[Steve Young]

Cognitive User Interfaces

[Intelligent human interaction—is there an app for that?]

PHOTO BY MATTHIEU BARRAGUÉ

This article argues that future generations of computer-based systems will need cognitive user interfaces to achieve sufficiently robust and intelligent human interaction. These cognitive user interfaces will be characterized by the ability to support inference and reasoning, planning under uncertainty, short-term adaptation, and long-term learning from experience. An appropriate engineering framework for such interfaces is provided by partially observable Markov decision processes (POMDPs) that integrate Bayesian belief tracking and reward-based reinforcement learning. The benefits of this approach are demonstrated by the example of a simple gesture-driven interface to an iPhone application. Furthermore, evidence is provided that humans appear to use similar mechanisms for planning under uncertainty.

A limiting factor of the POMDP framework is that exact computation is intractable, and, hence, POMDPs are often thought to be impractical for real-world problems. The second part of the article therefore attempts to demonstrate that the essential benefits of the POMDP approach can be retained by the use of sensible approximations. To illustrate this, two real-world spoken dialog systems (SDSs) are described. Each make very different approximations but both achieve significant performance gains. The first hidden information state (HIS) system represents an evolutionary path from current systems. The second Bayesian update of dialog state (BUDS) system exploits more recent developments in Bayesian networks, and, although more challenging to scale, it offers significantly more potential for short- and long-term adaptation. The article concludes by noting that while the challenges facing future development of cognitive user interfaces are considerable, the need to develop this style of interface appears to be inevitable.

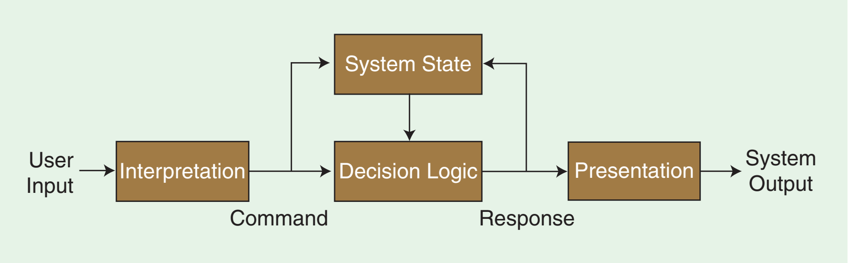
INTRODUCTION

As the complexity of computer-based systems continues to increase, there will be an increasing demand for much more robust and intelligent interaction than is possible with the current generation of human-computer interfaces. The technology drivers for this demand

Digital Object Identifier 10.1109/MSP.2010.935874

1053-5888/10/$26.00©2010IEEE

are already emerging. The introduction of highly capable touch-screen smartphones is spawning a new generation of highly sophisticated mobile applications in which complex interactions must be accomplished using combinations of gesture and voice. However, screens are small and ambient noise levels are often high. Hence, maintaining acceptable levels of robustness will be a significant challenge. The development of smart robots for use in mass markets such as aids for the elderly will further create new challenges. Here voice control will be essential but a wide range of issues such as inadequate signal source separation, unreliable voice activity detection, speech understanding errors, and user uncertainty and confusion will combine to ensure that accurate interpretation of the user's intentions will be very difficult. A further example is the rapidly expanding computer games industry which in the United States, now exceeds the film industry measured by turnover. New generations of fully immersive games that involve the human player engaging in realistic dialogues with computer-generated characters will push our ability to create robust and natural user interfaces to the limit. Less frivolous but equally challenging examples of similar technology will emerge in areas such as health-care support and education.



[FIG1] **Finite state automaton model of human-computer interaction. Each user input is regarded as a command that is used by decision logic to determine the transition from one state to the next. Entering each new state generates a response to the user.**

The premise of this article, which is based on a 2009 IEEE International Conference on Acoustics, Speech, and Signal Processing plenary talk, is that future human-computer interfaces will only be able to meet the above challenges if they exhibit the following four key characteristics:

1) *Ability to support reasoning and inference*. Natural human communication relies on imprecise analog signals such as gestures, facial expressions, and speech. The user interface must be capable of interpreting these inputs in context to robustly resolve ambiguities and minimize errors.

2) *Ability to plan under uncertainty*. Effective communication involves meeting specific goals with incomplete knowledge. This requires that communicative goals are defined objectively and strategies then optimized to meet the objectives as efficiently as possible.

3) *Ability to adapt online to changing circumstances*. Contexts and environments change and the user interface must be able to modify its behavior to maintain an acceptable level of performance.

4) *Ability to learn from experience*. In addition to short-term adaptation, a user interface should be capable of learning from its own interactions over the long term. The more it is used, the smarter it should become.

User interfaces that have these four essential properties will be referred to as cognitive user interfaces.

Virtually all existing human-computer interfaces follow the finite state automaton model outlined in Figure 1. All relevant information is encoded within a finite state machine. Each user input is regarded as a command, which is used by decision logic to determine the transition from one state to the next. Entering each new state generates a response to the user. This model applies equally from the simple button interface of a washing machine to the complex natural language interface of a speech-driven information inquiry system. The only difference is that the level of ambiguity in the latter is much higher since the spoken input signal will often be unrecognized. Nevertheless, both operate on the assumption that the complete state is known.

In fact, this assumption can never be satisfied. Even though the push of a button on a washing machine may be entirely unambiguous, it may not represent the intention of the human user. Humans are fallible, and they often operate with half-formed intentions, based on incomplete information. Thus, there will always be uncertainty in a human user's intentions and this uncertainty will increase as the IT systems that they interact with gain in complexity. Add into this mix an increasing reliance on imprecise multimodal inputs such as gestures, emotion detectors, gaze trackers, and speech, and the need for more robust mechanisms for handling uncertainty in future systems becomes overwhelming.

The claims presented in this article are straightforward. Uncertainty cannot be avoided and future interfaces will need to exhibit cognitive behavior if they are to be fit for a purpose. The sections that follow will argue that the framework of POMDPs underpinned by Bayesian inference and Bellman's optimality principle provide an appropriate engineering approach to building future cognitive user interfaces.

It should be noted that this is a position article and not a review article. Hence, references are given where appropriate but there is no attempt to be comprehensive.

AN EXAMPLE: A SIMPLE GESTURE-DRIVEN INTERFACE

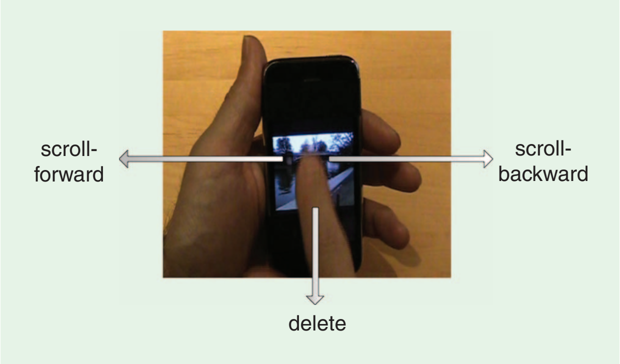
The iPhone is a great demonstration of how an intuitive gesture-driven interface can improve our ability to communicate with a device [1]. Nevertheless, some operations are not as slick as we might wish. For example, suppose you have taken a large number of photos and you want to quickly skim through them and delete all the ones that you do not wish to keep. The default interface requires you to select each photo, hit a delete button, and then explicitly confirm every operation. As illustrated in Figure 2, a quicker way to do this might use just three gestures to scroll forward, scroll backward, and delete. The only problem with such an interface is that when you try to do it quickly, your gestures become unreliable and errors occur. Of course, in a practical system, there would need to be some form of recovery mechanism to undo unintended deletions, but here the concern is with minimizing the occurrence of such errors in the first place.

THE ESSENTIAL BENEFITS OF THE POMDP APPROACH CAN BE RETAINED BY THE USE OF SENSIBLE APPROXIMATIONS.

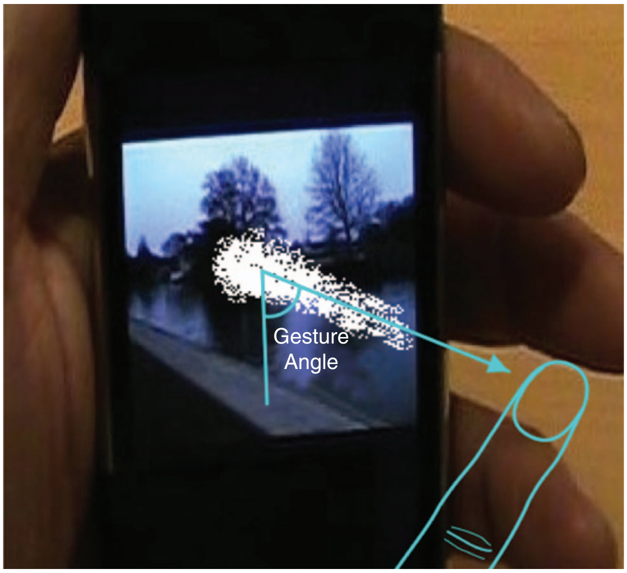
Consider first the classical approach to implementing this interface as exemplified by Figure 1, and suppose that each gesture is decoded simply on the basis of the angle it makes on the screen as shown in Figure 3. The classical approach entails a two-stage process: first the angle of each gesture is decoded into one of the three possible commands, and second, the decoded command forms the input to some decision logic that determines the response.

