Towards Quantitative Modeling of Task Confirmations in Human-Robot Dialog

Junaed Sattar and Gregory Dudek

***Abstract*— We present a technique for robust human-robot interaction taking into consideration uncertainty in input and task execution costs incurred by the robot. Specifically, this research aims to quantitatively model confirmation feedback, as required by a robot while communicating with a human operator to perform a particular task. Our goal is to model human-robot interaction from the perspective of risk minimization, taking into account errors in communication, risk involved in performing the required task, and task execution costs. Given an input modality with non-trivial uncertainty, we calculate the cost associated with performing the task specified by the user, and if deemed necessary, ask the user for confirmation. The estimated task cost and the uncertainty measure is given as input to a *Decision Function*, the output of which is then used to decide whether to execute the task, or request clarification from the user. In cases where the cost or uncertainty (or both) is estimated to be exceedingly high by the system, task execution is deferred until a significant reduction in the output of the Decision Function is achieved. We test our system through human-interface experiments, based on a framework custom designed for our family of amphibious robots, and demonstrate the utility of the framework in the presence of large task costs and uncertainties. We also present qualitative results of our algorithm from field trials of our robots in both open– and closed–water environments.**

I. Introduction

When a human gives instructions to a robot using a "natural" interface, communication errors are often present. For some activities the implications of such errors are trivial, while for others there may be potentially severe consequences. In this paper, we consider how such errors can be considered explicitly in the context of risk minimization. While fully autonomous behaviors remain the ultimate goal for robotics research, there will always be a prevailing need for robots to act as assistants to humans. As such, we focus on the interim, on a control regime between full teleoperation and complete autonomy, where a semi-autonomous robot acts as an assistant to a human operator and a robust interaction mechanism exists between the two. In particular, whereas many robotic systems operate using only imperative commands, we wish to enable the system to engage in a dialog with the user.

This research is a natural extension of previous work on visual languages for robot control and programming [1], which has been successfully used to operate the Aqua2 family of underwater robots [2]. In that work, divers communicate with the robot visually using a set of fiducial markers, by forming discrete geometric gestures with a pair of such markers. While this fiducial-based visual control language, *RoboChat*, has proven to be robust and accurate, we do not have any quantitative measure of uncertainty or cost assessment related to the tasks at hand. The framework we propose here is designed to be an adjunct to a language such as RoboChat and provide a measure of uncertainty in the utterances. Moreover, by providing additional robustness (*e.g.* through uncertainty reduction and ensuring robot safety) as a result of the dialog mechanism itself, a reduced level of performance is required from the base communication system allowing for more flexible alternative mechanisms.

Any interaction protocol will carry a certain degree of uncertainty with it and for accurate human-robot communication, that uncertainty must be incorporated and accounted for by a command-execution interface. In the presence of high uncertainty, large degree of risk, or moderate uncertainty coupled with substantial risk, the robot should ask for confirmation. The principled basis for this decision to ask for confirmation is our concern.

Our current work has been developed for application specifically, but not exclusively, to the domain of underwater robotics. In this context, the robot operates in concert with a human and the primary risk factors are measured as a function of the difficulties incurred if the robotic system fails, and as a function of the total length of an experiment. The longer the diver has to stay underwater the less desirable a situation it is. In addition, if the robot fails far from the diver it is much more serious than if it fails nearby. Finally, if the robot travels far away, it is intrinsically more dangerous due to reduced visibility, current and other factors. Thus, risk is primarily described in terms of risk to the human operator from a more extensive experiment, and risk to the human and the robot as a result of being separated during a procedure or as a result of a failure during the execution of a task.

The work described in this paper focuses on two principal ideas: uncertainty in the input language used for human-robot interaction, and analysis of cost of the task. We present a theoretical framework for initiating dialogs between a robot and a human operator using a model for task costs and a model of uncertainty in the input scheme. A *Decision Function* takes as input both these parameters, and based on the output of this function, the system prompts the user for feedback (*e.g* in the form of confirmation of the commands), or executes the given command. The cost assessment is a combination of an *external* cost in the form of operational risk, and an *internal* cost expressed in terms of operational overhead.

Junaed Sattar and Gregory Dudek are with the School of Computer Science, McGill University, 3480 University Street, Montréal, QC, Canada H3A 2A7. {junaed,dudek}@cim.mcgill.ca

II. Background and Related Work

This work uses a gesture-like interface to accomplish human-robot interaction, and this is somewhat related to visual programming languages. The inference process we use is based on a Markovian dialog model. Hence, we briefly comment on prior work, necessarily in a rather cursory manner, in each of these disparate and rich domains. As this particular research builds on our past work in vision-based human-robot interaction, and we briefly revisit those in this section as well.

