INSTRUCTION DIALOGUES: TEACHING NEW SKILLS TO A ROBOT

C. Crangle and P. Suppes
IMSSS, Ventura Hall, Stanford University, Stanford, CA 94305.
COLLEEN@SUWATSON.BITNET

1. Introduction

This paper describes extended dialogues between a human user and a robot system. The purpose of each dialogue is to teach the robot a new skill or to improve the performance of a skill it already has. Our particular interest is in natural-language dialogues but the techniques we illustrate in this paper can be applied to any high-level language. The primary purpose of this paper is to show how verbal instruction can be integrated with the robot's autonomous learning of a skill.

The learning techniques we apply are based on a set of concepts developed within mathematical learning theory and thoroughly tested in human learning ([8] – [11]). These techniques relate directly to skill-performance tasks in which the subject (human or robot) learns to make responses along a continuum of values. A great many tasks we would want a robot to perform fall into this category. Whenever the robot is required to position its end-effector, for instance, some specific point on a line, or on a surface, or in 3-dimensional space, must be selected.

It is a feature of most high-level task specifications that they underspecify in that they fail to express many of the details the robot requires to carry out the task. Nowhere is this more evident than in natural-language descriptions. A request as simple as *Put the wrench away!* says nothing about where the wrench is to go or where on the target surface it is to be placed. A command such as *Place the wrench parallel with the front edge of the shelf, 4 inches from the left end and 2 inches from the front edge!* will leave little doubt about where to place the wrench. But such detailed and specific descriptions are tedious and, in fact, unnecessarily detailed. The speaker will typically have in mind not some one point but an area of the shelf — the left side, somewhere in the middle, towards the far right, for instance — and will sometimes be quite neutral as to the object's orientation. Some way must be found for the specific intention of the speaker to be communicated naturally and for the robot to respond appropriately. We aim to show how verbal instruction accompanied by autonomous learning provides a way.

In the instruction dialogues we have in mind the operator uses high-level commands to request some action from the robot — *Pick up the receiver!* or *Pick up the fork in the middle!*, for instance. When the robot responds to such a request, the operator makes free use of qualitative commands to correct or confirm the action taken — *Not so far up!*, *That's fine!*, or *Be more careful!*, for instance. These qualitative corrections are expected not only to alter the robot's current behavior, but also to influence its behavior in the future whenever the original request is repeated. In other words, the robot is expected to learn from its interaction with the operator.

It is useful to ask in what circumstances robot instruction is most appropriate. Whenever the robot's basic repertoire of skills must be expanded over time to meet the demands of a changing task load or a changing environment, instruction has an important role to play. If, in addition, it is the operator who must adjust the robot's functioning and the operator is not a robotic specialist — in space applications, for instance, the operator will typically be a specialist in his or her own field — instruction has an essential role to play. It is imperative in these circumstances that the operator be able to request action from the robot and adjust the robot's subsequent behavior in as natural a way as possible.

Skills or tasks to be taught are categorized by the number of response variables involved. For many robot tasks, each response variable will have the dimension of space or the dimension of time and we therefore talk of tasks being 1-dimensional or 2-dimensional, and so on. A typical 1-dimensional task is that of learning to select an interval on a line. This learning problem would arise, for instance, with a request to put one object on another much larger object — a box on a table, for instance — if the robot did not know the desired position for the box along the length of the table. The task becomes 2-dimensional if the setback of the box from the front of the table were also to be learned. A 2-dimensional learning task arises whenever the robot is directed to go somewhere (or move its arm someplace) in order to perform a specific action. For instance, if the robot is to go to a refrigerator unit or a storage cabinet to fetch something, it must stop in front of the refrigerator or cabinet at a point where it will not impede the door's opening. The set of points that are near enough to the refrigerator or cabinet but not in the way constitutes the region the robot must learn. Another 2-dimensional task is the seemingly straightforward one of setting the table for dinner. This activity entails a large number of learned skills: what the orientation of each knife, fork, and spoon should be relative to the plate, how far right of the plate the dinner knife should go, how far from the edge of the table the dinner plate and side plate should be placed, and so on. A wide range of 3-dimensional task skills are required in any assembly or disassembly process — placing one part in, under, or next to another part, for example. When the dimension of time is introduced, motions and sequences of motions can also be learned. In this paper we concentrate on 1-dimensional tasks.