The first of these stages is a classification problem. If the angle that the gesture makes with the vertical is θ, then since errors can occur, the usual technique is to estimate a probability distribution $P(\theta|\omega)$ for each class $\omega = \{\text{forward, delete, backward }\}$ (see Figure 4). Optimal decision boundaries θ_i^* can then be determined based on the class posterior probabilities $P(\omega|\theta)$. For example, Figure 4 shows a typical distribution for the case where the average error is $\approx$20%. A typical choice for θ_i^* would then be to set equal posterior probabilities at the decision boundaries as shown by the vertical dotted lines in Figure 4. Each input gesture can then be decoded by simply comparing the angle with the decision boundaries and choosing an appropriate action. As a refinement, the probability of error rate can be estimated from the posterior distributions and a confidence margin δ determined. Thus, the second stage of the classical approach is typically a simple program or flowchart of the form outlined in Figure 5.

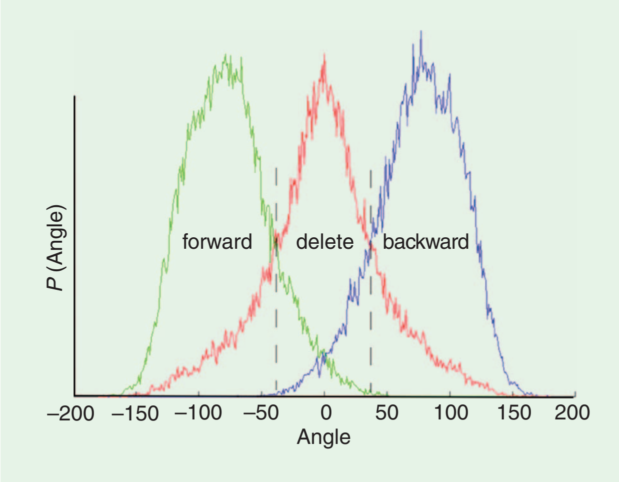
So what is missing in this approach to user interface design? First, there is no explicit modeling of uncertainty. As shown in the example, confidence margins can be used to inform the decision



[FIG2] iPhone photo selection using three gestures: a) swipe left to scroll forward through photo roll, b) swipe right to scroll backward, and c) swipe downward to delete the photo.



[FIG3] Decoding a gesture. Each gesture is defined by its angle with the vertical from which the intended command can be inferred.



[FIG4] Empirical distributions of gesture angles for each command. Given the angle of a gesture, the most likely command can be determined by comparing it with the class boundaries shown as dotted lines. The degree of overlap between distributions determines the error rate.

```
Let $\theta$ = angle of input gesture and
    $\theta_1$ and $\theta_2$ be lower and upper thresholds
Let $\delta$ be a confidence margin around
    each threshold
if $\theta < \theta_1 - \delta$ then
    scroll-forward
else if $\theta_1 + \delta < \theta < \theta_2 - \delta$ then
    delete-photo
else if $\theta_2 + \delta < \theta$ then
    scroll-backward
else
    do-nothing
end if
```

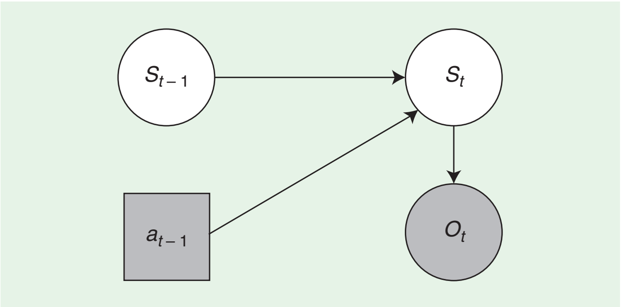
[FIG5] Decision logic for decoding each gesture. The effect of errors is reduced by introducing a margin of width 2δ centered on each decision boundary. When a gesture lies within this margin, the command is ignored.

process, but the decoding itself is still a hard decision and once taken, it cannot be easily undone. Second, there is no attempt to track what the user might be trying to achieve. Hence, the system has no way of determining whether or not its interpretation of the user's gesture is consistent with what the user might be trying to do. For example, in the case here, it may be observed that users rarely move backwards after deleting a photo, but when moving backwards the most likely next action is to delete a photo. This behavioral information can be useful when disambiguating 'noisy' gestures. Third, since there are no quantifiable objectives, there is no basis for optimizing the decision rules embodied in the flowchart. As consequence of all of this, the criteria required for a cognitive user interface cannot be met.

The key to building a cognitive user interface is to acknowledge the uncertainty in interpreting input gestures and therefore to view them not as commands, but rather as observations from which the system can infer the user's intent. The effectiveness of the system in responding to the user's intent is then quantified by a set of rewards and the necessary decision logic can be optimized by maximizing these rewards. The implementation of this approach depends on two fundamental ideas: Bayesian inference and Bellman's optimality principal, which leads to a framework that is commonly referred to as a POMDP [2], [3].

Returning to the iPhone example, the user has one of three possible intentions at each time step: move forward, move backward, or delete the current photo. Each of these is represented by a discrete state $s = \{\text{forward, delete, backward}\}$. Notice here that it is the state of the user that is being modeled, not the state of the machine. To satisfy the user, the machine has four possible actions available to it: $a = \{\text{scroll-forward, delete-photo, scroll-backward, do-nothing}\}$.

The user's intention s_t at time t will depend on their previous intention s_{t-1} and the previous system action a_{t-1}. Thus, the dynamics of the user's behavior can be captured in the transition probability $P(s_t|s_{t-1}, a_{t-1})$. The resulting gesture o_t at time t will then depend only on the corresponding user state s_t, and hence, the variability in the user's expression of intent is captured by the observation probability density $p(o_t|s_t)$. Notice that the observation is simply the measured angle of the gesture; there is no attempt to classify the gesture as in the previous classical decode and execute approach.



[FIG6] The Bayesian influence network showing one time step for the example gesture interface. The system state *s* is hidden as indicated by the open circles. The observation *o* and the action *a* are both observed as indicated by the shading.

The key problem of course is that the intent of the user cannot be directly observed. Hence, it is a hidden variable, and its value can only be inferred from knowledge of the transition and observation probabilities and the observed gestures. These relationships can be depicted by a Bayesian influence network as shown in Figure 6 where the open circles represent hidden variables, the shaded circles denote observed variables, and the squares represent actions [4].

Let the distribution of the hidden state s_{t-1} at time $t-1$ be denoted by $b_{t-1}(s_{t-1})$, then the inference problem is to find $b_t(s_t)$ given b_{t-1}, a_{t-1} and o_t. This is easily solved using Bayes' rule

$$\begin{aligned}
b_t(s_t) &= P(s_t|o_t, a_{t-1}, b_{t-1})\\
&= p(o_t|s_t)P(s_t|a_{t-1}, b_{t-1})/p(o_t|a_{t-1}, b_{t-1})\\
&= p(o_t|s_t)\sum_{s_{t-1}} P(s_t, s_{t-1}|a_{t-1}, b_{t-1})/p(o_t|a_{t-1}, b_{t-1})\\
&= k\,.\,p(o_t|s_t)\sum_{s_{t-1}} P(s_t|s_{t-1}, a_{t-1})b_{t-1}(s_{t-1}), \qquad (1)
\end{aligned}$$

where $k = 1/p(o_t|a_{t-1}, b_{t-1})$ is a normalization constant. The distribution of states is often denoted by an N-dimensional vector $b = [b(s_1), \ldots, b(s_N)]'$ called the belief state. The belief update can then be written in matrix form as

$$b_t = k\,.\,O(o_t)T(a_{t-1})b_{t-1}, \qquad (2)$$

where $T(a)$ is the $N \times N$ transition matrix for action a, and $O(o) = \mathrm{diag}([p(o|s_1), \ldots, p(o|s_N)])$ is a diagonal matrix of observation probabilities. Thus, the computational complexity of a single inference operation is $\mathcal{O}(N^2 + 3N)$ including the normalization. For the case of the simple iPhone example where $N = 3$, this is entirely manageable. However, more complex examples will have very much larger values of N making exact computation intractable. This topic will be discussed in more detail later.

