A Representation for Interaction

Alan R. Wagner, *Member, IEEE*

Abstract—**This paper explores the use of an outcome matrix as a computational representation of social interaction suitable for implementation on a robot or software agent. An outcome matrix expresses the reward afforded to each interacting individual with respect to pairs of potential behaviors. We detail the use of the outcome matrix as a representation of interaction in social psychology and game theory, present a formal notation based on these fields for describing interaction, and contribute a novel algorithm for creating outcome matrices from perceptual information and predefined knowledge. We also explore the representation's sensitivity to different types of error and present results showing that, in many cases, outcome matrices are not affected by error and uncertainty. Experiments are conducted in a novel simulation environment with the potential to aid the repeatability of human-robot interaction experiments.**

I. Introduction

Many scientists have recently come to recognize the social aspects of intelligence [1]. In contrast to purely cognitive intelligence, which is most often described by problem solving ability and/or declarative knowledge acquisition and usage, social intellect revolves around an individual's ability to effectively understand and respond in social situations [2]. Compelling neuroscientific and anthropological evidence is beginning to emerge supporting theories of social intelligence [3, 4]. From a roboticist's perspective, it then becomes natural to ask how this form of intelligence could play a role in the development of an artificially intelligent robot. As an initial step, one must first consider which concepts are most important to social intelligence.

Social interaction is one fundamental concept [5]. Social psychologists define *social interaction* as influence—verbal, physical, or emotional—by one individual on another [6]. If a goal of artificial intelligence is to understand, imitate, and interact with humans then researchers must develop computational representations for interaction that will allow an artificial system to: (1) use perceptual information to generate its representation for interaction; (2) represent its interactions with a variety of human partners in numerous different social environments; and (3) afford the robot guidance in selecting interactive actions.

This paper presents a representation that allows a robot to manage these challenges. A general, established, computational representation for social interaction that is not tied to specific social environments or paradigms is presented [7, 8]. Moreover, we contribute an algorithm that allows a robot to create representations of its social interactions. High fidelity simulation results demonstrate our algorithm in several different domains and with numerous different types of partners. Moreover, we investigate the robustness of this representation when faced with several types of errors. Overall, the purpose of this paper is to introduce the outcome matrix as an important potential representation of social interaction in artificial systems.

The remainder of this paper begins by first summarizing relevant research. Next, we present a representation for social interaction and argue why this representation is suitable for implementation on a robot. We then present our algorithm for populating the representation with information. This article concludes with experiments demonstrating the resiliency of the representation to different types of error and a discussion of these results including directions for future research.

Manuscript received January 8, 2008.

A. R. Wagner is a doctoral candidate at Georgia Institute of Technology 85 5th Street, Room S27, NW, Atlanta, GA. 30308, USA (phone 1-404-894-9311; email: alan.wagner@cc.gatech.edu)

II. Related Work

Representations for interaction have a long history in social psychology and game theory [7, 8]. Interdependence theory, a type of social exchange theory, is a psychological theory developed as a means for understanding and analyzing interpersonal situations and interaction [8]. The term interdependence specifies the extent to which one individual of a dyad influences the other. Interdependence theory is based on the claim that people adjust their interactive behavior in response to their perception of a social situation's pattern of rewards and costs. Thus, each choice of interactive behavior by an individual offers the possibility of specific rewards and costs—also known as outcomes—after the interaction. Interdependence theory represents interaction and social situations computationally as an outcome matrix (figure 1). An outcome matrix represents an interaction by expressing the outcomes afforded to each interacting individual with respect each pair of potential behaviors chosen by the individuals.

Game theory also explores interaction. Moreover, game theory has been described as "a bag of analytical tools" to aid one's understanding of strategic interaction [7]. As a branch of applied mathematics, game theory thus focuses on the formal considerations of strategic interactions, such as the existence of equilibriums and economic applications [9]. Game theory uses the normal form game as its representation of interaction. This normal form game is equivalent to social psychology's outcome matrix. Numerous researchers have used game theory to control the behavior of artificial agents in multi-agent environments [10]. We, however, know of no direct exploration of the outcome matrix as a means of representing human-robot interaction.