Sattar *et al*. looked at using visual communications, and specifically visual servo-control with respect to a human operator, to handle the navigation of an underwater robot [3]. In that work, while the robot follows a diver to maneuver, the diver can only modulate the robot's activities by making hand signals that are interpreted by a human operator on the surface. Application of that work where robot control was purely "open-loop" motivate this paper. Visual communication has also been used by several authors to allow communication between systems, for example in the work of Dunbabin *et al*. [4]

The work of Waldherr, Romero and Thrun [5] exemplifies the explicit communication paradigm in which hand gestures are used to interact with a robot and lead it through an environment. Tsotsos *et al*. [6] considered a gestural interface for non-expert users, in particular disabled children, based on a combination of stereo vision and keyboard-like input. As an example of implicit communication, Rybski and Voyles [7] developed a system whereby a robot could observe a human performing a task and learn about the environment.

Fiducial marker systems, as mentioned in the previous section, are efficiently and robustly detectable under difficult conditions. Apart from the ARTag toolkit mentioned previously, other fiducial marker systems have been developed for use in a variety of applications. The ARToolkit marker system [8] consists of symbols very similar to the ARTag flavor in that they contain different patterns enclosed within a square black border. Circular markers are also possible in fiducial schemes, as demonstrated by the Fourier Tags [9] fiducial system.

Gesture-based robot control has been considered extensively in Human-Robot Interaction (HRI). This includes explicit as well as implicit communication frameworks between human operators and robotics systems. Several authors have considered specialized gestural behaviors [10] or strokes on a touch screen to control basic robot navigation. Skubic *et al.* have examined the combination of several types of human interface components, with special emphasis on speech, to express spatial relationships and spatial navigation tasks [11].

Vision-based gesture recognition has long been considered for a variety of tasks, and has proven to be a challenging problem examined for over 20 years with diverse well-established applications [12][13]. The types of gestural vocabularies range from extremely simple actions, like simple fist versus open hand, to very complex languages, such as the American Sign Language (ASL). ASL allows for the expression of substantial affect and individual variation, making it exceedingly difficult to deal with in its complete form. For example, Tsotsos et al.[14] considered the interpretation of elementary ASL primitives (i.e. simple component motions) and achieved 86 to 97 *per cent* recognition rates under controlled conditions. While such rates are good, they are disturbingly low for open-loop robot-control purposes.

While our current work looks at interaction under uncertainty in any input modality, researchers have investigated uncertainty modeling in human-robot communication with specific input methods. For example, Pateras *et al*. applied fuzzy logic to reduce uncertainty to reduce high-level task descriptions into robot sensor-specific commands in a spoken-dialog HRI model [15]. Montemerlo *et al*. have investigated risk function for safer navigation and environmental sampling for the Nursebot robotic nurse in the care of the elderly [16]. Bayesian risk estimates and active learning in POMDP formulations in a limited-interaction dialog model [17] and spoken language interaction models [18] have also been investigated in the past. Researchers have also applied planning cost models for efficient human-robot interaction tasks [19] [20].

III. Methodology

In a typical human–robot interaction scenario, the human operator instructs the robot to perform a task. Traditionally this takes place using a pragmatic interface (such as keyboards or mice), but the term "human-robot interaction" usually implies more "natural" modalities such as speech, hand gestures or physical body movements. Our approach is, in principle, independent of the specific modality, but our experimental validation described later in the paper uses gestures. The essence of our approach is to execute costly activities only if we are *certain* they have been indicated. For actions that have low cost, we are willing to execute them even when the level of certainty is low, since little is lost if they are executed inappropriately.

Whatever the modality, the robot has to determine the instructions and for most natural interfaces this entails a substantial degree of uncertainty. The interaction starts with the human operator providing input *utterances* to the robot. The robot estimates a set of actions and generates a plan (in this case a potential trajectory) needed to perform the given task using a simulator. The generated action and trajectory is then evaluated by a cost and risk analysis module, comprised of a set of *Assessors*. This module outputs estimated total cost and together with the uncertainty in the input dialog, is fed into a *Decision Function*. If the relationship between cost and uncertainty is unacceptable then the robot decides to ask for feedback. Otherwise, the robot executes the instructed task. A flowchart illustrating the control flow in this process can be seen in Figure 1.

The core of our approach relies on calculating a probabilistic measure of the uncertainty in the input language, and also calculating the cost involved in making the robot perform the task as instructed by the human operator. The

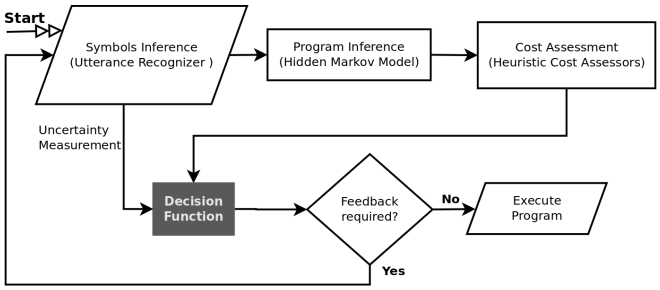


Fig. 1. Control flow in our risk-uncertainty model.

following two subsections describe in detail these two aspects of our framework.

A. Uncertainty Modeling

To interact with a mobile robot, a human operator has to use a mode of communication, such as speech, gestures, touch interface, or mouse input on a computer screen. In practice, there will almost always be noise in the system that will introduce uncertainty in the dialog.

