between the agent's head and its opponent's axe might be 30°.

Feature projector: A feature projector is a set of indices that is responsible for predicting the value of one or more outcome fields.

Projector: A projector is a collection of feature projectors that can be used to predict an entire future state of the combat simulation from current and previous states of the combat simulation.

Building and using the personal combat projector consists of two separate phases. In the first phase, individual feature projectors are built off-line. In the second phase, the resulting projector is used to rapidly predict future states of the combat simulation. There is no reason, in principle, why learning and action could not be integrated. However, the computation resources required to perform learning and the real-time requirements of action in the domain have mandated this separation.

How the Projector is Built

A projector consists of a large number of feature projectors, each of which is an inductively formed discrimination tree that indexes a set of cases. The algorithm for generating these indices, called **PVClus**, is summarized below:

1. The Spearman's Rho correlation between pairs of fields is used to cluster these fields into groups that have high correlation using a Leader algorithm (a variation of the K-means clustering algorithm (Hartigan 1975)). Each group of correlated fields will serve as a set of outcome fields for an individual feature projector. The reason for grouping such fields is that highly correlated fields may have common underlying factors that govern their behavior.
2. The set of fields describing the combat simulation is enhanced by adding interpretations that capture temporal and other relationships between fields and values. For example, the system creates new fields that capture the first and second discrete derivatives of the existing fields.
3. Each feature projector with associated outcome fields is used to build a discrimination tree. The leaves of this tree are sets of cases with similar values for their outcome fields. The internal nodes of the tree are binary decisions. For example, an internal node might check whether the agent's shield is between the agent and its opponent's axe. Such an internal node might separate cases where the agent was protected from damage from other situations. These discrimination trees are built as follows:
   (a) The set of field values associated with the outcome fields for the feature projector, the discrete first derivatives of these outcome fields, and the discrete second derivatives of these outcome fields are compared. The set of fields that minimizes the following equation is selected:

$$Error = \sum_{i=0}^{n} \left(\frac{|f_i|}{|f|}\right)^2$$

   Where each f_i is the cardinality of the subset of cases that all have the ith field value for their outcome field, and f is the cardinality of the set of cases as a whole. This measure gives an indication of the *a priori* error rate if the system were to use this set of fields as outcome fields. A more accurate measure of such error would be:

$$Error = \sum_{i=0}^{n}\sum_{j=0}^{n} \frac{FieldValue(c_i) - FieldValue(c_j)}{n^2}$$

   Where c_i represents the ith case in the set. There are two drawbacks to this form, however. First, it takes $O(n^2)$ time to compute whereas the first form is $O(n)$, and second, it is only appropriate for comparing numerical values. The set of fields with minimum *a priori* error are used as the outcome fields for the discrimination tree.

The purpose of this computation is to try and reduce the complexity of the decision tree needed to accurately project a value. For example, consider a rocket tracking domain. Before firing, the rocket is at rest and the easiest way to predict its location at the next time step is to use its current location. When the rocket begins firing, its acceleration will be constant and easy to predict, whereas its velocity and location start changing and become more difficult to predict. After the rocket engine cuts out, its X location will change, whereas its velocity will be constant (barring air resistance). Meanwhile, its Y location and velocity will change, but its acceleration will still be constant due to gravity. A natural approach to solving this problem is to find some way for a given cluster of cases to predict which of these representations for the future state of the rocket is easiest to predict and to use this as the outcome for building the set of indices inductively.

(b) The algorithm gathers all the fields that describe the previous state of the combat simulation, and associated interpretations. For each such field, given a set of associated cases, all field values for that field are tested as possible discriminators. Each discrimination divides the cases into two sets, those whose field values are less than or equal to the candidate field value, and those whose field values are greater than the candidate field value. The field value for a particular field that accounts for the largest degree of variance in the outcome fields for that feature projector is found.

(c) The winning field value is tested for statistical significance using the Mann-Whitney U Test. If the field value passes this test, the associated field is saved as a match field. If not, the field is ignored. The Mann-Whitney U Test is an appropriate test for statistical significance because it is non-parametric, and, in general, the system cannot make assumptions about the distribution of values in particular fields.