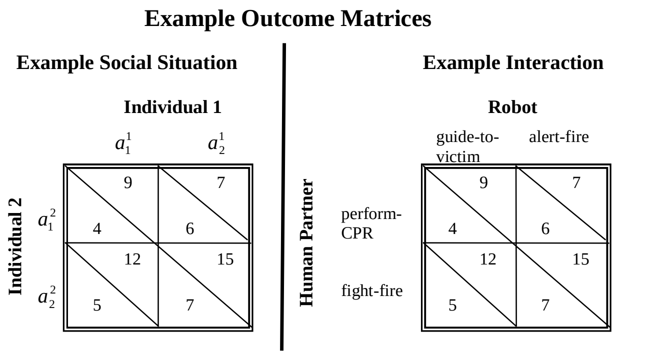


Figure 1. Example outcome matrices are depicted above. The right hand side depicts an outcome matrix representing an actual interaction encountered by the robot in the experiments. The left hand side depicts a social situation. Social situations abstractly represent all interactions with the same outcome values.

This work differs from much of current human-robot interaction research in that our work investigates theoretical aspects of human-robot interaction. Typically, HRI research explores the mechanisms for interaction, such as gaze following, smooth pursuit, face detection, and affect characterization [11].

III. Representing Social Interaction

As mentioned in the preceding section, the outcome matrix (see figure 1 for an example) is a standard computational representation for interaction [7, 8]. It is composed of information about the individuals interacting, including their identity, the interactive actions they are deliberating over, and scalar outcome values representing the reward minus the cost, or the outcomes, for each individual. Thus, an outcome matrix explicitly represents information that is critical to interaction. Typically, the identity of the interacting individuals is listed along the dimensions of the matrix. Figure 1 depicts an interaction involving two individuals. For this paper the term individual is used to indicate either a human or a social robot or agent. We will focus on interaction involving two individuals—dyadic interaction. An outcome matrix can, however, represent interaction involving more than two individuals. The rows and columns of the matrix consist of a list of actions available to each individual during the interaction. Finally, a scalar outcome is associated with each action pair for each individual. Outcomes represent unitless changes in the robot, agent, or human's utility. Thus, for example, an outcome of zero reflects the fact that no change in the individual's utility will result from the mutual selection of that action pair.

The outcome matrix also contains information relating to Theory of Mind [12]. Theory of mind describes that ability of an individual to attribute particular mental states to other individuals. Accurate population of an outcome matrix requires the ability to calculate the outcome values for another individual. Moreover, creation of an outcome matrix assumes the ability to determine which actions are possible or even probable a partner. Thus, as a representation of interaction, the outcome matrix highlights the role of Theory of Mind that is necessary for proper interaction.

A. Outcome Matrix Notation

Because outcome matrices are computational representations, it is possible to describe them formally. Doing so allows for powerful and general descriptions of interaction. In this section, we present a formal notation for interaction drawing heavily from game theory [7, 9]. A representation of interaction consists of 1) a finite set *N* of interacting individuals; 2) for each individual $i \in N$ a nonempty set A^i of actions; 3) the utility obtained by each individual for each combination of actions that could have been selected [9]. Let $a_j^i \in A^i$ be an arbitrary action *j* from individual *i*'s set of actions. Let $\left(a_j^1, \ldots, a_k^N\right)$ denote a combination of actions, one for each individual, and let u^i denote individual *i*'s utility function: $u^i\left(a_j^1, \ldots, a_k^N\right) \rightarrow \Re$ is the utility received by individual *i* if the individuals choose the actions $\left(a_j^1, \ldots, a_k^N\right)$.

The term O is used to denote an outcome matrix. The superscript -*i* is used to express individual *i*'s partner. Thus, for example, A^i denotes the action set of individual *i* and A^{-i} denotes the action set of individual *i*'s interactive partner. A particular outcome can also be expressed as a function of an outcome matrix and an action pair, thus for two interacting individuals $O^1\left(a_1^1, a_1^2\right) = o_{11}^1$ and $O^2\left(a_1^1, a_1^2\right) = o_{11}^2$. In words, the selection of action a_1^1 by individual 1 and action a_1^2 by individual 2 results in outcome o_{11}^1 for individual 1 and o_{11}^2 for 2. Applied to figure 1 $O^1\left(a_1^1, a_1^2\right) = 9$ and $O^2\left(a_1^1, a_1^2\right) = 4$.