In our human-robot interaction framework, utterances are considered to be inputs to the system. *Gestures*, g_i, are symbols containing specific instructions to the robot. A gesture set, G, is made up of a finite number of gestures, and is thus expressed as

$$G = g_1, g_2, \cdots g_n \tag{1}$$

Each gesture g_i has associated with it a probability $p(g_i)$ of being in an utterance. The robot is aware of the gesture set G (*i.e.* the "language" set), and the probabilities $p_i(g)$ are precomputed and is available to the system before interaction begins with a human operator. This notion of discrete symbols combined with probabilities is commonplace in speech understanding.

Statements, S, (*i.e.* sentences) in our framework can be atomic gestures (*e.g.* "go left", or "take picture"), or they can be compound commands constituted of several atomic gestures, including repetitions. We assume each gesture is independently uttered by the operator, and thus the probability of a sequence of gestures chained together to form a compound statement simply becomes:

$$p(S) \equiv p(g_1, g_2, \ldots g_n) = \prod_{i=1}^{n} p(g_i) \tag{2}$$

Given a table of values for all $p(g_i)$, it is trivial to compute the probability of any given sequence of gestures.

Programs or *Tasks* are either a smaller subset of or equal to a set of statements. By definition, we indicate programs to contain only consistent instructions (*i.e.* instructions that are illegal in semantics or syntax). This implies that programs, P

$$|P| \supseteq |S| \tag{3}$$

Each program P_i has a likelihood of occurrence l_i and a cost c_i associated with it. It is worth noting however, given the input language, the set of all possible programs will be reduced, as the inconsistent ones, both syntactically and semantically, are going to be expunged.

Since there will always exist uncertainty in input observation, we can model the input language scheme as a Hidden Markov Model [21], with the actual input gestures becoming the hidden states of the HMM. An HMM requires three probability matrices to be specified to estimate the input utterances, namely:

1) Initial probabilities of the hidden states, $\mathbf{\Pi}$.
2) Transition probabilities between the hidden states, $\mathbf{A}$.
3) Confusion matrix, or the emission probabilities, $\mathbf{B}$.

For any given input mechanism, we assume the matrices can be estimated or learned. Once the matrices are available, the Baum-Welch algorithm can be used to train the HMM parameters and the Viterbi algorithm can be applied to estimate the likelihood of the input utterance [22].

B. Cost Analysis

Once the uncertainty is computed, we perform a cost assessment of the given task. This is performed irrespective of the uncertainty; *i.e.* a low uncertainty measure will not cause the cost calculation task to be suspended. To estimate the cost of running a program, we use a set of *Assessors*, that are applied on the robot state as the task is simulated. After executing each command in the input statement, the set of assessors examine the current state of the robot and produce an estimated value of risk. At the end of the simulation the overall program cost is a sum of all the assessor's outputs over the duration of the simulated program. This sum is eventually taken into consideration by the Decision Function.

We approach the cost factor from two different perspectives; namely the risk associated from the operator's perspective, and the cost involved in terms of operational overhead of the robot while attempting to perform the task. In conventional dialog models used for confirmation only, Bayes risk is often applied [23], where the system only confirms in order to avoid error. Nevertheless, there are often scenarios where the system should ask for confirmation even in high-confidence programs because the executing the task will place the underlying system in a high-risk state. This implies that Bayes risk cannot be used naively.

1) Risk Measurement: Risk encompasses many factors, including domain specific ones. In our case, the risk model reflects the difficulty of recovering the robot in the event of a total systems failure. In addition, the level of risk to the human operators is a function of time. Examples of high risk scenarios could be the robot venturing too far from safety, or drifting too close to the obstacles, or other objects that pose significant threat to the robot, requiring excessive time to perform the task, etc. We denote the set of such factors by $A = \{\alpha_1, \alpha_2, \ldots \alpha_n\}$. The examples presented here are by no means exhaustive, but only serve to demonstrate a possible set of parameters that can be included for measuring risk.

2) Cost Measurement: This component measures the operational overhead associated with robot operation, over the duration of the task to be executed. The overhead measures are a function of factors such as power consumption, battery condition, system temperature, computational load, total distance travelled etc. We denote the set of such factors by $B = \{\beta_1, \beta_2, \ldots \beta_n\}$. Note that the exact measurement of these factors is not possible until the task at hand is completed, hence the initial values obtained are estimates based on simulation or past system operational benchmarks. One can apply machine learning methods, supervised learning in particular, to learn a model of system overhead, although in this work we do not enforce any particular model.

3) Decision Function: Let f be the risk measurement function, and φ denote the overhead cost measurement function. Then, overall operational cost, C becomes,

$$C = f(\alpha_1, \alpha_2, \ldots, \alpha_n) + \varphi(\beta_1, \beta_2, \ldots, \beta_n) \tag{4}$$

If we denote the uncertainty measure as P, the *Decision Function* ρ can be expressed as,

$$\tau = \rho(C, P) \tag{5}$$

The function ρ increases proportionally with the cost measure C and is inversely proportional with P. If τ exceeds a given threshold, the system prompts the user for clarification, and the feedback is passed through the uncertainty model and cost estimation process in a similar fashion. Until the τ falls below a threshold, the system will keep asking the user to provide feedback. To estimate the threshold τ, we introduce the concept of the *Confirmation Space*.