2. Instruction dialogues

Each instruction dialogue is seen as a sequence of trials or steps. On each trial, the robot responds to a natural-language command from the operator by taking the action described in the command. This response is followed by feedback from the operator indicating whether the response was acceptable or not. The feedback is itself a natural-language command, either a congratulatory command such as *That's fine!* or *OK!* which indicates that the response was acceptable, or a corrective command such as *Further to the left!* or *That's way out!* which indicates that the response was unacceptable. When the response is acceptable we refer to it as a "hit," when unacceptable a "miss." After a hit, the operator typically repeats the original command to check that the robot has learned to respond appropriately to it.

The corrective feedback is nondeterminate in that it does not let the robot know exactly

what its response should have been. It merely indicates what the robot can do to improve its response on subsequent trials. Typically, there is no one correct response on a trial anyway but a range of acceptable responses within the target interval or region, or a range of motion paths. In addition, the operator will often not be able to provide determinate feedback. He or she will have a target interval, region, or motion in mind but will be unable to specify the endpoints of the interval or the exact coordinates of the region or the precise trajectory of the motion path. Instead the operator will use his or her judgement to determine whether the observed response appears acceptable or not. Although an essential guide to the robot's learning, that judgement is not infallible.

There are three categories of feedback: congratulatory feedback given after a hit (*Good!, That will do!, OK!, Fine!,* etc.), positional feedback given after a miss (*To the right!, Much further to the right!, A little bit to the right!, Too far right!, Much too far right!, Not that far!, More!, Again!, Further still!, A bit more!,* etc.), and accuracy feedback given after a miss (*Be more careful!, No need to be so cautious!, Slower!, Faster!, Slower next time!,* etc.). Figure 1 gives an example of an instruction dialogue. Note that there is no immediate motor response to congratulatory feedback or accuracy feedback. In either case, the robot waits for the original command to be repeated or for positional feedback.

		Robot's response
Original command:	*Put the wrench on the shelf!*	x_0
Positional feedback:	*Not that far left!*	x_1
Congratulatory feedback:	*That's fine!*	—
Original command:	*Put the wrench on the shelf!*	x_2
Positional feedback:	*A little further to the right!*	x_3
Positional feedback:	*More!*	x_4
Accuracy feedback:	*Be more careful!*	—
Positional feedback:	*A little to the left now!*	x_5
Congratulatory feedback:	*Good!*	—

Figure 1: Example of an instruction dialogue

It is assumed that on each trial the state of the robot with respect to the skill being taught is represented by a probability distribution. This distribution enters into the interpretation of the original command (the one that describes the skill being learned) and it comes into play in the interpretation of all feedback from the operator. In addition, the distribution changes in response to the operator's feedback, which in turn alters the interpretation of all subsequent commands.

The distribution plays another important role. It represents the target interval, region or motion associated with the skill being learned. For 1-dimensional tasks, therefore, we can use a single distribution of one variable. We use the notation $k_{m,v}(x)$ where m is the mean of the distribution, v the variance, and x a value of the response variable. If on trial n, $m = m_n$ and $v = v_n$ and the robot responds to the original command, then the probability

that the response on this trial will lie between a and b is given by $\int_a^b k_{m_n,v_n}(x)dx$. After the robot has made response x_n on trial n in response to a natural-language command, i.e., after it has selected a point on the response continuum as evidenced by its moving to that point in response to the command, one of the two kinds of events described earlier occurs: the robot is either told that it was unsuccessful, i.e., it missed the target interval, or that it was successful, i.e., it landed within or hit the target interval. Known as a *smearing* or *smoothing* function, $k_{m,v}(x)$ has the effect of spreading the effect of feedback at a point m around m on the continuum of responses.

The question we face in designing the robot so that it learns from its interaction with the operator is as follows: What effect should a hit or miss have on the function $k_{m,v}(x)$? Before we can answer that question, we must describe the choice of distribution for 1-dimensional tasks and the interpretation of commands.