B. Representing social situations

The term interaction describes a discrete event in which two or more individuals select interactive behaviors as part of a social situation or social environment. Interaction has been defined as influence—verbal, physical, or emotional—by one individual on another [6]. The term situation has several definitions. The most apropos for this work is "a particular set of circumstances existing in a particular place or at a particular time [13]." A social situation, then, characterizes the environmental factors, outside of the individuals themselves, which influence interactive behavior. A social situation is abstract, describing the general pattern of outcome values in an interaction. An interaction, on the other hand, is concrete with respect to the two or more individuals and the social actions available to each individual. For example, the prisoner's dilemma describes a particular type of social situation. As such, it can, and has been, instantiated in numerous different particular social environments ranging from bank robberies to the trenches of World War I [14]. Interdependence theorists state that interaction is a function of the individuals interacting and of the social situation [15]. Although a social situation may not afford interaction, all interactions occur within some social situation. Interdependence theory represents social situations involving interpersonal interaction as outcome matrices (see figure 1 for a graphical depiction of the difference).

In previous work, we presented a situation analysis algorithm that calculated characteristics of the social situation or interaction (such as interdependence) when presented with an outcome matrix [16]. These characteristics were then used to influence the robot's action selection. Our results showed that by analyzing the situation, the robot could better select interactive actions. Thus, using an outcome matrix as a representation of interaction can benefit the robot in terms of selecting the best action.

C. *Action selection strategies*

As mentioned in section I, a computational representation for interaction should afford the robot guidance in selecting interactive actions. Outcome matrices afford several simple action selection strategies. The most obvious method for selecting an action from an outcome matrix is to choose the action that maximizes the robot's outcome. This strategy is termed *max_own*. An individual's use of the *max_own* strategy results in egoistic interactive behavior. Alternatively, the robot may select the action that maximizes its partner's outcome, a strategy termed *max_other*. An individual's use of the *max_other* strategy results in altruistic behavior. Yet another action selection strategy is for the robot to select the action that maximizes the sum of its and its partner's outcome. The use of this strategy results in a cooperative style of behavior. Outcome matrices afford many other simple action selection strategies (see [16] for other examples).

If we are to use the outcome matrix as a representation of interaction for a robot, it becomes critical to develop algorithms for creating outcome matrices. In the next section, we present a preliminary algorithm capable of generating outcome matrices.

IV. From Interaction To Outcome Matrix

Figure 2 depicts our algorithm for generating outcome matrices from social interaction. The algorithm takes as input a partner type, robot type, and environment type. The algorithm returns an outcome matrix representing the social interaction faced by the robot. Overall, the algorithm acts as a stepwise method for filling in the information contained within an outcome matrix. The first line creates an empty matrix—a matrix devoid of information pertaining to the interactive partner, any actions, or outcome values.

Outcome Matrix Creation

Input: Environment type $e \in E$, partner type $t^{-i} \in T^{-i}$, robot type $t^i \in T^i$.
Output: Outcome matrix O representing an interaction.

```
1.   Create empty matrix $O$.
2.   Set $O$.partner = $t^{-i}$
3.   Set $A^{i*} = f(t^i); A^{-i*} = g(t^{-i}); A^e = h(e)$
4.   $A^i = \{a^i_j \mid a^i_j \in A^{i*} \wedge a^i_j \in A^e\}$
5.   $A^{-i} = \{a^{-i}_j \mid a^{-i}_j \in A^{-i*} \wedge a^{-i}_j \in A^e\}$
6.   Set $O$.columns = $A^i$, $O$.rows = $A^{-i}$
7.   For each action pair $a^i_j, a^{-i}_k$ in $A^i, A^{-i}$
8.       $O^i(a^i_j, a^{-i}_k) \leftarrow u^i(a^i_j, a^{-i}_k)$.
9.       $O^{-i}(a^i_j, a^{-i}_k) \leftarrow u^{-i}(a^{-i}_k, a^i_j)$.
     End
10.  Return $O$.
```

Figure 2. An algorithm for outcome matrix creation is presented above. The algorithm takes the environment, robot, and partner type as input. The algorithm acts as a stepwise method for adding the information required by the outcome matrix.

The second line sets the identification of the partner in the outcome matrix to their type. This line simplifies a process that we expect will become more complex in future refinements of this algorithm. In future work, perceptual characteristics of the partner will be used to construct the partner's identification. For example, perceptual features such as male or female, hair color and body type could all be used to construct the identification of a new partner. Other perceptual features will relate to the partner's type. For example, a badge could be used to distinguish a police officer from a firefighter. Perceptual information will also be used to determine the type of environment. For example, smoke could be used to indicate a search and rescue environment.