C. Confirmation Space

Before executing a program with high cost or low likelihood, a robot should confirm the desirability of the task with the user. This ensures that the task is truly requested by the user and is not erroneously misinterpreted by the robot. Since asking for feedback from the user is itself not a cost-free task, any HRI system should ideally want to minimize the number of confirmation requests. There are three possible alternatives to choose from, namely,

1) Pick the safest (*i.e* lowest cost) program and execute it.
2) Pick the program with the highest likelihood, ignoring the task cost, and execute it.
3) Pick a combination of the two above, combining high task likelihood with low cost and execute it.

Clearly, considering cost without regard for likelihood and vice-versa would be foolhardy, and thus we opt for option 3 above. We generate all possible consistent sentences based on the observed input (by using the confusion matrix B of gestures, g_i), and pass them through the HMM to obtain likely observation values. These sentences are also passed through the task simulator (*i.e.* set of assessors) to evaluate the cost measures for all of the sentences. When the inverse of the cost values (*i.e* safety) are plotted against the observation likelihoods of these sentences, we obtain the *Safety-Likelihood Graph*, as illustrated in the example plot in Figure 2.

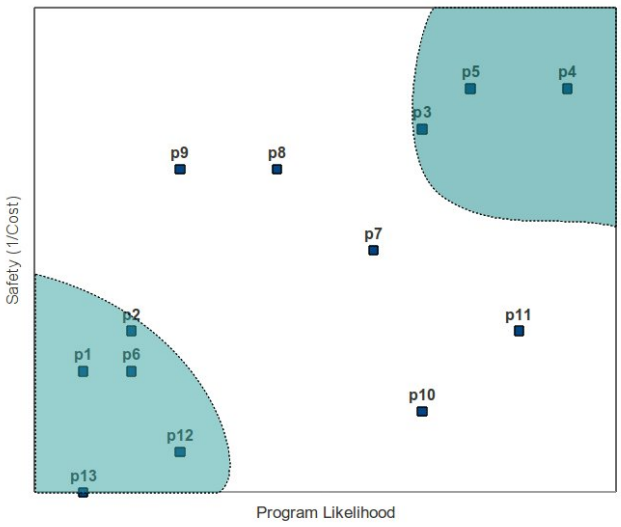


Fig. 2. A pictorial depiction of programs P_i in the Safety-Likelihood graph. Programs in the non-shaded areas are in the Confirmation Space.

The shaded areas at the extremities indicate regions where tasks have high certainty and high safety (upper right), and low likelihood and low safety (lower left). Tasks that fall outside these areas are in-between these ranges, and are candidates for requiring confirmations. As such, we label this region as the *Confirmation Space* in the Safety-Likelihood graph.

Once the possible sentences have been generated and their corresponding likelihoods and costs have been computed, we take the average cost of these programs and set that as the value of threshold τ. Next, we pick the most likely program and compare its likelihood to that of the threshold. If it exceeds the threshold, we ask for confirmation. Otherwise, we execute the program as instructed.

IV. Experiments and Results

In order to validate our approach and quantify the performance of the proposed algorithm, we conducted a set of dialog-based experiments, both on-board and off-board. In the off-board experiments, a set of users were asked to program the robot to perform certain tasks, with an input modality that ensured a non-trivial amount of uncertainty in communication. Since the key concept in this work involves a human-robot dialog mechanism, we did not require task execution for the off-board trials. We performed field trials on-board on the Aqua2 underwater robot, both in open-water and closed-water environments, to qualitatively assess the feasibility of a real-world deployment of the system. Results and experiences from both sets of experiments are presented in the following sections, preceded by a brief description of the input language.

A. Language Description

The language used for programming the robot is designed to be easily deployable in a human-robot dialog context for the Aqua2 robot. For these experiments, we used a subset of the complete language. The language tokens (gestures) comprised of basic motion commands, commands for localizing and commands to track and follow an object of interest. The commands can be optionally followed by numeric arguments, which denote the number of seconds the commands should be executed for. In our experiments, the actual input argument was multiplied by three to prolong the execution time of the robot. One could use large number of tokens to address a large space of numeric arguments, but in theory that space is infinitely large, and a non-trivial subset of such tokens can impose a significant cognitive burden on the user. The commands are mostly self-explanatory (as seen in Tables I and II). The visual following task is a two-step process – the `TUNETRACKER` command instructs the robot to calibrate the vision system to follow the target directly in front of the robot; the `FOLLOW` command instructs the robot to actually start following the target of interest as it moves away. The numeric argument after `FOLLOW` is the duration for which the robot should follow the target. The system only starts to evaluate the input after it encounters an `EXECUTE` command. A common task in the underwater domain is that of surveillance and inspection. As such, the commands chosen for the trials instruct the robot to carry out such surveillance tasks in different trajectories.