3. The beta distribution

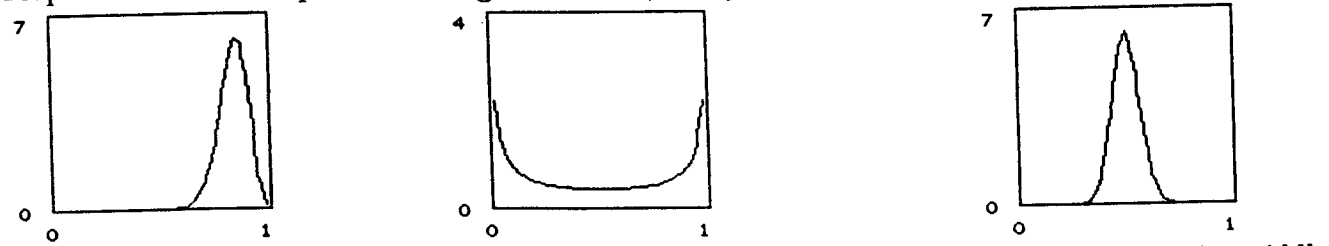
The probability distribution we use for 1-dimensional tasks is the beta distribution. It has two parameters α and β and is defined as follows for $0 < x < 1$ (Γ designates the gamma distribution):

$$\beta(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, \qquad \alpha > 0, \ \beta > 0.$$

The mean, m, and variance, v, of the distribution are calculated as follows:

$$m = \frac{\alpha}{\alpha+\beta}, \qquad v = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.$$

The beta distribution has several properties that make it suitable for our purposes. First, its usefulness in models of learning has already been demonstrated in studies of human learning (see [8] – [11]). In addition, with appropriate values of α and β the distribution quite effectively represents target intervals we wish the robot to learn. For instance, with $\alpha = \beta = 1$ the distribution is the uniform distribution on (0, 1) and represents the intervals described by phrases such as *anywhere on the shelf* or *anywhere in front of the desk.* (The (0,1) interval must, of course, be mapped onto the actual interval of interest for the task — that corresponding to the length of the shelf or the width of the desk, for instance.) The uniform distribution is also generally used to represent the state of the robot before all instruction starts. Figure 2 shows various curves for different choices of α and β along with the natural-language expressions describing the intervals associated with the curves. The response variable is plotted along the x-axis, the probability distribution along the y-axis.



the righthand side *anywhere but preferably at either end* *anywhere around the middle*

Figure 2: The representation and description of target intervals

A computationally efficient way of altering the initial distribution and all subsequent distributions is to change the values of α and β directly. As long as the ratio $r = \alpha/\beta$ remains constant, the mean of the distribution does not change. If $r = 1$ (i.e., $\alpha = \beta$) then the distribution is symmetrical around the midpoint of the (0, 1) interval. If $r > 1$ (i.e., $\alpha > \beta$), the distribution is shifted to the right. If $r < 1$ (i.e., $\alpha < \beta$), the distribution is shifted to the left. See Figure 3. To shift a distribution to the right, therefore, we increase r. To shift it to the left, we decrease r.

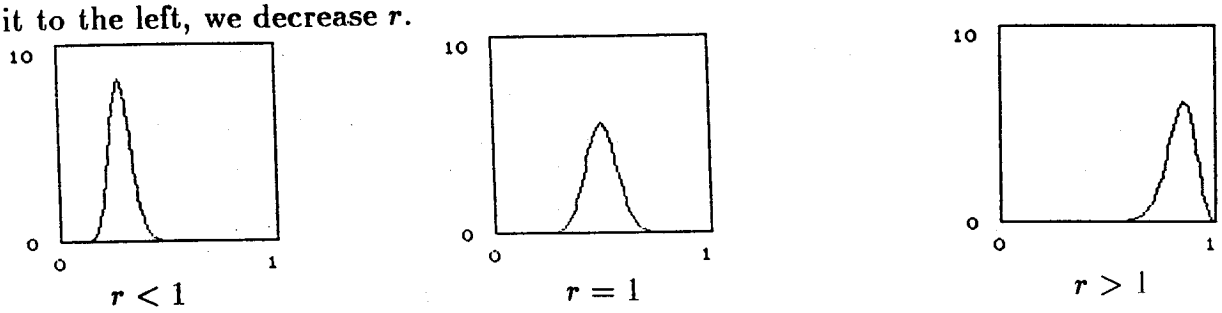


Figure 3: Adjusting the distribution to the left and right

If the ratio of α to β is kept fixed, increasing α (and β correspondingly) reduces the variance without altering the mean. Reducing the variance has the effect of increasing the accuracy of the robot's subsequent responses to positional feedback. It further has the effect (for $\alpha, \beta > 1$) of reducing the target interval. See Figure 4 for examples in which $r = 2$. The mean is indicated by a vertical bar.