The third line sets the action set for the environment type, partner type, and robot type (table 1 lists the different partner and environment types). Associative memory was used to assign the actions sets from the different types.

The fourth line constructs the robot's action set for the interaction. This step uses knowledge of what actions the robot can perform (A^{i*}) and what actions can be performed in a given environment (A^e) to construct a set of actions that the robot can perform in the environment. The fifth step constructs the partner's action set in the same manner and assumes that the robot knows what actions the partner can perform (A^{-i*}). We are currently developing algorithms that will allow the robot to learn this information. In the sixth step, the rows and columns of the outcome matrix are set to the robot and partner's action sets.

Table 1. The different environment, partner, and robot types.

Environment Type	Partner Type	Robot Type
assistive	police officer	police officer aid
household	firefighter	firefighter aid
museum	accident victim	medical aid
prison	hospital patient	
search and rescue	citizen	
	medical staff	

Lines seven through nine populate the empty outcome matrix with outcome values. This is accomplished by iterating through all pairs of actions $\left(a_j^i, a_j^{-i}\right)$ and for each pair using the individual's utility function to produce the outcome value. These steps assume that robot has a utility function both for itself and for its partner.

Finally, line ten returns the matrix.

Clearly, the use of this algorithm requires a great deal of knowledge on the part of the robot. The robot must have information not only about its partner, but also about the environment and itself. This begs two important questions: 1) where does this information come from? and 2) how accurate must this information be? We are currently working to address the first question by developing algorithms that will allow the robot to learn much of this information. We address the second question in the experiments presented below.

V. Simulation Environment

We conducted simulation experiments to test the proposed algorithm. Our experimental environment was built on USARsim, a collection of robot models, tools, and environments for developing and testing search and rescue algorithms in high-fidelity simulations [17]. USARsim's robot models have been shown to realistically simulate actual robots in the same environment [18]. Moreover, USARsim provides support for sensor and camera models that allow a user to simulate perceptual information in a realistic manner. USARsim is freely available online.

USARsim is built on Epic's Unreal Tournament (UT) game engine. Unreal Tournament is a popular 3D first person shooter game. Unreal Tournament's game engine produces a high-quality graphical simulation environment that includes the kinematics and dynamics of the environment. Numerous tools for the creation of new environments, objects, and characters are included with the game. These tools can be used to rapidly prototype novel environments at minimal cost. Moreover, numerous complete environments and decorative objects are freely available online. Figure 3 depicts several examples of environments we created for this work using Unreal Tournament tools.

Collectively, USARsim and Unreal Tournament offer the exciting possibility of creating standard testbeds for HRI. For example, the environments created as part of this work were loosely designed from the setup of Georgia Tech's Mobile Robot Laboratory. Carpin et al. describe a process for creating high fidelity environments from CAD models of actual search and rescue arenas [18]. Using this method, precise simulation environments of HRI laboratories could also be created. In this manner, standard environments for a household robot, search and rescue robot, and assistive robot could potentially be created from actual homes, disaster sites, and hospitals. Once posted on the internet, these simulation environments could then be used by others to confirm or test HRI algorithms and architectures.

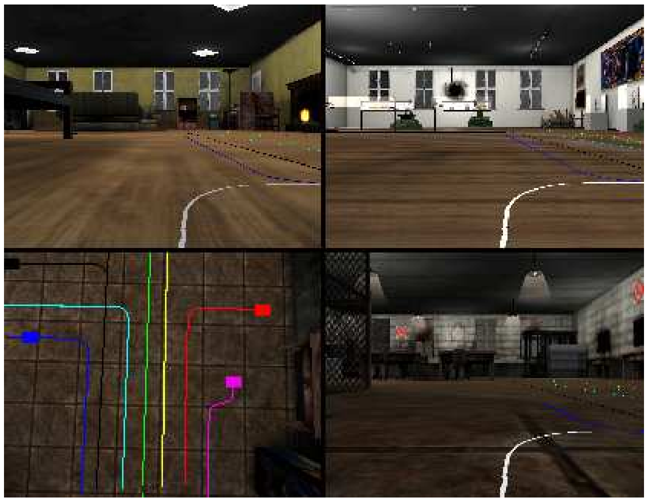
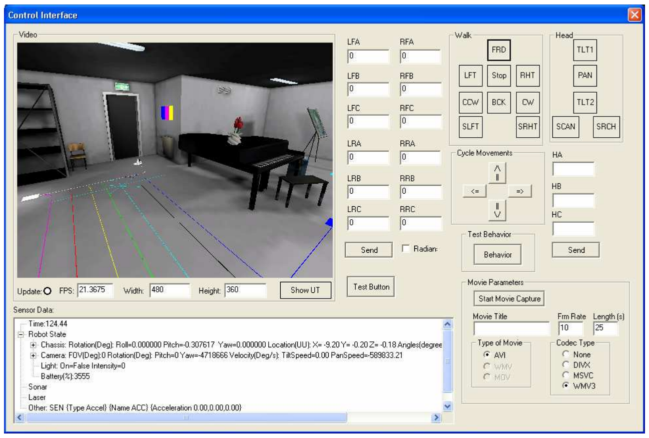