Given some assumed initial value b_o, (2) allows the belief state to be updated as each successive gesture is observed. Since the actual state is unknown, the action taken at each turn must be based on the belief state rather than the underlying hidden state. This mapping from belief state to action is determined by a policy $\pi : b \rightarrow a$. The quality of any particular policy is quantified by assigning rewards $r(s, a)$ to each possible state-action pair. For example, in the iPhone example, the rewards shown in Table 1 might be used. These give a positive reward for taking the action that matches the user's intent and a negative reward for taking the

[TABLE 1] REWARDS FOR EACH STATE-ACTION COMBINATION.

		ACTION			
		scroll-backward	delete-photo	scroll-forward	do-nothing
STATE	backward	+1	−20	−1	0
	delete	−1	+5	−1	0
	forward	−1	−20	+1	0

[TABLE 2] TRANSITION MATRIX $P(s'|s, a)$. EACH 3 × 3 GRID CORRESPONDS TO THE STATE TRANSITION MATRIX FOR ONE SPECIFIC ACTION. THE COLUMN LABELS *b*, *d*, AND *f* ARE ABBREVIATIONS FOR THE STATES backward, delete, AND forward.

		STATE s'											
		b	**d**	**f**	**b**	**d**	**f**	**b**	**d**	**f**	**b**	**d**	**f**
STATE s	backward	1	0	0	0.3	0.4	0.3	1	0	0	1	0	0
	delete	1	0	0	0	0	1	0.1	0.4	0.5	0	1	0
	forward	0.1	0.4	0.5	0	0	1	0.2	0.3	0.5	0	0	1
ACTION a			scroll-forward			scroll-backward			delete-photo			do-nothing	

wrong action. Incorrect deletions are punished most strongly since from the user's point of view, accidentally deleting the wrong photo is the worst error that the system can make.

The choice of specific rewards is a design decision and different rewards will result in different policies and differing user experiences. The choice of reward function may also affect the learning rate during policy optimization. However, once the rewards have been fixed, the quality of a policy is measured by the expected total reward over the course of the user interaction

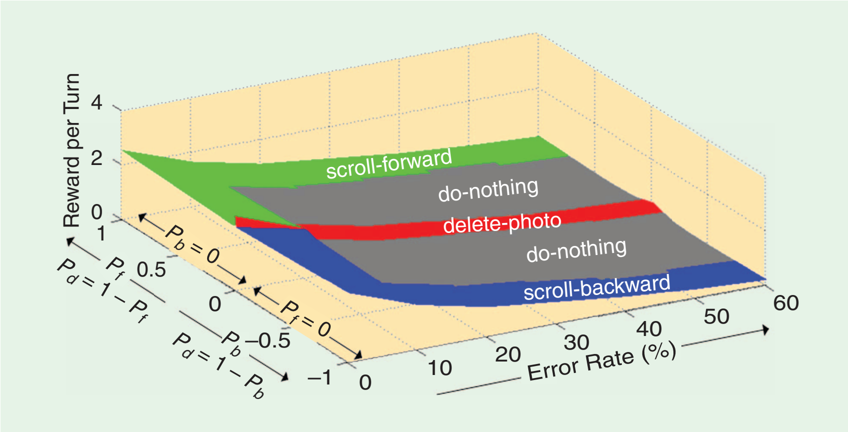
$$R = \mathcal{E}\left\{\sum_{t=1}^{T}\sum_{s} b_t(s)r(s,a_t)\right\} = \mathcal{E}\left\{\sum_{t=1}^{T} r(b_t,a_t)\right\}. \qquad (3)$$

Policy optimization is then equivalent to maximizing R.

If the process is Markovian, the total reward $V^{\pi}(b)$ expected in traversing from any belief state b to the end of the interaction following policy π is independent of all preceding states. Using Bellman's optimality principle, it is possible to compute the optimal value of this value function iteratively

$$V^*(b) = \max_{a}\left\{r(b,a) + \sum_{o} p(o|b,a)V^*(\tau(b,a,o))\right\}, \quad (4)$$

where $\tau(b,a,o)$ represents the state update function defined in (2) [5]. This iterative optimization is an example of reinforcement learning [6].



[FIG7] Policy value functions plotted over a single compressed belief space for varying gesture error rates. The horizontal axis is divided into two and denotes the probabilities that the user wishes to move forward (P_f), move backward (P_b) and delete (P_d). In the left part $P_b = 0$ and P_f varies from one down to zero while P_d increases from zero to one. The right axis is the mirror where $P_f = 0$ and P_b varies from zero to one while P_d decreases from one to zero. The other horizontal dimension denotes the error rate and the vertical axis is the average reward per turn. The coloring of the surface denotes the optimal action to take at each point along the belief state dimension.

This optimal value function for finite interaction sequences is piecewise-linear and convex. It can be represented as a finite set of N-dimensional hyperplanes spanning belief space where each hyperplane in the set has an associated action. This set of hyperplanes also defines the optimal policy since at any belief point b all that is required is to find the hyperplane with the largest expected value $V^*(b)$ and select the associated action [3].

Returning to the iPhone example, the user's behavior can be captured in a transition matrix T as in Table 2, and an observation probability matrix O can be estimated from distributions such as the one shown in Figure 4. Note that O will depend on the error rate of the user's gestures so for this illustration, observation matrices have been estimated for seven discrete error rates that range from 0% to 60%. Here we assume that a gesture is in error if its angle θ lies on the wrong side of the minimum error decision boundary i.e., $p(\omega_{\text{intended}}|\theta) < \text{p}(\omega_{\text{not-intended}}|\theta)$.

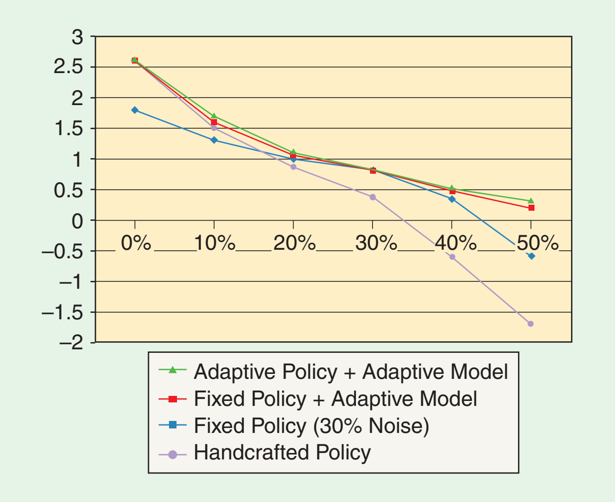
Given values for T and O and a reward function, a policy can be optimized using Bellman's optimality principle. As noted above, the policy will consist of a set of hyperplanes whose upper surface defines the optimal value function. The complexity of this surface increases with increasing gesture error rate. For the example here, the policy at 0% error rate consists of just three hyperplanes while the policy for 60% error rate consists of ≈37,000. Figure 7 summarizes the policies learned at each of the seven error rates where belief space has been compressed into a single dimension by assuming that $P_b = P(\text{backward}) = 0$ when $P_f = P(\text{forward})$ is significant. Thus, the rear area shows the value function surface when choosing between the forward and delete states, and the front area shows the value function surface when choosing between the delete and backward states. The color of the surface indicates the optimal system action to take at any point in belief space. As can be seen, at 0% errors, the policy will choose to move forward or backward unless the probability of delete is very close to one. At higher error rates, a do-nothing zone is introduced to avoid inadvertently deleting a photo. This also shows that the value function itself (expressed as an average reward per turn) falls steadily as the error rate increases.

The performance of the iPhone photo-sort application can be studied by sampling the transition and observation models to simulate the intentions and associated gestures of a typical user. This enables the average reward per turn to be determined for various configurations of the photo sorter running over a range of different user error rates. Figure 8 shows the results of these simulations. As can be seen, the conventional "hand-crafted policy" based on the algorithm described in Figure 5 performs well at low error rates but performance drops off rapidly in high noise. In Figure 8, the remaining plots all relate to policies trained using reinforcement learning. The plot marked "Fixed Policy (30% Noise)" shows the performance when the observation parameters and the policy are trained on data with a 30% error rate. As can be seen, robustness at high noise is improved relative to the hand-crafted policy but performance at low noise suffers. Examining the policy shows that this system is over-cautious at low error rates causing it to waste moves by choosing the do-nothing action. The plot labeled "Fixed Policy + Adaptive Model" shows what happens when the same fixed policy is used but the observation matrix is updated to model the true error rate. As can be seen, performance at low error rates is now restored and performance at high error rates is further improved. This shows the importance of accurate model parameters. Finally, the plot labeled "Adaptive Policy + Adaptive Model" shows the performance when the policy is also adapted to each error rate. In this case, a further small improvement is achieved.