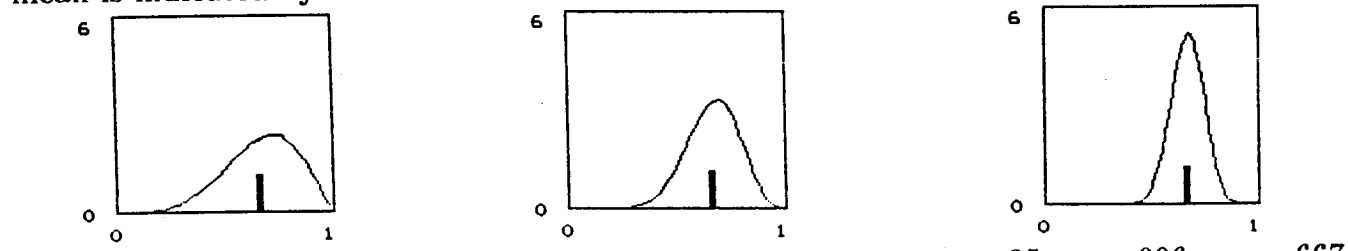


Figure 4: Adjusting the variance of the distribution

4. Interpretation of commands: the robot's motor response

As the preceding discussion has shown, the beta distribution can be altered through direct manipulation of the parameters α and β. Before we can describe in detail how these changes are brought about, however, we must discuss the link between the probability distribution and the interpretation of the natural-language commands. We will not discuss in detail the process by which commands are interpreted by the robot nor the control mechanisms that enable the robot to move in response to a command. Our earlier publications describe this work in some detail as implemented for the robotic aid, a device being developed for people with severe physical disabilities by the Rehabilitation and Research Center of the Veterans Administration in Palo Alto ([2], [7]). More general discussions of our work on natural-language understanding for robots can also be found in other publications (see [1], [3] – [6]).

Given limited space, our primary concern here is to show how a probability distribution enters into the interpretation of a command, specifically, the role it plays in generating the robot's motor response. For simplicity, we present highly schematic interpretations of sample commands. We use as our example of a command whose meaning must be learned by the robot through instruction (the original command in an instruction dialogue) *Put the wrench on the shelf!*

Original command: Suppose *Put the wrench on the shelf!* is translated into the following robot plan: SEQ(Grasp-Free(wrench), Move-Gripper(xval,yval,zval), Place(wrench)). SEQ indicates that the robot actions Grasp-Free, Move-Gripper, and Place are executed in sequence. The point in 3-dimensional space that the robot must move its gripper to is given by the triple <xval, yval, zval >. Values for yval and zval are obtained from the robot's knowledge base (information that is provided either by the sensors or through earlier instruction). The value of xval, however, the point along the length of the shelf where the wrench must be placed, is obtained by sampling the $(0, 1)$ interval using the probability distribution that is currently associated with the command. Initially, this distribution is the uniform distribution and any point along the length of the shelf is as likely as any other to be selected. In general, if $k_{m_n,v_n}(x)$ is the current probability distribution (step n of the instruction dialogue has just been completed), a response to the original command at step $n+1$ is generated by the following sampling procedure: Take the cumulative probability distribution $K_n(x) = \int_0^x k_{m_n,v_n}(x)dx$, generate a random number y between 0 and 1, and find x such that $K_n(x) = y$. This x gives the robot's response at step $n+1$. We write $x_{n+1} = sample(k_{m_n,v_n})$.

Positional feedback: Suppose the interpretation of the command *Much further left!* is DO(Pilot(Translate, Left) UNTIL Distance-Covered(Translate, Left, <distance>)). This interpretation ensures that the gripper is moved left until the designated distance (indicated by <distance>) has been covered. DO ... UNTIL is one of the control structures, like SEQ described above, that control the temporal and logical order in which basic robot actions such as Pilot and Distance-Covered are executed. The Pilot action moves the gripper in the direction indicated by its second argument; the motion is either translational or rotational as indicated by its first argument. The actual distance covered depends on the current probability distribution in the following way. If the variance, v, of the distribution is small, a move leftward or rightward should be correspondingly small. If the variance is large, the move should be correspondingly large.