B. User Study

We performed a set of user studies to collect quantitative performance measures of our algorithm. When operating as a diver's assistant in underwater environments, the system uses fiducials to engage in a dialog with the robot. However, in the off-board bench trials, we employed a simplified "gesture-only language", where the users were limited to using mouse input. We used a vocabulary set of 18 tokens defined by oriented mouse gestures, and as such each segment is bounded by a 20^o-wide arc. The choice for using mouse gestures stemmed from the need to introduce uncertainty in the input modality, while keeping the cognitive load roughly comparable to that experienced by scuba divers.

To calculate uncertainty in input, we trained a Hidden Markov Model using commonly used programs given to the robot (such as those used in previous experiments and field trials). To estimate task costs, we simulated the programs using a simulation engine and used a set of assessors that takes into account the operating context of an autonomous underwater vehicle. The simulator has been designed to take into account the robot's velocity, maneuverability and propulsion characteristics to accurately and realistically simulate trajectories taken by the robot while executing commands such as those used in our experiments.

In particular, we applied the following assessors during the user studies:

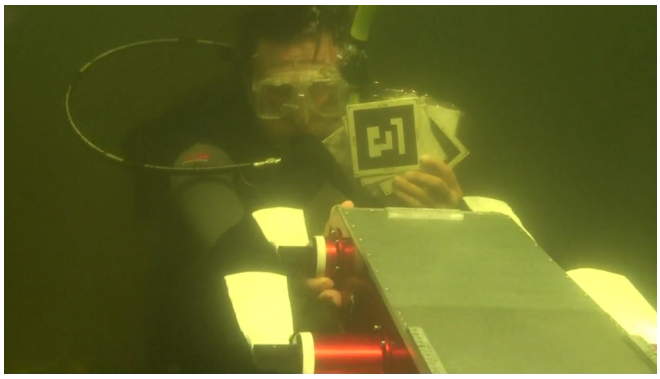
1) **Total distance**: The operating cost and risk factors both increase with total distance traveled by the robot. The cost associated with the amount of wear is a function of total travel, and higher travel distances also increase external operational risks.
2) **Farthest distance**: The farther the robot goes from the initial position (*i.e.* operator's position), the higher the chance of losing the robot. In the event that the robot encounters unusual circumstances which it is not equipped to handle, the involvement of a human operator is also a small possibility, thereby increasing the overall task cost.
3) **Execution Time**: An extremely long execution time also carries the overhead of elevated operational and external risk.
4) **Average Distance**: While the farthest and total distance metrics consider extremes in range and travel, respectively, the average distance looks at the distance of the robot (from start location) where most of the task execution time is spent.

Each user were given three programs to send to the system, and each program was performed three times. A total of 10 users participated in the trials, resulting in 30 trials for each program, and 90 in all; please refer to Table I for the programs used for the experiments, and whether confirmations were expected or not. Except for mistakes that created inconsistent programs, users did not receive any feedback about the correctness of their program. When a user finished "writing" a program, she either received feedback notifying her of program completion, or a confirmation dialog was generated based on the output of the Decision Function. The users were informed beforehand about the estimated cost of the program; *i.e.* whether to expect to receive a feedback or not. In case of a confirmation request for Programs 1 and 3, the users were instructed to redo the program. For Program 2, the users were informed of the approximate values of the outputs of the assessors. If the values shown in the confirmation request exceeded the expected numbers by 10%, the users required to reprogram it. Thus, in all cases, users required to conduct the programming task until the presence or absence of confirmation dialogs were consistent with the expected behavior. It is worth noting, however, that this does not necessarily indicate correctness of the programming, but merely indicates that the Decision Function has judged the input program (and likely alternatives of that) to be sufficiently inexpensive and thus safe for execution.

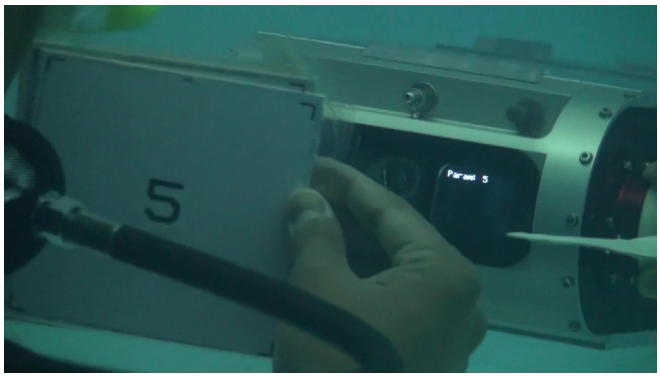
C. Field Trials

We performed field trials of our system on-board the Aqua2 underwater robot, in both open-water and closed-water environments. In both trials, the robot was visually programmed with the same language set used for the user studies, using ARTag [24] and ARToolkitPlus [8] fiducials used as input tokens; see Tab. II for the programs used in the field trials. The assessors used for the user studies were also used in the field trials; in addition, we provided an assessor to take into account the depth of the robot during task execution. Because of the inherent difficulty in operating underwater, the trials were not timed. Users were asked to do each

(a) Divers programming Aqua2 during pool trials.