Overall, these performance results illustrate the potential of Bayesian belief tracking and policy optimization for robustly responding to imprecise and ambiguous gestures. The gains illustrated in Figure 8 come from three main sources. First, the representation of the environment in the form of a probabilistic transition model enables knowledge of user behavior to help disambiguate badly formed gestures. Second, the use of an explicit observation model allows the expected noise characteristics to be modeled such that the implicit decision thresholds can be optimized. Third, reinforcement learning enables the policy to be tuned to maximize the expected reward and hence optimize the achievement of the desired communicative goals.

Of course, this is a toy example designed only to illustrate the basic ideas and the performance results displayed in Figure 8 and should be treated with caution. For example, in the adapted cases, the user simulator is using exactly the same parametric model as the system hence there is a perfect match. The upper curve in Figure 8 therefore represents an upper bound that would be hard to achieve in practice. Also, the hand-crafted policy was designed without knowledge of the reward function, hence the use of average reward as the performance metric biases in favor of the trained system. Nevertheless, the potential of combining Bayesian belief tracking with policies optimized via reinforcement learning should be clear.

As noted in the introduction, the system outlined in this section is an example of a POMDP. POMDPs meet all of the criteria required for a cognitive user interface: they support reasoning and inference via the Bayesian belief state tracking; they can plan under uncertainty via their policies that are based on belief states and trained using reinforcement learning; they are parametric and hence the underlying models can be rapidly adapted online; and since the policies are trained on data, they can be updated over longer time frames to learn from experience.



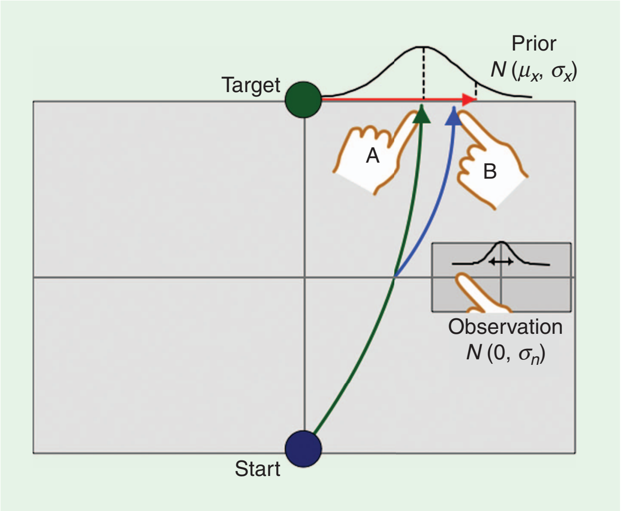
[FIG8] Average reward per turn versus gesture error rate for four configurations: hand-crafted policy, fixed policy trained at 30% noise, the same policy with adapted model parameters, and adapted policy and adapted model parameters.

POMDPs are by no means new. They arose originally in operational research [2], [5] and have since been extensively explored by the machine learning community. However, their widespread use has been hampered by severe tractability issues [7], [8]. These were touched on earlier when it was noted that belief tracking and policy optimization are exponentially dependent on the size of the state space. In fact, they are also exponentially dependent on the size of the action and observation spaces. As a result, their use in real-world applications such as user interfaces is not straightforward (this will be returned to later). Before that, however, it is instructive to take a small detour to explore how humans approach the problem of planning under uncertainty.

HUMAN DECISION MAKING AND PLANNING

A central tenet of this article is that human-computer interfaces will need to exhibit cognitive behavior to remain fit for purpose in future generations of computer-based systems, and it is further argued that Bayesian inference and reinforcement learning must underpin such interfaces. Since most interactions require collaborative behavior between a human and a machine, it would be comforting to know that the human side of this collaboration was operating under broadly similar mechanisms to the machine side. It is self-evident that humans rely heavily on reinforcement learning [9], but it is not so clear that they are capable of Bayesian inference. So it is interesting to address the question: "Is human decision making Bayesian?"

In evolutionary terms, one of the main drivers for the development of brain function is movement [10, Ch 5.]. In fact, it can be argued that the only reason humans have brains is so that they can move [11]. Hence, to understand the core mechanisms for inference in humans, it is necessary to understand how humans plan movement. Many experiments have been performed to answer this question, but the first to address the



[FIG9] A simple movement planning task. The subject must move their finger from the blue "Start" to the green "Target." However, during the movement, the subject's vision is obscured and the frame of reference is shifted by an amount x sampled from a Gaussian distribution $\mathcal{N}(\mu_x, \sigma_x)$. After repeated trials, subjects learn to move to the mean of the Gaussian as shown by Hand A. However, when subjects are shown a noisy glimpse of their finger position in midflight, they modify their trajectory as shown by Hand B. The question then is "What model is being used to make the correction?"

specific question as to whether or not movement planning is Bayesian was a colleague at Cambridge, Daniel Wolpert.

Wolpert's experiment was conceptually simple, and the main elements are illustrated in Figure 9 [12]. A human subject is asked to move her finger across a table from the blue home spot to the green target spot. However, during the movement, the subject's vision is obscured and the frame of reference is shifted by an amount x sampled from a Gaussian distribution $\mathcal{N}(\mu_x, \sigma_x)$. At the end of each movement, the subject can see the amount by which she missed the target and hence, after repeated trials, the subject learns to move her finger to the mean μ_x of the Gaussian (track a in Figure 9). The process is then repeated except that this time the subject is given a blurred glimpse of her finger relative to the shifted reference frame midway between home and target. This blurring is equivalent to applying Gaussian noise $\mathcal{N}(0, \sigma_n)$ to the observation (track b in Figure 9). So after many trials, the subject now has available, in principle at least, a prior and a noisy observation. The question then is how does the subject use this information to modify her plan?

There are three plausible models a subject could use to determine the most likely target position given a noisy observation mid-way through the movement. First, the prior information could be ignored and the noisy observation alone used to predict the target. In this case, as shown in Figure 10(a), the average error would be zero but there would be a large variance as shown by the green band.

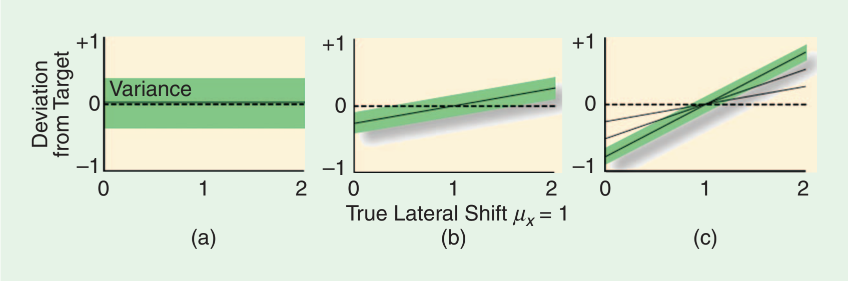
Second, subjects could learn a direct mapping between the noisy observation and the ensuing target error. By minimizing this error over a large number of trials, subjects could learn some form of optimal mapping without explicitly representing the prior distribution or the observation noise. Since the observation error was generated by blurring the image, subjects could estimate σ_n visually. However, they were only shown the target error for the case of zero blur case where $\sigma_n = 0$ and therefore if they did use a direct mapping algorithm, they would have been forced to use the same $\sigma_n = 0$ mapping for all trials regardless of its actual value. This leads to a response similar to that shown in Figure 10(b). Finally, if it was assumed that the human could internalize both the prior and observation distributions, then Bayes' rule could be used to predict the target. This would result in the familiar maximum a posteriori (MAP) estimate

$$\hat{x} = \frac{\sigma_n^2}{\sigma_n^2 + \sigma_x^2}\mu_x + \frac{\sigma_x^2}{\sigma_n^2 + \sigma_x^2}x, \qquad (5)$$

where as can be seen in Figure 10(c), the average deviation varies with the shift and the slope depends on the variance of the observation σ_n. Note also that this model gives the smallest variance and indeed it is the minimum variance solution for this estimation problem.

The results of this experiment showed unambiguously that the Bayesian model was the only one that fitted the experimental data. Further, analysis of the data showed that subjects were indeed learning the prior distributions. Furthermore, when a bimodal distribution was used for the prior, similarly consistent results were obtained suggesting that humans can estimate and compute with more complex distributions than simple Gaussians [13], [14].

More recent studies have shown that humans use Bayesian inference for other processing activities. For example, another Cambridge colleague, Máté Lengyel, has shown that humans use Bayesian learning for visual chunking [15]. In his experiments, subjects were shown patterns of the form shown in Figure 11(b). These patterns were constructed from an inventory of tile combinations (combos) shown in Figure 11(a). The inventory was kept hidden from the subjects. After



[FIG10] Average predicted deviation from target as a function of the observed shift for each of the three models: (a) full compensation; (b) direct mapping; (c) Bayesian. The thickness of the green band indicates the variance and the increasing slopes of the plots in (c) correspond to increasing values of observation noise σ_n.