In addition, a correction such as *Much further left!* must generate a larger move than a correction such as *Left just a little!* Clearly, the adverbial, and sometimes the adjectival, component of the language must be allowed to make its contribution. Currently, all positional corrections fall into three equivalence classes: those that generate a relatively small move, those that generate a relatively large move, and those that generate a move of intermediate extent. Three range constants are therefore needed, c_L, c_M, and c_S for "large," "medium," and "small" moves respectively. These constants are set for the robot and the task (in our current instruction model, $c_L = 2$, $c_M = 1$, and $c_S = .5$). The total displacement

for a move is given by the product of the appropriate range constant and the square root of the variance. A request for a "large" move therefore generates a displacement of $c_L\sqrt{v}$. The robot must never move outside the $(0, 1)$ interval; extreme leftmost and rightmost points (*minx* and *maxx* respectively) are therefore designated. They are set for the robot and the task. It is important to note that although the response to positional feedback is not generated by a sampling procedure on the underlying probability distribution, the actual extent of the move depends on that distribution.

Accuracy feedback and congratulatory feedback: The interpretation of a command giving accuracy feedback during instruction does not generate any motor response from the robot. Nor does the interpretation of a command giving congratulatory feedback. Both forms of feedback change the probability distribution, however.

5. Effect of feedback on the probability distribution

Positional feedback: For positional feedback, we want to shift the distribution to the left or the right. As indicated earlier in our discussion of the beta distribution, we can accomplish this shift by adjusting the ratio r, increasing it for a shift to the right and decreasing it for a shift to the left. As before, the adjustments fall into three equivalence classes, large, medium, and small. The same three range constants (c_L, c_M, and c_S) are used to determine the amount of increase or decrease in r. Keeping α fixed, we decrease (or increase) β by the amount necessary to increase (or decrease) r by the appropriate range constant.

Accuracy feedback: For feedback such as *Be more careful!* which asks for greater accuracy in the robot's responses, we decrease the variance of the current distribution. For feedback such as *No need to be so cautious!* which encourages the robot to make larger adjustments, we increase the variance. As indicated by our earlier discussion of the beta distribution, the variance can be increased or decreased by increasing or decreasing α. The amount of increase or decrease should be in proportion to the value of α itself. We determine the appropriate amount of increase or decrease from the slope of the tangent to the variance, $f_\beta(\alpha) = (\alpha\beta)/((\alpha+\beta)^2(\alpha+\beta+1))$, at α. Specifically, if we are at step $n+1$ of the instruction dialogue ($\alpha = \alpha_n$, $\beta = \beta_n$), we increase or decrease α by the square root of the absolute value of the derivative of the variance (with respect to α) evaluated at α_n, i.e., the square root of the absolute value of $f'_{\beta_n}(\alpha_n)$. The ratio of α to β is kept constant to preserve the mean. We therefore alter β correspondingly by setting β_{n+1} to α_{n+1}/r_n.

Congratulatory feedback: For congratulatory feedback, the mean of the distribution shifts to the robot's response x on that trial and the variance is halved. A new probability distribution is produced by solving for α and β given this new mean and new variance.

Each kind of physical robot will have its own performance constraints — both accuracy limits and, in the case of a manipulator, limits of reach. We have already introduced explicit range constants to determine the extent of a move to the left or right. Other performance

constraints are reflected in maximum and minimum values for α and β. In our current model we have set $max\alpha\beta = 30$ and $min\alpha\beta = .25$. As with the range constants, these constants are set for the robot and the task to get a proper balance between sensitivity to feedback and realistic performance demands. The computation that increases or decreases r for positional feedback at step $n+1$ takes $max\alpha\beta$ and $min\alpha\beta$ into account with the constraint that $\alpha_n/max\alpha\beta < r_{n+1} < \alpha_n/min\alpha\beta$. The computation that alters α and β for accuracy feedback also takes $max\alpha\beta$ and $min\alpha\beta$ into account. To ensure that α_{n+1} and β_{n+1} are both greater than $min\alpha\beta$ and smaller than $max\alpha\beta$, if α is being increased and $r_n > 1$, it obeys the constraint that $r_n min\alpha\beta < \alpha_{n+1} < max\alpha\beta$. Similar constraints hold for $r_n < 1$, and when α is being decreased. In response to congratulatory feedback, α and β are permitted to exceed $max\alpha\beta$ up to a current maximum of 100 (once again, a constant set for the robot and the task). At that time $max\alpha\beta$ is reset to the new α or β, whichever is greater.