(d) The best field value overall is used as an initial discrimination. The discrimination, A, is enhanced as follows:

   i. For each match field and its associated field values, each field value is used to form a candidate discrimination, B.

   ii. Boolean conjunctions of features $A \& B$, $\bar{A} \& B$, $A \& \bar{B}$ and $\bar{A} \& \bar{B}$ are tested. The conjunction of features that best accounts for variance in the outcome fields of the feature projector is selected. If the variance accounted for by the conjunction is greater than the variance accounted for by A, then the addition of B is checked for statistical significance using the Mann-Whitney U Test. If adding B passes this test for statistical significance, then A is set to the conjunction of features, and the process repeats at item 3(d)i. If adding B fails the test for statistical significance, then the next best candidate discriminator is tried. If no discriminator passes the test for statistical significance, or no discriminator improves the variance accounted for by A, then A is used as the basis for a parent node in the tree. This technique is essentially hill climbing to improve the discriminations in the tree. The chief advantage of such a technique is that it conserves data. With all such discrimination tree learning algorithms, the more examples that are available at any given point, the more accurate is the credit assignment on individual features. By grinding as much information out of each discrimination in the tree, the system ends up with a short tree with relatively few leaves, measures that have been argued are particularly important (Fayyad & Irani 1990).

(e) The set of cases are partitioned based on the internal node. The process repeats for each subset of cases at item 3a. The algorithm terminates when no new, statistically significant discriminations can be added that improve the variance in outcome fields.

How the Projector is Used

At run-time, a case is created that describes the current state of the combat simulation. For each feature projector, this case is used to traverse the associated discrimination tree of the feature projector. The mean values for the outcome field values in the leaf cluster are used to set the corresponding fields in a fresh case buffer. When each feature projector has been traversed, the system has a hypothetical case that describes the predicted state of the combat simulation. This predicted state can then serve as the basis for further projection.

Traversing each discrimination tree takes roughly $O(\log n)$ where n is the number of cases. The total time complexity for projection is therefore $O(pm \log n)$ where p is the projection window, or how far into the future the situation is projected, m is the number of feature projectors (which is effectively constant for any given system), and n is the number of cases. It is, therefore, possible to do projection very quickly.

Learning with Domain Knowledge

There are two primary sources of domain knowledge available to the system. The first source of domain knowledge is a set of *interpretations* that can be derived from the raw case representation. For example, in the personal combat domain, the system may automatically create representations of the spatial relationships between objects in the world (such as the joints of the bodies of the agent and its opponent)

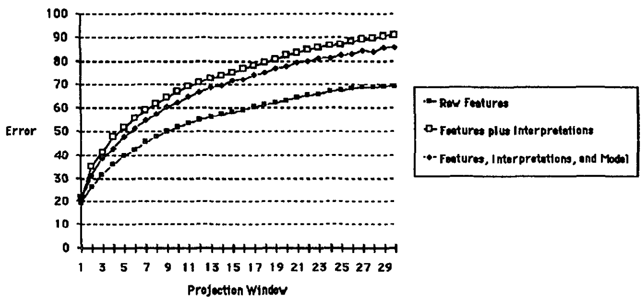

Figure 1: Error VS. Projection Window for Systems with and without Domain Knowledge

based on Cartesian, deictic, and landmark reference systems (Miller & Johnson-Laird 1976). Automatically deriving new representations is similar to constructive induction (Callan & Utgoff 1991). The effect of these new, derived representations is two-fold. On the positive side, it reduces the *hypothesis space bias*, defined as being a component of inductive bias that defines the space of hypotheses that are being searched (Buntine 1990). On the negative side, since the number of possible discriminations is drastically increased, the chance of beta error (accepting a discrimination as statistically significant because the confidence interval is too loose) is also increased. This can actually decrease the overall performance of the system (Almuallim & Dietterich 1991).