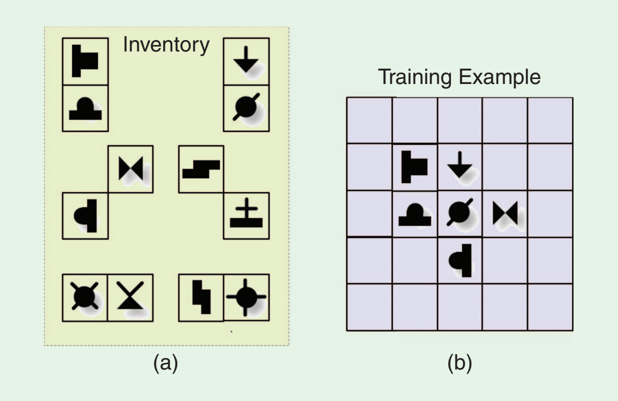
training, subjects were shown a variety of combos some of which were from the inventory and others were not. When subjects were asked to decide in each case whether or not the combo was familiar, they typically identified combos from the inventory as being familiar with around 75% accuracy, well above chance level. Several models have been proposed for how humans accomplish this task including a sophisticated associative learning algorithm. In contrast, Lengyel proposed that humans use a Bayesian chunk learning process that included an automatic Occam's razor property for determining the optimal model complexity and hence the combo size. By changing the frequencies of combo occurrence and the complexity of the combos (e.g., by using triples), these postulated mechanisms could be tested on the data and compared to human performance. In all cases, the Bayesian method fit the data closest.

Overall, experimental data shows that humans can implicitly assimilate Bayesian statistics and use Bayesian inference to solve problems of planning under uncertainty. The empirical evidence outlined here has subsequently been confirmed by many further experiments [16]–[18]. There are also architectural arguments to support the conjecture that the human nervous system is well suited to Bayesian inference [19]. Thus, it seems clear that humans have evolved the machinery both to learn statistical distributions by observation and to use Bayes' rule to infer posteriors from these distributions. Thus it would seem that humans do indeed use Bayesian inference in problem solving and planning under uncertainty.

SCALING-UP TO REAL-WORLD SYSTEMS

In the section "An Example: A Simple Gesture-Driven Interface," the basic ideas of POMDPs were outlined and their potential for providing robust and adaptable user interfaces was illustrated via a simple example. The key features of the POMDP framework are the maintenance of a system of beliefs, continually updated using Bayesian inference, and the use of a policy whose performance can be quantified by a system of associated rewards and optimized using reinforcement learning. As indicated in the preceding section, there are strong indicators that humans exploit similar mechanisms and overall, POMDPs appear to address all of the requirements listed in the introduction for an interface to qualify as being cognitive. So why is the POMDP framework not used in current user interfaces?

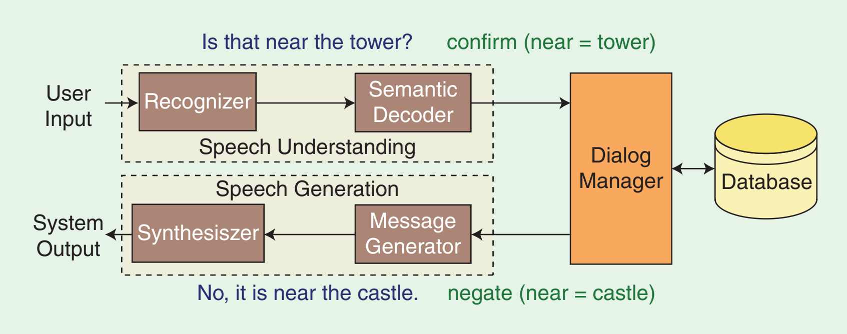
The answer to this question was alluded to at the end of the section "An Example: A Simple Gesture-Driven Interface." The state space of a real-world human-computer interface is typically very large. Hence, naïve implementation of belief monitoring via (1) would be far too costly for real-time operation. Furthermore, exact implementation and optimization of POMDP policies is intractable for all but the smallest of toy problems. Nevertheless, none of these issues need be a barrier to progress.



[FIG11] Visual chunking task: subjects were shown training patterns of the form shown in (b) from the inventory of basic tiles shown in (a) while keeping the inventory hidden. Subjects were then shown individual tiles from the inventory and asked if the tile was familiar or not.

The key elements of the POMDP framework are the ability to represent uncertainty by maintaining alternative hypotheses and the use of a quantifiable decision process that can be optimized. There are several approaches to POMDP approximation that preserve these essential elements and that can be shown empirically to yield significantly improved performance compared to conventional approaches. The remainder of this section will illustrate these by using the design of a statistical SDS as an example.

SDSs are widely used to provide voice-based access to information systems in areas such as banking, finance, and travel. More recently they are finding increasing use in call center automation. The architecture of a typical spoken dialog system for the tourist information domain is shown in Figure 12. The user's voice in the form of an audio waveform is first converted by a speech recognizer into words and then a semantic decoder converts the words into dialog acts. The latter are abstract representations of a user's intentions such as *inform(food=chinese), confirm(near=tower).* Typically the type of a dialog act (inform, confirm) is application independent whereas the attribute-value pairs that form the arguments are application specific. The user's dialog acts are passed to



[FIG12] Architecture of a spoken dialog system for the tourist information domain. The speech recognizer produces a word string that is converted by the semantic decoder into an abstract representation of the user's intent called a dialog act. The user's dialog acts are passed to a dialog manager that interprets the act, updates its internal state and generates a suitable response in the form of an output dialog act that is expanded back into natural language and converted into a waveform by a speech synthesizer.

a dialog manager that interprets the act, updates its internal state, and generates a suitable response in the form of an output dialog act. This is then expanded back into natural language and converted into a waveform by a speech synthesizer.

SDSs exemplify all of the issues that arise in the design of a real-world cognitive user interface. The internal state s is usually decomposed into three factors $s = \{g, u, h\}$ where g represents the user's goal, u represents the user input act, and h represents the dialogue history [20]. All of these are highly complex. Furthermore, since speech recognition error rates are typically high, there is significant uncertainty in the decoded user utterances u and this propagates uncertainty into g and h. In addition, the system action space must cover every possible system response so policies must map from complex and uncertain dialog states into a large space of possible actions. All of these factors combine to make implementation of SDSs within the POMDP framework a significant challenge.

Nevertheless, the POMDP framework can be scaled to real-world tasks by exploiting a few simple ideas. First, belief monitoring can be made tractable by simplifying the representation of the state distribution. For example, suppose that in a tourist information domain the user's goal consisted of four discrete values: type, location, price, food. Exact belief monitoring would require that the full joint distribution over $P(type, location, price, food)$ be maintained; but even a modest number of types, locations, price-points, and food types would quickly render the full joint impossibly large. One of the simplest ways of dealing with this is to use an M-best approximation, whereby the probability of all state values are ranked and pruned to retain only the M most likely states. For instance, the ranked list for the tourist example might be as follows:

P(hotel, east, cheap, none) = 0.65

P(hotel, west, cheap, none) = 0.21

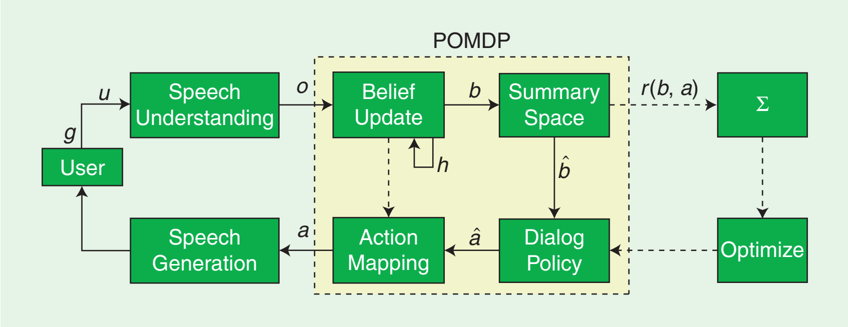
P(restaurant, east, cheap, italian) = 0.08

P(bar, east, cheap, none) = 0.04

P(hotel, east, expensive, none) = 0.01

…

where all combinations other than those shown have such low probabilities that they are not worth maintaining.



[FIG13] The HIS system. The HIS dialog manager maintains a belief distribution over all possible dialog states. To make policy representation and optimization tractable, the belief distribution is mapped into a simpler summary space. Responses to the user are generated by expanding summary actions into full system actions using a heuristic action mapping.

A second approach to belief state approximation is to factor the joint distribution by making some independence assumptions. For example, from knowledge of the domain it might be argued that the kind of food and the price of a venue depends on only on its type, and the type and location are independent, then

$$P(type, location, price, food) \approx P(price|type) \\ P(food|type)P(type)P(food). \quad (6)$$

This results in a Bayesian network representation for the dialog state.