(b) A diver programming Aqua2 during an HRI trial held at a lake in central Québec.



(c) Example of command acknowledgement given on the LED screen of the the Aqua2 robot during field trials.

Fig. 3. Field trials of the proposed algorithm on board the Aqua2 robot.

program once. Unlike in the user study, where there were no execution stage, the robot performed the tasks that it was programmed to do, when given positive confirmation to do so. In all experimental cases, the robot behaved consistently, asking confirmations when required, and executing tasks immediately when the tasks were inexpensive to perform. Unlike the user study, where the users had no feedback, the field trial participants were given feedback in the form of symbol acknowledgement using a LED display at the back of the robot (as seen on Figure 3(c)). Also unlike the user studies, the field trial users were given access to a command to delete the program and start from the beginning, in case they made a mistake. For a demonstration of our system

ID	Sequence	Confirm?
1	FORWARD, 3, PICTURE, LEFT, 3, PICTURE, UP, GPSFIX, GPSBEARING, EXECUTE	No
2	FORWARD, 9, LEFT, 6, FORWARD, 9, MOVIE, 9, RIGHT, 3, SURFACE, STOP, GPSFIX, EXECUTE	Yes
3	LEFT, 6, RIGHT, 3, MOVIE, 3, TUNETRACKER, FOLLOW, 6, UP, GPSFIX, EXECUTE	No

TABLE I

PROGRAMS USED IN THE USER STUDY.

ID	Sequence	Confirm?
1	FORWARD, 9, LEFT, 5, FORWARD, 9, LEFT 5, STOP, MOVIE, 9, EXECUTE	Yes
2	FORWARD, 5, LEFT, 3, FORWARD, 5, LEFT 3, FORWARD, 5, LEFT 3, STOP, EXECUTE	No
3	SWIMCIRCLE, 3, STOP, EXECUTE	No
4	SWIMCIRCLE, 3, FORWARD, 5, PICTURE, LEFT, 2, PICTURE, FORWARD, 3, PICTURE, RIGHT, 2, PICTURE, SURFACE, STOP, EXECUTE	Yes
5	TUNETRACKER, FOLLOW, 9, SURFACE, STOP, EXECUTE	Yes

TABLE II

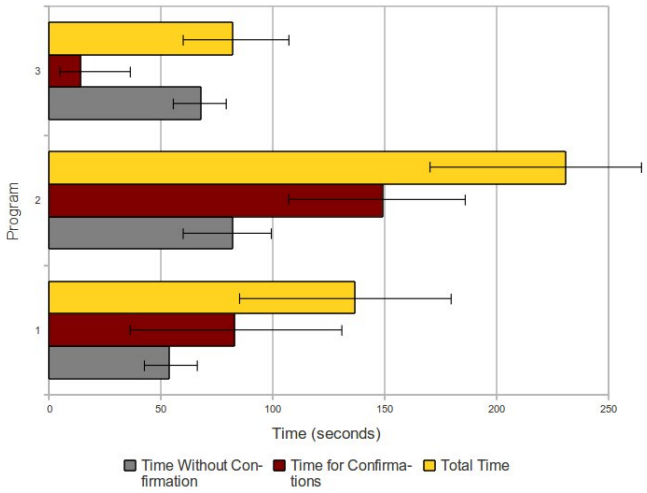
PROGRAMS USED IN THE FIELD TRIALS.

in action during field trials, we draw the reader's attention to the accompanying video clip, which demonstrates the visual programming mode for Aqua2, and task executions, including a target following mode. In case the tracked object is out of the field of view, the tracking algorithm we use allows the robot to attempt to reacquire the target, and this can be seen in the video clip. However, a detailed description of the target following and other robot behaviors are outside the scope of this paper, and thus will not be discussed further.

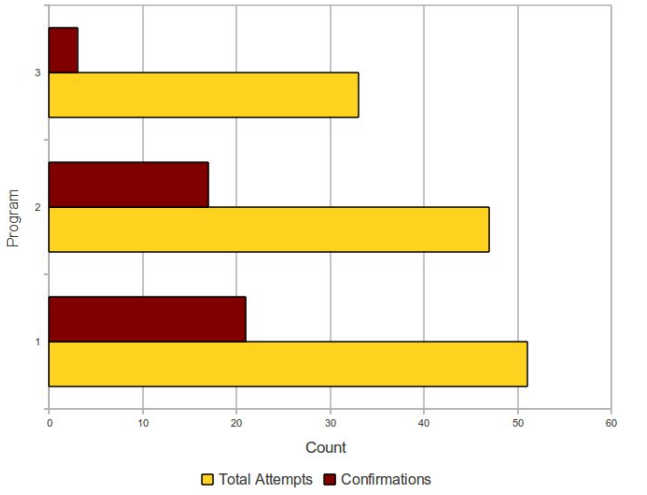
D. Results

From the user studies, it was observed that in cases where the programs were correctly entered, the system behaved consistently in terms of confirmation requests. Program 2 was the only one that issued confirmations, while Programs 1 and 3 only confirmed that the task would be executed as instructed. As mentioned in Sec. IV-B, the users were not given any feedback in terms of program correctness. Thus, the programs sent to the robot were not accurate in some trials; *i.e.* the input programs did not match exactly the programs listed in Tab. I. In case of mistakes, the Decision Function evaluated the input program and most likely alternatives, and only allowed a program to be executed (without confirmation) if and only if the task was evaluated to be less costly.

