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 are 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. 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 tele–operation 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. 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. 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.

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.

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, 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

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
was purely “open–loop”, motivates this paper. Visual communica-
tion 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 envi-
ronment. 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 markers, as mentioned in the previous section,
are efficiently and robustly detectable under difficult condi-
tions. Our previous work in visual robot programming uses
the ARTag family of fiducials [8]. The ARToolkit marker
system [9] 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 [10]
fiducial system.

Gesture–based robot control has been considered exten-
sively in Human–Robot Interaction (HRI). This includes ex-
PLICIT as well as implicit communication frameworks be-

tween
human operators and robotics systems. Several authors have
considered specialized gestural behaviors [11] 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 [12].

Vision–based gesture recognition has long been consid-
ered for a variety of tasks, and has proven to be a challenging
problem examined for over 20 years with diverse well–
established applications [13][14]. The types of gestural
vocabulary range from extremely simple actions, like simple
fist versus open hand, to very complex languages, such as
the American Sign Language (ASL) [15].

While our current work looks at interaction under un-
certainty in any input modality, researchers have investi-
gated 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 [16]. Montemerlo et al.
have investigated risk functions for safer navigation and
environmental sampling for the Nursebot robotic nurse in the
care of the elderly [17]. Bayesian risk estimates and active
learning in POMDP formulations in a limited–interaction di-
alog model [18] and spoken language interaction models [19]
have also been investigated in the past. Researchers have
also applied planning cost models for efficient human–robot
interaction tasks [20] [21].

III. METHODOLOGY

In a typical human–robot interaction scenario, the human
operator instructs the robot to perform a task. Traditionally

\[ G = g_1, g_2, \cdots, g_n \]  

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 are then evaluated by a cost and risk analysis
module, comprised of a set of Assessors. This module outputs

\[ p(g_i) \]

the 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 prob-
abilistic 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
following two subsections describe in detail these two aspects
of our framework.

A. Uncertainty Modeling

To describe uncertainty modeling in our problem domain,
we introduce the following concepts. 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

\[ G = g_1, g_2, \cdots, g_n \]  

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

Fig. 1. Control flow in our risk-uncertainty model.
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 [24], 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 traveled, 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.

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

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

If we denote the uncertainty measure as $U$, the Decision Function $\rho$ can be expressed as,

$$\tau = \rho(C, U) \quad (5)$$

The function $\rho$ increases proportionally with the cost measure $C$ and is also proportional to $U$. If $\tau$ 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 $\tau$ falls below a threshold, the system will keep asking the user to provide feedback. To estimate the threshold $\tau$, 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 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 programs 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 programs are also passed through the task simulator (i.e., set of assessors) to evaluate the cost measures for all of the programs.

Once the possible programs 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 $r$. 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 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) are 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. 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. 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 $20^\circ$–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.
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.
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 was 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 programs 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 were required to
reprogram it. Thus, in all cases, users were required to
conduct the programming task until the presence or absence
of confirmation dialogs was consistent with the expected
behavior. It is worth noting, however, that this does not nec-
essarily 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.

<table>
<thead>
<tr>
<th>ID</th>
<th>Sequence</th>
<th>Confirm?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FORWARD, 3, PICTURE, LEFT, 3, PICTURE, UP, GPSFIX, GPSBEARING, EXECUTE</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>FORWARD, 9, LEFT, 6, FORWARD, 9, MOVIE, 9, RIGHT, 3, SURFACE, STOP, GPSFIX, EXECUTE</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>LEFT, 6, RIGHT, 3, MOVIE, 3, TUNETRACKER, FOLLOW, 6, UP, GPSFIX, EXECUTE</td>
<td>No</td>
</tr>
</tbody>
</table>

**TABLE I**
**PROGRAMS USED IN THE USER STUDY.**

<table>
<thead>
<tr>
<th>ID</th>
<th>Sequence</th>
<th>Confirm?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FORWARD, 9, LEFT, 5, FORWARD, 9, LEFT 5, STOP, MOVIE, 9, EXECUTE</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>FORWARD, 5, LEFT, 3, FORWARD, 5, LEFT 3, FORWARD, 5, LEFT 3, STOP, EXECUTE</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>SWIMCIRCLE, 3, STOP, EXECUTE</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>SWIMCIRCLE, 3, FORWARD, 5, PICTURE, LEFT, 2, PICTURE, FORWARD, 3, PICTURE, RIGHT, 2, PICTURE, SURFACE, STOP, EXECUTE</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>TUNETRACKER, FOLLOW, 9, SURFACE, STOP, EXECUTE</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**TABLE II**
**PROGRAMS USED IN THE FIELD TRIALS.**

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 program once.
Unlike in the user study, where there was 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 limited feedback in the
form of symbol acknowledgement using an Organic–LED (Light Emitting Diode) display at the back of the robot. Also
unlike the user studies, the field trial participants 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 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 not in the robot’s field of view, the
tracking algorithm attempts to reacquire the target, and this can be seen in the video clip.

**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

![Fig. 2](image)

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

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, with ARTag and ARToolkitPlus 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

![a](image)

(a) Divers programming Aqua2 during pool trials.

![b](image)

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

Fig. 2. Field trials of the proposed algorithm on board the Aqua2 robot.
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 3(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 use of confirmations increases total programming time by approximately 50%. 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 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 $6 \times 3 = 18$ seconds to $9 \times 3 = 27$ 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 3(b).

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 a reduction in overall cost of action while requiring a relatively small number of confirmatory interactions.

In our current framework we do not combine a 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.

REFERENCES