Both of the above approaches can deliver tractable belief monitoring, but they still leave the state space too large to facilitate effective policy optimization. The general approach to dealing with this is to map the so-called master belief space b into a more compact summary space $\hat{b}$ via a mapping function $\phi : b \to \hat{b}$ and then perform optimization in that summary space. In a similar way, a compact action set $\hat{a}$ can be defined in summary space, which is then mapped back into master space by an inverse mapping $\psi : \hat{a} \to a$ [21].

The following two sections briefly outline the operation of two statistical dialog systems built at Cambridge that illustrate these two approaches to approximating the POMDP framework. The first is the HIS system that exemplifies the M-best approach. The second is the BUDS system that exemplifies the Bayesian network approach. Both use master-summary space mappings but in differing ways. A more general overview of statistical dialog systems is given in [22].

THE HIDDEN INFORMATION STATE SYSTEM

A block diagram of the HIS system is shown in Figure 13 [23], [24]. It exploits both the M-best approximation for belief monitoring and summary space mapping for policy optimization. A typical dialog turn follows the basic cycle illustrated in Figure 12. Each user input is processed by the speech understanding component that outputs an N-best list of alternative hypotheses and associated confidence scores $[< u_1, c_1 >, \ldots, < u_N, c_N >]$. This list is treated as an observation o by the POMDP dialogue manager from which the belief state b is updated. The updated belief state is then mapped into a summary belief state $\hat{b}$. The dialog policy associates a summary action $\hat{a} = \psi(\hat{b})$ with every possible summary belief state. This summary action is then mapped back into master space and converted into a system response a to the user.

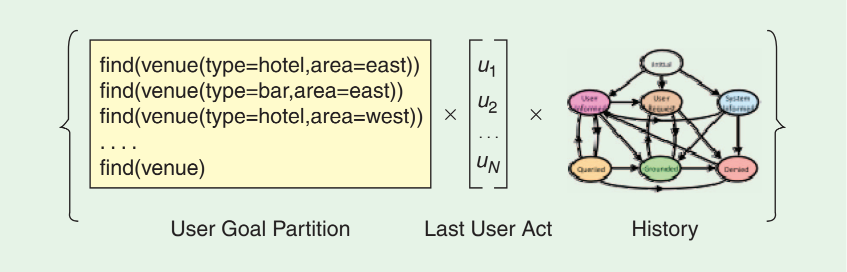
The HIS system state is composed of the three principal factors mentioned earlier, that is, $s = \{g, u, h\}$ where g is the user's goal, u is the last user act and h is the dialog history. If this factorization is plugged into (1) and some reasonable independence assumptions are made, it is straightforward to show that

$$b'(g',u',h') = k \cdot \underbrace{P(o'|u')}_{\substack{\text{observation}\\ \text{model}}} \underbrace{P(u'|g',a)}_{\substack{\text{user action}\\ \text{model}}} \sum_{g,h} \underbrace{P(g'|g,a)}_{\substack{\text{user goal}\\ \text{model}}} \underbrace{P(h'|g',u',h,a)}_{\substack{\text{dialogue history}\\ \text{model}}} b(g,h), \qquad (7)$$

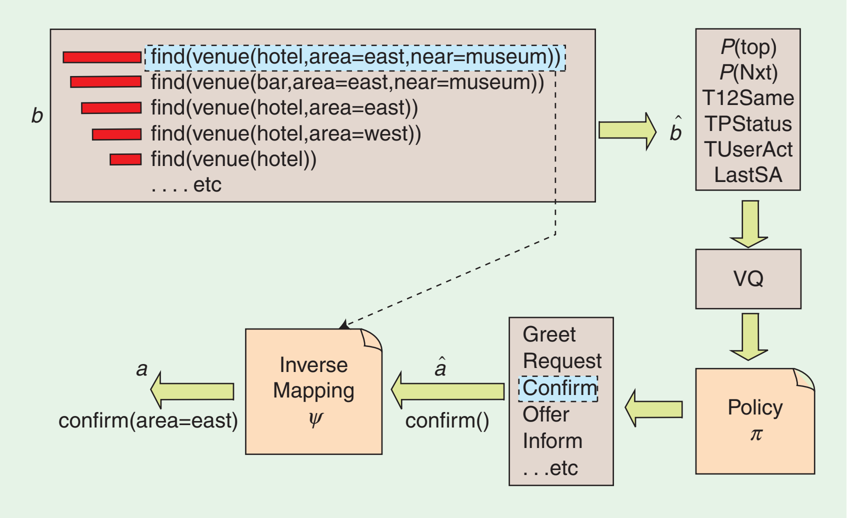
where the primes indicate the next time step [25]. As shown by under-braces, the belief update equation for a spoken dialog system involves four distinct probability models. The user goal model and the history model represent the dynamics of the underlying Markov decision process. In the HIS system, it is assumed that the user's goal does not change and the history model is replaced by a deterministic finite state grounding model. More interesting are the observation and user action models. The observation model encodes the error characteristics of the speech understanding system in an analogous manner to the observation matrix in the iPhone example of the section "An Example: A Simple Gesture-Driven Interface." The user action model then allows this observation probability to be scaled by the probability that the user would speak u' given the goal g' and the last system prompt a. Since the observation is an N-best list of hypotheses, the user action model effectively allows this list to be reranked according to the context. Thus the user action model provides a context sensitive filter that is extremely effective at reducing errors, especially in high noise conditions [26].

To simplify belief monitoring further, the HIS system groups user goal states into equivalence classes called partitions. At the start of a dialog all goal states are in a single partition. As new evidence is received via the input user acts, the partitions are expanded to preserve the different possible goals. This expansion follows a set of ontology rules derived from the database, and it is tree-structured to ensure that the union of all partitions equals the complete state space. Since all of the goals in a partition are indistinguishable based on the evidence so far, beliefs can be updated at the partition level rather than at the individual state level and this significantly reduces the computational load. The resulting HIS state space is illustrated schematically in Figure 14. Each partition of HIS states consists of a user goal partition combined with a hypothesized last user act and a set of grounding information. The latter constitutes the dialog history such that every node in the partition tree has a grounding state that changes according to a finite state transition network. Thus overall, the HIS state space consists of all possible partitions combined with all possible hypothesized user acts combined with all possible combinations of grounding states. The probability of every partition of states in this set is evaluated, rank ordered and pruned. The system typically maintains 300–3,000 active partitions and together these constitute its belief state b.

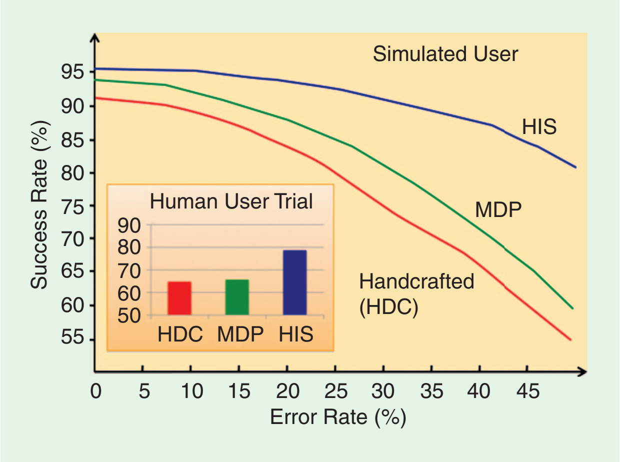
Master-summary space mapping and policy representation in the HIS system are shown in Figure 15. The summary space $\hat{b}$ consists of a fixed-length vector of features such as the probability of the top state in master space, the probability of the next-to-top state, and a Boolean denoting whether the top and next-to-top state could represent the same entity. This summary vector is then mapped into a fixed grid point in summary space by a vector quantizer [27], [28]. Each grid point has an associated summary system action that is selected and mapped back to master space by an inverse mapping function, which in most cases assumes that the topic of the system action concerns the most likely state in master belief space. This is illustrated in Figure 15 where the confirm action is selected in summary space and the specific attribute to confirm is selected from the top-ranked state in b.



[FIG14] **The HIS belief space. Each combination of a goal partition, last user act and set of history states constitutes a single partition of states in belief space. The history states record grounding and query status information but the details are not relevant here (see [24]).**



[FIG15] **HIS master-summary state mapping. The summary space $\hat{b}$ consists of a fixed-length vector of features that is mapped into a fixed grid point in summary space by a vector quantizer. Each grid point has an associated summary system action that is selected and mapped back to master space by an inverse mapping function.**



[FIG16] HIS performance as a function of the input error rate. The main graph shows percentage success rate as a function of semantic error rate using a simulated user. The inset shows the comparable results for a human user trial held in noisy conditions for which the average error rate was 25%.

The vector quantization of the continuous summary space converts the HIS POMDP into a simple discrete Markov decision process (MDP) for which many optimization algorithms exist. In fact, the HIS system uses the Monte Carlo control algorithm in conjunction with a user simulator to estimate the optimal action set via online reinforcement learning [29].