We note here that we are assuming congratulatory feedback to be thoroughly effective in that the mean and variance of the distribution always change in response to the feedback. It is common practice to introduce a learning parameter θ into the model and to assume that with probability θ feedback is effective on any trial (the distribution changes), and with probability $(1-\theta)$ it is not effective (the distribution stays the same). We have in effect set θ to 1 in our instruction model. We made this choice in part because of the explicitness of verbal instructions — congratulatory feedback, in particular, is hard to misconstrue — but also because we are assuming that the robot's total cognitive resources are dedicated to learning from instruction. This assumption would change if the robot were simultaneously attending to some other task. For example, in a space application the robot might be monitoring the status of a process for an emergency condition to which it had to respond.

Table 1 summarizes the core computations in our current instruction model using representative commands. It assumes that $k_{m_n,v_n}(x)$ is the current probability distribution, with parameters α_n and β_n. The mean of the distribution is m_n, the variance is v_n, $r_n = \alpha_n/\beta_n$, x_n is the robot's most recent motor response, $f'_{\beta_n}(\alpha_n)$ is the derivative of the variance with respect to α at α_n, and c_S, c_L, $max\alpha\beta$, $min\alpha\beta$, $maxx$, and $minx$ are all performance constants as described earlier.

Response and feedback	Distribution changes and robot's motor response
A HIT:	
Good!	$m_{n+1} \leftarrow x_n$ $v_{n+1} \leftarrow v_n/2$
A MISS:	
Positional Feedback:	
A little to the left!	$r_{n+1} \leftarrow \begin{cases} r_n - c_S & \text{if } r_n - c_S > \alpha_n/max\alpha\beta \\ \alpha_n/max\alpha\beta & \text{otherwise} \end{cases}$
	$x_{n+1} \leftarrow \begin{cases} x_n - c_S\sqrt{v_n} & \text{if } x_n - c_S\sqrt{v_n} > minx \\ minx & \text{otherwise} \end{cases}$

Much further right!

$$r_{n+1} \leftarrow \begin{cases} r_n + c_L & \text{if } r_n + c_L < \alpha_n / min\alpha\beta \\ \alpha_n / min\alpha\beta & \text{otherwise} \end{cases}$$

$$x_{n+1} \leftarrow \begin{cases} x_n + c_L\sqrt{v_n} & \text{if } x_n + c_L\sqrt{v_n} < maxx \\ maxx & \text{otherwise} \end{cases}$$

Accuracy Feedback:

Be more careful!

$$\alpha' \leftarrow \alpha_n + \sqrt{\left| f'_{\beta_n}(\alpha_n) \right|}$$

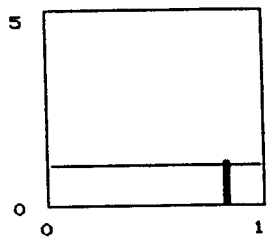
$$\alpha_{n+1} \leftarrow \begin{cases} r_n min\alpha\beta & \text{if } \alpha' < r_n min\alpha\beta \\ r_n max\alpha\beta & \text{if } \alpha' > r_n max\alpha\beta \text{ and } r_n < 1 \\ max\alpha\beta & \text{if } \alpha' > max\alpha\beta \text{ and } r_n > 1 \\ \alpha' & \text{otherwise} \end{cases}$$

$$\beta_{n+1} \leftarrow \alpha_{n+1} / r_n$$

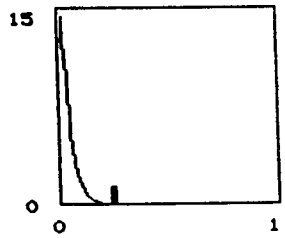
Table 1: Core computations in the 1-dimensional model

6. Sample instruction sessions

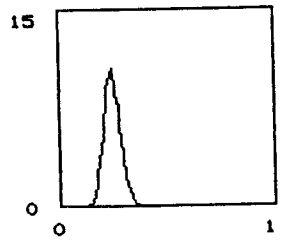
We now show two sample instruction sessions. The 1-dimensional skill being taught to the robot is where to stop in front of double swing doors so that the robot can enter through the righthand door when it opens. The doors swing out; the robot is on the outside. Taking the combined width of the two doors as the interval of interest, we want to teach the robot to select any point somewhat to the left of the midpoint. The robot begins in each case with the beta distribution initialized to $\alpha = \beta = 1$. We show at each step what was said, what the robot's motor response was (if there was indeed a response), and what the resulting distribution looks like. The robot's response is indicated by a vertical bar. The response variable is plotted along the x-axis, the probability distribution along the y-axis. The first instruction dialogue was quickly successful (only three steps), although in subsequent instruction the operator may wish to shift the responses nearer to the middle. The second dialogue lasted longer (nine steps). The y-axis on each graph goes from 0 to 5 or the least integer greater than the maximum distribution value plotted.