The second source of domain knowledge is a simple model of qualitative influences between fields (or *qualitative model* (Kuipers 1986)). The system uses such a model in conjunction with the inductive algorithm described in Section as follows: 1). the system determines the set of fields that are directly relevant to the outcome field using the qualitative model. 2). This subset of fields is used as the set of initial match fields by the inductive algorithm. The inductive algorithm terminates when it is unable to add any more discriminations that are statistically significant, based on these match fields. 3). The system determines the set of fields that are directly relevant to the previous set of match fields. Another pass of induction is performed. 4). The system continues to expand the set of match fields and perform additional induction phases until no new match fields can be added. At that point, a final pass of induction is performed with all available match fields.

Hence, the model is used to create an *application-specific bias* (Buntine 1990). There is some similarity between this technique and the use of primary and secondary features for indexing cases in Explanation Based Indexing (Barletta & Mark 1988).

Results

Figure 1 shows the effect of the projection window (or how far the system projects into the future) on the error rate for projection. The data points in the chart were generated as follows:

- Three projectors were built using 16K cases from observation of two reactive agents battling each other in the combat simulation. The trend labeled "Raw Features" was built with only raw fields (164 different fields) and first and second discrete derivatives used as match fields (for a total of around 500 fields). The trend labeled "Features plus Interpretations" was built using interpretations as well as raw case fields (for a total of around 2000 fields). The trend labeled "Features, Interpretations, and Model" was built using both interpretations and a simple qualitative model of the domain.
- One at a time, each projector was integrated with the combat simulation. Whenever a combat situation was detected, the system would begin projecting. These projections were saved. At the next time step, the actual state of the world was compared to the previously projected states of the world. For each of the 164 fields describing the current situation, the sum of the absolute value of the difference between the actual and projected situations was calculated. For each of the three projectors, 1,000 simulations were observed, amounting to approximately 50,000 comparisons for each graphed point.
- The maximum mean error for each field over the entire projection window was found. The corresponding field values for each of the error logs was then normalized by dividing the actual error over the maximum error. Hence, all errors for each field were normalized to the interval [0..1]. This was done so that the composite error for the projector overall would not be dominated by individual fields where the range of projection was much larger than other fields. For example, the value of a distance field could be anywhere from 0 to 4000, whereas the value of a pain field might only be in the range of 0 to 4.
- the datapoints in the trends are the sum of the normalized error rates for all of the fields in the projector. Hence, the maximum possible error rate is 164.

Note that the error rate forms a kneed-over curve with respect to the projection window. This is because as the system projects farther and farther, more and more of its projections become no better than a guess based on the *a priori* distribution of values in the set of cases. As more and more of these projections "top off," the sum of their errors becomes a kneed-over curve.

As shown in Figure 1, the projector built without interpretation or model was the most accurate over all. This result is in keeping with other work (Almuallim & Dietterich 1991). However, as shown in Figure 2, projectors built with interpretations were more accurate for complex or difficult to explain features. Figure 2 shows the results of projecting the angle of movement

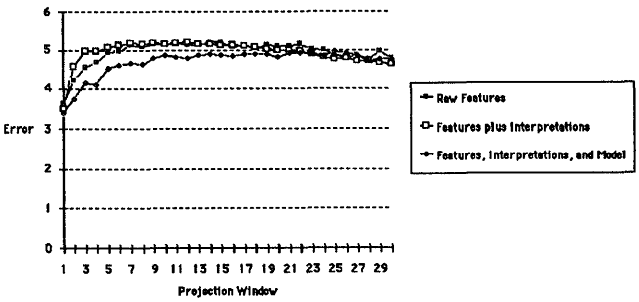

Figure 2: Error VS. Projection Window on a Complex Concept for Systems with and without Domain Knowledge