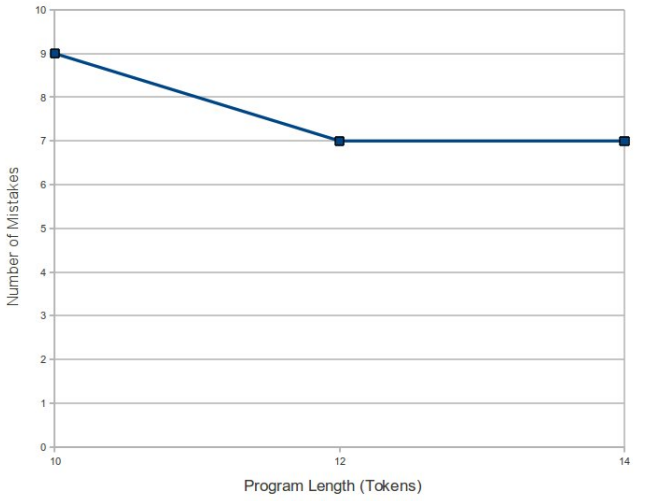
The cost of feedback, not unexpectedly, is the required time to program the robot. As seen in Figure 4(a), all three programs took more time to program on average with confirmations (top bar in each program group). From the user studies data, we see that the time cost is in the order of approximately 50% of the time required to program without any reconfirmations. Although the users paid a penalty in terms of programming time, the absence of safety checks meant a greater risk to the system and higher probability



(a) Programming times, all users combined.



(b) Programming attempts and generated confirmations, all users combined.



(c) Mistakes with respect to program length, all users combined.

Fig. 4. Results from user studies, timing 4(a), confirmations 4(b) and mistakes 4(c).

of task failures. This was illustrated in all cases where the system issued a confirmation request; an example of which is demonstrated in a trial of program 3 by user 2. The input to the system was given as "`LEFT` **`9`** `RIGHT 3 MOVIE 3` **`FOLLOW`** `FOLLOW` **`9`** `UP GPSFIX EXECUTE`", where the mistakes are in bold. The system took note of the change in duration from $\mathbf{6 \times 3 = 18}$ seconds to $\mathbf{9 \times 3 = 27}$ seconds on two occasions, but more importantly, the `FOLLOW` command was issued without a `TUNETRACKER` command. This, and the change in parameters to the higher values prompted the system to generate a confirmation request, which helped the user realize that mistakes were made in programming. A subsequent reprogramming fixed the mistakes and the task was successfully accepted without a confirmation. The distribution of confirmation requests and total number of attempts to program is shown in Figure 4(b).

One of the tangential issues in the study was the effect of long programming sequences on mistakes made by the users; *i.e.* whether more mistakes were made in longer programs. While this might seem like an obvious conclusion, we did not observe that behavior in the user trials, as demonstrated in Fig, 4(c). More detailed analysis and further experiments are required to obtain a definitive answer to this issue.

During the field trials, we were not able to collect quantitative data, but the system consistently generated confirmations based on the expensiveness of the task. In the underwater environment, where divers are cognitively loaded with maintaining dive gear and other life-support tasks, having feedback on input and the ability to start over proved to be especially important. These two features relieved some of the burden of programming, and also ensured correct task execution by the robot, as the diver could restart programming in case of mistakes.

V. Conclusions

This paper has presented an approach to human-robot dialog in the context of obtaining assurance prior to actions that are both risky and uncertain. Our model for risk is slightly unconventional in that it expresses the risk of a system failure and the associated recovery procedure that may be needed on the part of a human operator. Our model of dialog uncertainty is a direct product of the HMM used for recognition, and by simulating the program and likely alternatives that this observation encodes, we can obtain an estimate of the risk involved in executing the action. By seeking confirmation for particularly costly actions when they are also uncertain, we have demonstrated experimentally that this achieves an reduction in overall cost of action while requiring a relatively small number of confirmatory interactions.

In our current framework we do not combine of failure-based risk model with a cost function based on Bayes Risk. This appears to be a challenging undertaking due to the intrinsic complexity of the computation required, but it would be an appealing synthesis that would capture most of the key aspects of our problem domain. It remains an open problem for the moment. We are also interested in evaluating

the interaction mechanism across a wider user population and a larger range of dialog models, across multiple robotic platforms, including terrestrial and aerial vehicles. This study is ongoing and new results are expected on a continual basis.