1. Go to the door!



That's much too far right !



That's fine!

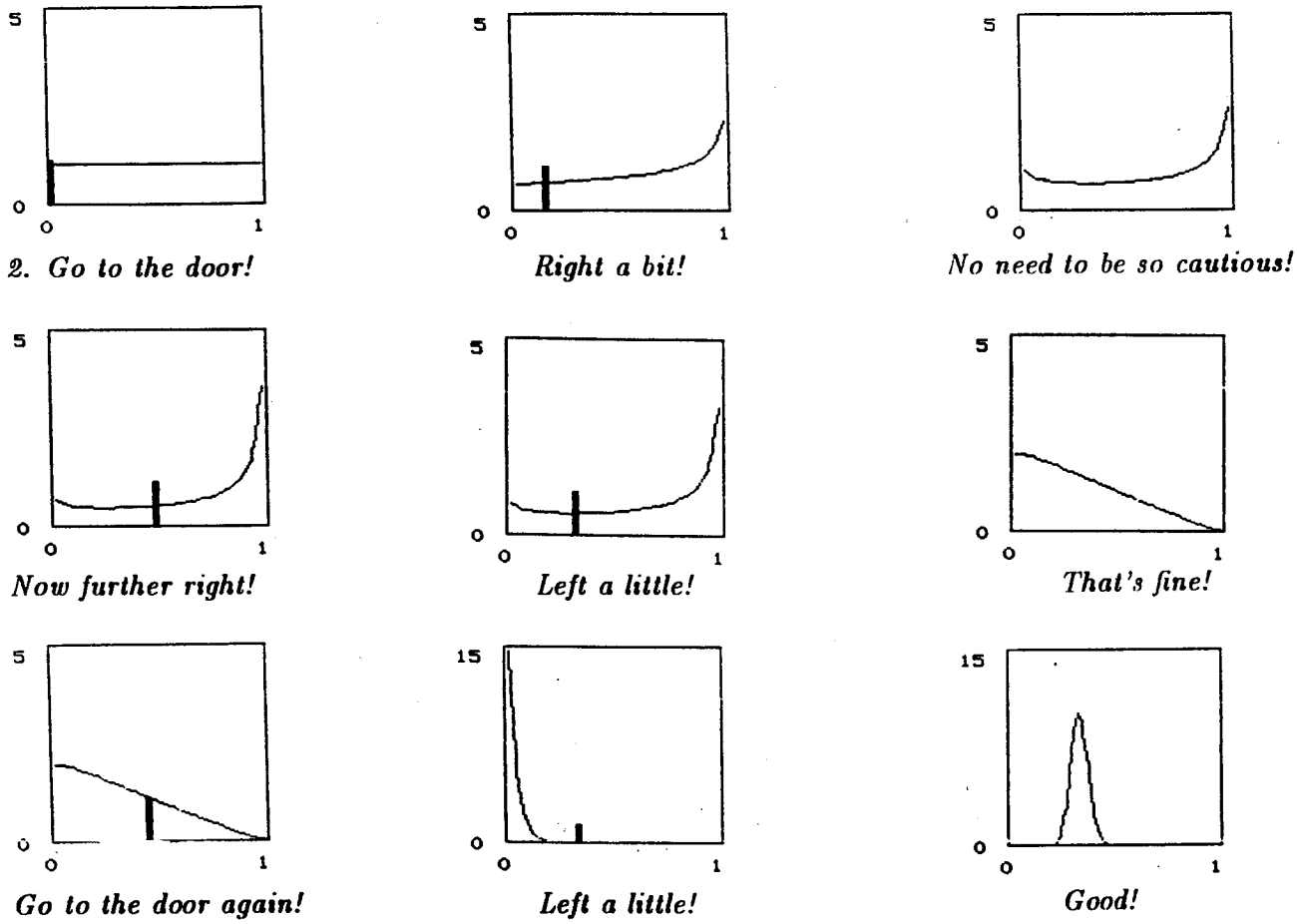


Figure 5: Two sample instruction sessions

Note the somewhat surprising shape of the distribution after the first trial in the second dialogue. This is due to the particular form of learning, i.e., the rule for changing the beta distribution, we have adopted. We are in the process of exploring and testing other learning rules for comparison.