Figure 16 shows the performance of the HIS system compared to an MDP-based dialog manager that only maintains a single most likely dialog state and a version of the HIS system that has multiple states but only a hand-crafted policy. The main graph shows the average success rate as a function of the input semantic error rate using a user simulator, where a dialogue is said to be successful if it returns a venue to the user that satisfies her constraints and any information requested about that venue such as the address or telephone number. As can be seen, the HIS system is substantially more robust at high error rates. The inset summarizes the aggregate results of a user trial in which 36 subjects undertook a variety of tasks in noisy conditions [24]. Again, the HIS system is clearly more robust.

THE BAYESIAN UPDATE OF DIALOG STATE SYSTEM

The HIS system described in the previous section illustrates how the M-best approach to belief space approximation coupled with a master-summary state mapping can lead to a tractable real-world dialog system. Although the HIS system can deliver improved performance relative to a conventional system, it suffers from two major problems. First, the M-best approximation makes it difficult to represent a state transition matrix and hence, the HIS system assumes that the user's goal does not change during the course of a dialogue. Second, the probability models in the HIS system are hybrids of deterministic rules and statistical models that are difficult to train automatically from data.

As explained in the introduction to this section, an alternative approach is to use a Bayesian network representation for the dialog state. This form of approximation retains the ability to properly represent system dynamics and to use fully parametric models but at the cost of ignoring much of the conditional dependency inherent in real-world domains.

An example of this approach is another system built at Cambridge called the BUDS system [30]. This system uses the same factorization of the dialog state $s = (g, u, h)$ as the HIS system, but it further factorizes each component into concepts. For example, in the tourist information domain, the user might be interested in such concepts as *location, price, food, room-rate,* and *music*. Most of these concepts will be dependent on the type of venue involved (restaurant, bar, hotel, etc.) but otherwise they can be deemed to be independent. This results in a dynamic Bayesian network structure of the form shown in Figure 17 that shows the venue *type* and one dependent concept *food*. Note that in the actual system there are between ten and 20 concepts depending on the application. Each concept c has three principle nodes: a goal node g_c with values ranging over the user's possible choices; a user act node u_c denoting the type of the last user act or null if the last user act did not mention this concept; and a history node h_c taking the values of a simple grounding model such as (*initial, mentioned, grounded*). All u_c nodes depend on a single node representing the full user act and this in turn depends on the observation, which like the HIS system is normally an N-best list of hypothesized user dialog acts. System dynamics are represented by adding dependencies on the equivalent node in the previous time slot. In the BUDS system, both the goal nodes and the history nodes depend on their previous values.

The BUDS system represents all of the relevant dialog state information in a single Bayesian network structure. Belief monitoring can therefore utilize any of the existing algorithms for approximate inference in a Bayesian network. In the BUDS system, loopy belief propagation (LBP) is used [4]. However, to make this run in real time, various optimizations are essential. For example, goal nodes may range over a large set of values whereas only a very few of these values will ever be mentioned in a single dialog. Standard LBP can thus be made much faster by partitioning the values in each slot similar to the way states are partitioned in the HIS system. This typically reduces the effective cardinality of each slot down to two or three and this has a dramatic effect on computation times. Another very effective optimization is to assume that the probability of user goal changing is constant. This reduces the cardinality of a user goal transition matrix with n possible values from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [31].

Distributing the belief space over a large number of factors requires a different approach to policy representation since direct mapping of each master state into summary space is no longer possible. In the BUDS system, a stochastic policy is used in the form of a softmax function with parameters $\boldsymbol{\theta}$ as follows:

$$\pi(a|b, \boldsymbol{\theta}) = \frac{e^{\boldsymbol{\theta}.\boldsymbol{\phi}_a(b)}}{\sum_{a'} e^{\boldsymbol{\theta}.\boldsymbol{\phi}_{a'}(b)}}, \qquad (8)$$

where $\phi_a(b)$ is a basis function for action a. These basis functions can be separated into components to allow the separate effects of each concept goal node in the Bayesian network to contribute to the overall policy

$$\phi_a(b)^{\mathrm{T}} = [\phi_{a,1}(b)^{\mathrm{T}}, \ldots, \phi_{a,G}(b)^{\mathrm{T}}, \phi_{a,*}(b)^{\mathrm{T}}], \tag{9}$$

where the indices $1 \ldots G$ range over the relevant goal nodes and the final term $\phi_{a,*}(b)^{\mathrm{T}}$ allows global information to be incorporated such as the number of database entries matching the most likely user goal. This parametric policy can be optimized by maximizing the expected reward and in the BUDS system, the natural actor critic algorithm has been found to be the very effective for this [32].

Like the HIS system, the BUDS system has been tested both with a user simulator and in human trials. Performance results are typically similar or slightly better than the HIS results and will not be repeated here. However, the major advantage of the BUDS system is that the network can be extended to include the model parameters themselves along with appropriate Dirichlet priors. If loopy belief propagation is then replaced by expectation propagation [33], the system can learn the model parameters and adapt online from data. Thus, architectures similar to BUDS can fulfill all of the requirements listed in the introduction for building cognitive user interfaces.

CONCLUSIONS AND PERSPECTIVES

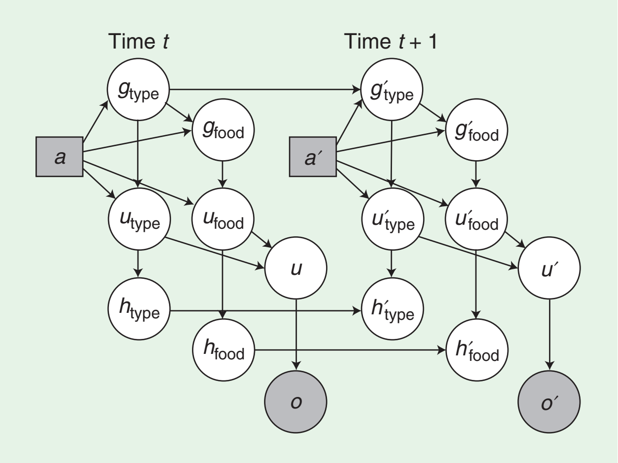
The central claim of this article has been that future generations of computer-based systems will need user interfaces that can support significantly more robust and intelligent interaction than is possible with current approaches. It has been argued that future interfaces must provide cognitive functionality that includes the ability to support inference and reasoning, planning under uncertainty, and both short-term and long-term adaptation. The framework proposed to support such cognitive user interfaces is based on POMDPs that integrate Bayesian belief monitoring and reward-based reinforcement learning. This framework has been shown empirically to provide significantly more robust interpretation of imprecise and ambiguous human interactions, and it also provides the ability to plan interactions so as to maximize the desired objective functions. Furthermore, it appears that humans utilize similar machinery.

If this argument is accepted, the implications are that we must start approaching the design of human-centric IT systems in a different way. The key is to identify what are the main sources of uncertainty and how they can be efficiently represented within the system. Inputs from the user must then be treated as evidence via which uncertainty can be resolved through Bayesian inference. Although POMDPs are often thought to be intractable for real-world problems, in practice, the use of sensible approximations can lead to practical systems that still retain the essential benefits of the POMDP framework.

The HIS spoken dialog system was described at some length to illustrate how this can be done. The HIS system also demonstrates that there is an evolutionary path from existing systems to this new paradigm. The HIS system is, in effect, an efficient representation of multiple dialog managers acting in parallel, each one making different assumptions about what the user intends (see also [34] and [35]). The HIS system contains symbolic components similar to those found in conventional dialog systems, and indeed, this ability to integrate conventional system components into a probabilistic framework is one of its main strengths.

In the longer term, however, cognitive user interfaces will need to be designed as distributed probabilistic models from the ground up since this is the only way to ensure that systems can evolve with time and adapt from experience. For these types of systems, the use of Bayesian networks as illustrated by the BUDS system is compelling. However, the deployment of large-scale Bayesian network-based POMDP systems presents many challenges. The immediate problem is maintaining real-time operation as network complexity increases since approximate inference requires considerable computation. As demonstrated by the BUDS system, significant speed gains can be achieved from conventional belief propagation algorithms by exploiting the fact that human interaction tends to focus on just a few specific entities, and further optimizations of this sort are clearly possible. However, ultimately we will need systems that can support very large dynamically changing networks, and for these support may be needed from the underlying hardware, perhaps in the form of distributed processors optimized for the type of message passing operations needed by belief propagation algorithms. There are also other challenges such as integrating multimodal input and output channels and handling the many subtle dialog phenomena found in natural communication between humans. There are also social challenges since, unlike present-day systems, it may not be possible to guarantee precisely how a cognitive user interface will respond in certain situations [36].