VI. Acknowledgments

The authors gratefully acknowledge the contributions of Professors Joelle Pineau and Nicholas Roy, for their invaluable suggestions and ideas provided during the research. We also thank Yogesh Girdhar, Dr. Ioannis Rekleitis and all members of the McGill Mobile Robotics Lab for assisting with the validation experiments.

References

[1] G. Dudek, J. Sattar, and A. Xu, "A visual language for robot control and programming: A human-interface study," in *Proceedings of the International Conference on Robotics and Automation ICRA*, (Rome, Italy), April 2007.

[2] G. Dudek, M. Jenkin, C. Prahacs, A. Hogue, J. Sattar, P. Giguère, A. German, H. Liu, S. Saunderson, A. Ripsman, S. Simhon, L. A. Torres-Mendez, E. Milios, P. Zhang, and I. Rekleitis, "A visually guided swimming robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Edmonton, Alberta, Canada), August 2005.

[3] J. Sattar, P. Giguere, G. Dudek, and C. Prahacs, "A visual servoing system for an aquatic swimming robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Edmonton, Alberta, Canada), pp. 1483–1488, August 2005.

[4] M. Dunbabin, I. Vasilescu, P. Corke, and D. Rus, "Data muling over underwater wireless sensor networks using an autonomous underwater vehicle," in *International Conference on Robotics and Automation, ICRA 2006*, (Orlando, Florida), May 2006.

[5] S. Waldherr, S. Thrun, and R. Romero, "A gesture-based interface for human-robot interaction," *Autonomous Robots*, vol. 9, no. 2, pp. 151–173, 2000.

[6] J. K. Tsotsos, G. V. S. Dickinson, M. Jenkin, A. Jepson, E. Milios, F. Nuflo, S. Stevenson, M. B. adn D. Metaxas, S. Culhane, Y. Ye, , and R. Mannn, "PLAYBOT: A visually-guided robot for physically disabled children," *Image Vision Computing*, vol. 16, pp. 275–292, April 1998.

[7] P. E. Rybski and R. M. Voyles, "Interactive task training of a mobile robot through human gesture recognition," in *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 664–669, 1999.

[8] I. Poupyrev, H. Kato, and M. Billinghurst, *ARToolkit User Manual Version 2.33*. Human Interface Technology Lab, University of Washington, Seattle, Washington, 2000.

[9] J. Sattar, E. Bourque, P. Giguere, and G. Dudek, "Fourier tags: Smoothly degradable fiducial markers for use in human-robot interaction," *Computer and Robot Vision*, vol. 0, pp. 165–174, 2007.

[10] D. Kortenkamp, E. Huber, and P. Bonasso, "Recognizing and interpreting gestures on a mobile robot," in *13th National Conference on Artifical Intelligence*, 1996.

[11] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, and D. Brock, "Spatial language for human-robot dialogs," *IEEE Transactions on Systems, Man and Cybernetics, Part C*, vol. 34, pp. 154–167, May 2004.

[12] R. Erenshteyn and P. L. R. F. L. M. G. Stern, "Recognition approach to gesture language understanding," in *13th International Conference on Pattern Recognition*, vol. 3, pp. 431–435, August 1996.

[13] V. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[14] K. Derpanis, R. Wildes, and J. Tsotsos, "Hand gesture recognition within a linguistics-based framework," in *European Conference on Computer Vision (ECCV)*, pp. 282–296, 2004.

[15] C. Pateras, G. Dudek, and R. D. Mori, "Understanding referring expressions in a person-machine spoken dialogue," in *International Conference on Acoustics, Speech, and Signal ProcessingICASSP.*, vol. 1, 1995.

[16] M. Montemerlo, J. Pineau, N. Roy, S. Thrun, and V. Verma, "Experiences with a mobile robotic guide for the elderly," in *Proceedings of the 18th National Conference on Artificial Intelligence AAAI*, pp. 587–592, 2002.

[17] F. Doshi, J. Pineau, and N. Roy, "Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs," in *Proceedings of the 25th international conference on Machine learning*, pp. 256–263, ACM New York, NY, USA, 2008.

[18] F. Doshi and N. Roy, "Spoken language interaction with model uncertainty: an adaptive human-robot interaction system," *Connection Science*, vol. 20, no. 4, pp. 299–318, 2008.

[19] K. Krebsbach, D. Olawsky, and M. Gini, "An empirical study of sensing and defaulting in planning," in *Artificial intelligence planning systems: proceedings of the first international conference, June 15-17, 1992, College Park, Maryland*, p. 136, Morgan Kaufmann Pub, 1992.

[20] D. Kulic and E. Croft, "Safe planning for human-robot interaction," in *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04*, vol. 2, 2004.

[21] L. Rabiner *et al.*, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[22] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The Annals of Mathematical Statistics*, pp. 164–171, 1970.

[23] T. Misu and T. Kawahara, "Bayes risk-based dialogue management for document retrieval system with speech interface," *Speech Commun.*, vol. 52, no. 1, pp. 61–71, 2010.

[24] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, (Washington, DC, USA), pp. 590–596, IEEE Computer Society, 2005.