7. Concluding remarks

The learning procedure we have described is congenial to a Bayesian viewpoint, but embodies a number of specific learning assumptions that go beyond a purely Bayesian framework. For example, it is not part of Bayesian theory to postulate how specific words of feedback should change the underlying response probability distribution used by the robot. Such interpretation is more in the spirit of mathematical learning theory (see [8] - [11]).

Directions for future work are clear. First, we must make extensions to the model for 2-dimensional and 3-dimensional tasks and for the dimension of time. We plan to move to 2-dimensional tasks immediately. The computation problems are more severe for probability distributions whose domains are arbitrary surfaces. Our initial effort will use some strong simplifying assumptions. First, we use as the 2-dimensional distribution the product of two beta distributions, which assumes we can assume for working purposes independence in

probability along the two coordinate axes. Second, we fit the shape of the surface of interest by conditionalizing the 2-dimensional distribution to the given surface as domain. The rules for interpreting English instructions will inevitably be more numerous and complex. The instructions themselves will certainly be more complex, for example, *To the left and up!, A good deal further back but not so far left!* Our ultimate objective is to have a substantial grammar of English that generates the rules of learning for these higher-dimensional tasks.

We also plan to implement the learning models on an actual robot system. We expect this implementation to greatly expand the range of feedback expressions we consider, and to provide a realistic test for the performance constants discussed earlier. An important overall change we plan for the models is to allow feedback to be given, and to take effect, while the robot is still responding to the previous command. So, for instance, if the robot is moving forward in response to a command from the operator, the operator should be able to say *That's far enough!* and have the robot stop where it is. Not only is this form of feedback entirely natural, it makes for safer robot operation and it should provide faster learning.

References

[1] Crangle, C. (in press). On saying *Stop* to a robot. *Language and Communication, 9, No. 1.*

[2] Crangle, C., Liang, L., Suppes, P, & Barlow, M. (1988). Using English to instruct a robotic aid: an experiment in an office-like setting. In *Proceedings of the International Conference for the Advancement of Rehabilitation Technology, 25-30 June, 1988, Montreal.* (pp. 466-7).

[3] Crangle, C., & Suppes, P. (in press). Geometrical semantics for spatial prepositions. In P. French, T. Uehling, & H. Wettstein (Eds.) *Midwest Studies in Philosophy Vol. 13, Contemporary Perspectives in the Philosophy of Language II.*

[4] Crangle, C., & Suppes, P. (1987). Context-fixing semantics for an instructable robot. *International Journal of Man-Machine Studies, 27,* 371-400.

[5] Crangle, C., Suppes, P., & Michalowski, S. (1987). Types of verbal interaction with instructable robots. In G. Rodriguez. (Ed.) *Proceedings of the Workshop on Space Telerobotics,* (JPL Publication 87-13, Vol. II). (pp. 393-402). Pasadena, California: NASA Jet Propulsion Laboratory.

[6] Maas, R.E., & Suppes, P. (1985). Natural-language interface for an instructable robot. *International Journal of Man-Machine Studies, 22,* 215-240.

[7] Michalowski, S., Crangle, C., & Liang, L. (1987). A natural-language interface to a mobile robot. In G. Rodriguez. (Ed.) *Proceedings of the Workshop on Space Telerobotics,* (JPL Publication 87-13, Vol. II). (pp. 381-392). Pasadena, California: NASA Jet Propulsion Laboratory.

[8] Suppes, P. (1959). A linear model for a continuum of responses. Reprinted in R.R. Bush and W.K. Estes (Eds.) *Studies in Mathematical Learning Theory.* Stanford: Stanford University Press. Pp. 400-414.

[9] Suppes, P. (1964). Some current developments in models of learning for a continuum of responses. (Discrete Adaptive Processes Symposium, American Institute of Electrical Engineers, June 1962.) *The Institute of Electrical and Electronics Engineers Transactions on Applications and Industry, 83,* 297-305.

[10] Suppes, P. & Zinnes, J.L. (1966). A continuous-response task with nondeterminate, contingent reinforcement. *Journal of Mathematical Psychology, Vol. 3, No. 1,* 197-216.

[11] Suppes, P. (in press). Current directions in mathematical learning theory. In E. Degreef and E. Roskam (Eds.) *Progress in Mathematical Psychology Vol. II.*