With a few notable exceptions, the traditional philosophy of an IT system design has been to encode all information as



[FIG17] BUDS uses a dynamic Bayesian network in which the dialog state is decomposed into *slots* representing features such as kind of *food, price,* and *location*. Each slot has goal *g*, an associated user dialog act *u*, and history information *h*. The slots are mostly independent except that all slots depend on the venue type.

tabulated data, manipulated by deterministic algorithms. Inputs are also viewed as being deterministic and the obvious frustration of the IT industry with speech recognition technology has arisen through the notion that speech is a keyboard substitute, and all that is required to make it useful as such is to reduce the error rate. This article has argued that such a view is misplaced and that progress towards providing truly cognitive human-computer interfaces will require a radically different approach in which the explicit modeling of uncertainty is at the core. POMDPs provide a well-founded framework for properly engineering such systems, and they are the key to the cognitive user interface of the future.

ACKNOWLEDGMENTS

The author would like to thank current and former members of the Cambridge Dialogue Systems Group: Milica Gašić, Filip Jurčíček, Simon Keizer, Fabrice Lefevre, François Mairesse, Jost Schatzmann, Matt Stuttle, Blaise Thomson, Karl Weilhammer, Jason Williams, and Kai Yu. Some of the research mentioned in this article was funded by the UK EPSRC under grant agreement EP/F013930/1 and by the EU FP7 Programme under grant agreement 216594 (CLASSIC project: www.classic-project.org). Thanks are also given to the anonymous reviewers whose comments and suggestions helped improve the final version of this article.

AUTHOR

Steve Young (sjy@eng.cam.ac.uk) is a professor of information engineering at Cambridge University, where he is currently senior pro-vice chancellor. His main research interests lie in the area of spoken language systems including speech recognition, speech synthesis, speech understanding, and statistical dialogue management. He was the original developer of the HTK Toolkit and codeveloper with Phil Woodland of the HTK Large Vocabulary Speech Recognition System. He was editor of *Computer Speech and Language* from 1993 to 2004, and he is currently chair of the IEEE Speech and Language Processing Technical Committee. He is a Fellow of the Royal Academy of Engineering, the Institution of Engineering and Technology, the IEEE, and the RSA. In 2004, he received the IEEE Signal Processing Society Technical Achievement Award. In 2008, he was elected Fellow of the International Speech Communication Association (ISCA), and in 2010 he received the ISCA Medal for Scientific Achievement.

REFERENCES

[1] Apple Inc. (2009). iPhone human interface guidelines [Online]. Available: http://developer.apple.com/iphone/library/documentation

[2] E. Sondik, "The optimal control of partially observable Markov decision processes," Ph.D. dissertation, Stanford Univ., Palo Alto, CA, 1971.

[3] L. Kaelbling, M. Littman, and A. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, pp. 99–134, 1998.

[4] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[5] R. Smallwood and E. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Oper. Res.*, vol. 21, no. 5, pp. 1071–1088, 1973.

[6] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction (ser. Adaptive Computation and Machine Learning)*. Cambridge, MA: MIT Press, 1998.

[7] M. Littman, A. Cassandra, and L. Kaelbling, "Learning policies for partially observable environments: Scaling up," in *Proc. 12th Int. Conf. Machine Learning*, A. Prieditis and S. Russell, Eds. San Francisco, CA: Morgan Kaufmann, 1995, pp. 362–370.

[8] Y. Virin, G. Shani, S. E. Shimony, and R. Brafman, "Scaling up: Solving POMDPs through value based clustering," in *Proc. 22nd Nat. Conf. Artificial Intelligence, AAAI 2007*, Vancouver, 2007.

[9] W.-T. Fu and J. Anderson, "From recurrent choice to skill learning: A reinforcement-learning model," *J. Exp. Psychol. Gen.*, vol. 135, no. 2, pp. 184–206, 2006.

[10] G. Cziko, *Universal Selection Theory and the Second Darwinian Revolution*. Cambridge, MA: MIT Press, 1995.

[11] D. Wolpert, Z. Ghahramani, and J. Flanagan, "Perspectives and problems in motor learning," *Trends Cogn. Sci.*, vol. 5, no. 11, pp. 487–494, 2001.

[12] K. Kording and D. Wolpert, "Bayesian integration in sensorimotor learning," *Nature,* vol. 427, pp. 224–227, 2004.

[13] R. Jacobs, "Optimal integration of texture and motion cues to depth," *Vision Res.*, vol. 39, pp. 3621–3629, 1999.

[14] M. Ernst and H. Bulthoff, "Merging the senses into a robust percept," *Trends Cogn. Sci.*, vol. 8, pp. 162–169, 2004.

[15] G. Orban, J. Fiser, R. Aslin, and M. Lengyel, "Bayesian learning of visual chunks by human observers," *Proc. Nat. Acad. Sci.*, vol. 105, no. 7, pp. 2745–2750, 2008.

[16] K. Kording and D. Wolpert, "Bayesian decision theory in sensorimotor control," *Trends Cogn. Sci.*, vol. 10, no. 7, pp. 319–326, 2006.

[17] H. Tassinari, T. Hudson, and M. Landy, "Combining priors and noisy visual cues in a rapid pointing task," *J. Neurosci.*, vol. 26, no. 40, pp. 10154–10163, 2006.

[18] D. Wolpert, "Probabilistic models in human sensorimotor control," *Hum. Mov. Sci.*, vol. 26, pp. 511–524, 2007.

[19] M. Sahani and P. Dayan, "Doubly distributional population codes: Simultaneous representation of uncertainty and multiplicity," *Neural Comput.*, vol. 15, pp. 2255–2279, 2003.

[20] J. Williams, P. Poupart, and S. Young, "Factored partially observable Markov decision processes for dialogue management," in *Proc. 4th Workshop Knowledge and Reasoning in Practical Dialogue Systems*, Edinburgh, 2005.

[21] J. Williams and S. Young, "Scaling POMDPs for spoken dialog management," *IEEE Trans. Audio, Speech and Lang. Processing*, vol. 15, no. 7, pp. 2116–2129, 2007.

[22] O. Lemon and O. Pietquin, "Machine learning for spoken dialogue systems," in *Proc. Interspeech*, Antwerp, Belgium, 2007, pp. 2685–2688.

[23] S. Young, J. Schatzmann, K. Weilhammer, and H. Ye, "The hidden information state approach to dialog management," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, ICASSP 2007,* Honolulu, HI, 2007.

[24] S. Young, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The hidden information state model: A practical framework for POMDP-based spoken dialogue management," *Comput. Speech Lang.*, vol. 24, no. 2, pp. 150–174, 2009.

[25] J. Williams and S. Young, "Partially observable Markov decision processes for spoken dialog systems," *Comput. Speech Lang.*, vol. 21, no. 2, pp. 393–422, 2007.

[26] S. Keizer, M. Gasic, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Modelling user behaviour in the HIS-POMDP dialogue manager," in *Proc. IEEE Workshop Spoken Language Technology (SLT'08)*, Goa, India, 2008.

[27] W. Lovejoy, "Computationally feasible bounds for partially observed Markov decision processes," *Oper. Res.*, vol. 39, pp. 162–175, 1991.

[28] R. Brafman, "A heuristic variable grid solution method for POMDPs," in *Proc. 14th Nat. Conf. Artificial Intelligence, AAAI*, Cambridge, MA, 1997.

[29] M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and S. Young, "Training and evaluation of the HIS POMDP dialogue system in noise," in *Proc. 9th SIGdial Workshop Discourse and Dialogue 2008*, Columbus, OH, 2008.

[30] B. Thomson, J. Schatzmann, and S. Young, "Bayesian update of dialogue state for robust dialogue systems," in *Proc. Int. Conf. Acoustics Speech and Signal Processing, ICASSP*, Las Vegas, 2008.

[31] B. Thomson and S. Young, "Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems," *Comput. Speech Lang.*, to be published.

[32] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7–9, pp. 1180–1190, 2008.

[33] T. Minka, "Expectation propagation for approximate Bayesian inference," in *Proc. 17th Conf. Uncertainty in Artificial Intelligence,* Seattle, WA, 2001, pp. 362–369.

[34] J. Henderson and O. Lemon, "Mixture model POMDPs for efficient handling of uncertainty in dialogue management," in *Proc. 46th Annu. Meeting Association for Computational Linguistics (ACL'08)*, Columbus, OH, 2008.

[35] K. Kim, C. Lee, S. Jung, and G. Lee, "A frame-based probabilistic framework for spoken dialog management using dialog examples," in *Proc. 9th SIGdial Workshop Discourse and Dialogue*, Columbus, OH, 2008.

[36] T. Paek and R. Pieraccini, "Automating spoken dialogue management design using machine learning: An industry perspective," *Speech Commun.*, vol. 50, no. 8–9, pp. 716–729, 2008.

SP