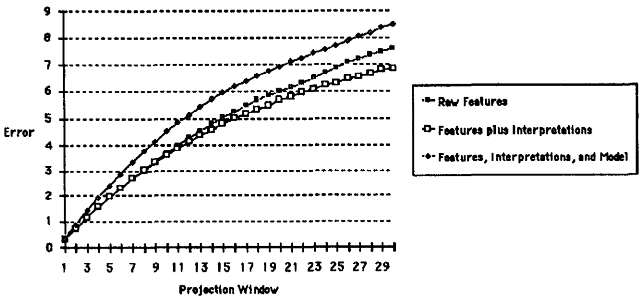

Figure 3: Error VS. Projection Window on a Complex Concept with Inadequate Model

of the agent, opponent, and devices in the simulation (a total of 18 different features). These features are particularly difficult to project accurately, because of interactions between the bodies of the agent and opponent (which may obstruct each other in a variety of complicated ways). Explaining these fields adequately requires the system to make use of higher-level interpretations. Also note that due to the difficulty in projecting this field accurately, the error rate tops off rather quickly.

In both Figure 1 and Figure 2, the accuracy of the projector built with a simple model was greater than the performance of the system with interpretation but without a model. This is because the model helps the system to reduce the effects of compounding errors. Such a situation isn't, however, guaranteed. Figure 3 shows the accuracy of the three projectors on how much pain the agent and opponent will be feeling in the future. Such a projection is very difficult to model directly, because it requires the model to explain when and under what circumstances the axe will come into contact with the agent's body in the future. A complete model would involve virtually every field of the situation description. In this case, allowing the system

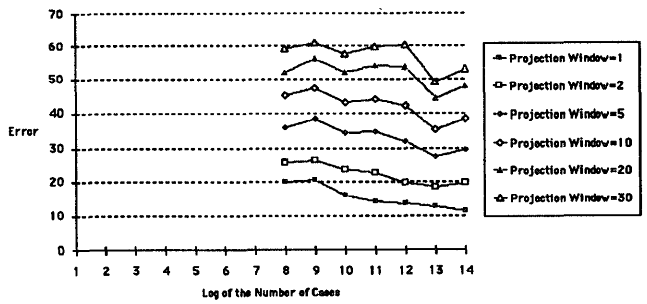

Figure 4: Error VS. Size of Training Set for Different Projection Windows

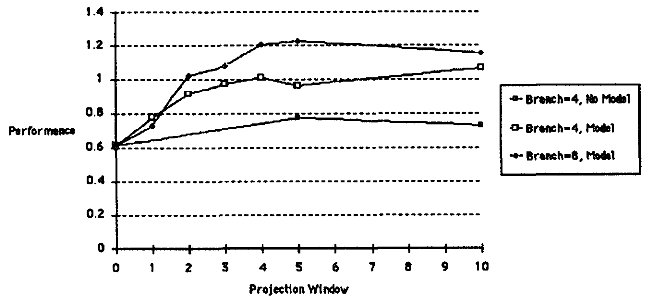

Figure 5: Performance of the Agent VS. Projection Window

to select its discriminations purely on a statistical basis is more effective than guiding induction with a model.

Figure 4 shows the effect of training set size on error rate for different projection windows. Separate projectors were built with 256, 512, 1024, ..., 16384 cases (these projectors were built with interpretations, but without a model). Error rates for each projector were generated as above. For small projection windows, the error rate falls roughly linearly in relation to the log of the number of cases in the training set. On the other hand, for larger projection windows, the effects of compounding error in projection (which can be considered a form of noise) grow to dominate the accuracy of the system. This supports the importance of reducing the effect of compounding error by using a model of the domain.

Figure 5 shows the effect of projection on one facet of the performance of the agent, namely how much damage the agent is able to inflict on its opponent in each game. These trends were generated as follows:

- The *branching factor* defines how many randomly generated patterns of control signals were evaluated (see (Goodman 1993) for a discussion of generating more accurate control signals using *action generators*). So, for a branch factor of 4, four different patterns of control signals are randomly generated, evaluated, and selected from.

- For each pattern of control signals, the system projects forward k steps.
- An evaluation of the projected situation k steps in the future is performed. For these tests, the simple difference between the sum of the damage to different parts of the agents body and the sum of the damage to the opponent's body was computed.
- The pattern of control signals with the best evaluation was determined and executed.
- At the end of the simulation, the total amount of damage to the agent's and opponent's bodies were logged. The mean values of the damage to the opponent's body are shown in the graph.

Three trends were generated. The first trend, "branch=4, no model" was generated using a projector without either a model or interpretations. The second trend, "branch=4, model" was generated using a projector built with both interpretations and a simple qualitative model, using a branch factor of 4. The third trend, "branch=8, model" was generated using the same projector as the previous trend, but with a branch factor of 8.

Note that a projection window of 0 will result in the system evaluating all patterns of control signals as being identically effective. Hence, with a projection window of 0, the system is acting completely randomly. The significant result shown in this graph is that projection does, in fact, help the system to perform more effectively, even when this projection is imperfect. Also note that the performance of the system continues to increase as the system projects farther into the future, until around 5 steps into the future where the performance levels off. This increase in performance is in spite of the compounding error shown in Figure 1.

The performance of the system with a projector that was built using both interpretations and a model was significantly higher than the performance of the system without such domain knowledge. This may seem surprising, given Figure 1 which shows that the projector without domain knowledge was more accurate overall. However, most of the fields that are most directly relevant to controlling action, such as projecting damage, rely heavily on such domain knowledge.

Figure 5 also shows that the performance of the system can be significantly increased by increasing the branching factor. Such a result, while not surprising, is good support for the usefulness of projection in selecting among actions. It also reaffirms the need for high-quality action generation (Goodman 1993).

Conclusions

This paper has demonstrated that it is possible to build systems that can project situations into the future using previous experience. It has also demonstrated that the usefulness of such projections may be increased through the judicious use of domain knowledge (specifically interpretations of raw data and simple qualitative models). Finally, it has shown that projection, even imperfect projection, can be used to improve the performance of an agent taking action in the world.

References

Almuallim, H., and Dietterich, T. 1991. Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence.*

Barletta, R., and Mark, W. 1988. Explanation based indexing of cases. In *Proceedings of the First DARPA Workshop on Case-Based Reasoning.*

Brieman, L.; Friedman, J.; Olshen, R.; and Stone, C. 1984. *Classification and Regression Trees.* Wadsworth.

Buntine, W. 1990. Myths and legends in learning classification rules. In *Proceedings of the Eighth National Conference on Artificial Intelligence.*

Callan, J., and Utgoff, P. 1991. Constructive induction on domain information. In *Proceedings of the Ninth National Conference on Artificial Intelligence.*

Fayyad, U., and Irani, K. 1990. What should be minimized in a decision tree? In *Proceedings of the Eighth National Conference on Artificial Intelligence.*

Forbus, K. D., and Falkenhainer, B. 1990. Self-Explanatory Simulations: An integration of qualitative and quantitative knowledge. In *Proceedings of the 1990 American Association for Artificial Intelligence*, 380–387. Lawrence Erlbaum Associates.

Goodman, M. 1989. CBR In Battle Planning. In *Proceedings the Second DARPA Workshop on Case Based Reasoning*, 312–326.

Goodman, M. 1991. A Case-Based, Inductive Architecture for Natural Language Processing. In *AAAI Spring Symposium on Machine Learning of Natural Language and Ontology.*

Goodman, M. 1993. Projective visualization: Learning to act from experience. In *Proceedings of the Eleventh National Conference on Artificial Intelligence.*

Hartigan, J. 1975. *Clustering Algorithms.* John Wiley and Sons.

Klein, G.; Whitaker, L.; and King, J. 1988. Using Analogues to Predict and Plan. In *Proceedings of the First DARPA Workshop on Case-Based Reasoning*, 250–259.

Kuipers, B. 1986. Qualitative simulation. *Artificial Intelligence* 29:289–338.

Miller, G. A., and Johnson-Laird, P. N. 1976. *Language and Perception.* Cambridge, Massachusetts: The Belknap Press of Harvard University Press.

Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning* 1:81–106.

Rissland, E., and Ashley, K. 1988. Credit Assignment and the Problem of Competing Factors in Case-Based Reasoning. In *Proceedings of the First DARPA Workshop on Case-Based Reasoning*, 250–259.