Figure 3. Screenshots from the simulation environment are depicted above. The top left shows a household environment. The top right depicts a museum environment. The bottom right shows the prison environment. These three screenshots are from the robot's perspective. The bottom left illustrates the system of colored lines used to aid the robot's navigation to different parts of each environment.

We created five different environments to test the generality of our algorithm. The household environment modeled a small studio apartment and contained couches, a bed, a television, etc. (figure 3 top left). The museum environment modeled a small art and sculpture gallery and contained paintings, statues, and exhibits (figure 3 top right). The prison environment modeled a small prison and contained weapons, visiting areas, and a guard station (figure 3 bottom right). The search and rescue environment modeled a disaster area and contained debris fields, small fires, victims, and a triage area. Finally, the assistive environment modeled a small hospital or physical therapy area and contained equipment for physical, art, music and occupational therapy. Each of the environments contained colored lines on the floor that helped the robot navigate to different locations in the environment (figure 3 bottom left for example). Line following code, created using OpenCV, allowed the simulated robot to follow the lines to different parts of the environment. These color lines simply aided navigation and had no impact on the algorithm itself.

The USARsim model of the Sony AIBO was used in all experiments. The robot used speech synthesis to communicate questions and information to the human partner. Speech recognition translated the spoken information provided by the human. Microsoft's Speech SDK provided the speech synthesis and recognition capabilities.



```
Update: FPS: 21.3675  Width: 480  Height: 360
LFA 0  RFA 0  LFB 0  RFB 0  LFC 0  RFC 0
LRA 0  RRA 0  LRB 0  RRB 0  LRC 0  RRC 0
Frm Rate 10  Length (s) 25
Type of Movie: (•) AVI ( ) WMV ( ) MOV
Codec Type: ( ) None ( ) DIVX ( ) MSVC (•) WMV3
Sensor Data:
  Time:124.44
  Robot State
    Chassis: Rotation(Deg): Roll=0.000000 Pitch=-0.307617  Yaw=0.000000 Location(UU): X= -9.20 Y= -0.20 Z= -0.18 Angles(degree
    Camera: FOV(Deg):0 Rotation(Deg): Pitch=0 Yaw=-4718666 Velocity(Deg/s): TiltSpeed=0.00 PanSpeed=-589833.21
    Light: On=False Intensity=0
    Battery(%):3555
  Sonar
  Laser
  Other: SEN {Type Accel} {Name ACC} {Acceleration 0.00,0.00,0.00}
```

Figure 4. The interface used by the robot's human partner is illustrated above. The software allows the human partner to view and move through the environment. Speech is used to communicate with the robot. This software was developed from tools provided in the USARSim package [19].

For our experiments, the robot's human partner used the interface depicted in figure 4 to interact with the robot. This interface was developed from an existing USARsim tool [19]. The interface allows the human to move around and view the environment. The human interacted with the robot by speaking a predefined list of commands.

VI. Experiments

Several experiments were performed using the simulation environment. The first experiment demonstrates the use of the algorithm in several different environments. Additional experiments examine the sensitivity of the outcome matrix to different types of error.

A. Experimental setup

In order to conduct the experiments, actions sets and utility functions needed to be developed for the robot and its partner. The action set for the robot was created by reasoning about the types of actions a Sony AIBO robot might be capable of in a particular environment. For example, the robot can use its lights and speakers to alert nearby people, use its camera to observe scenes, use its wireless card to send this information to an observation station, and guide a human to a location in the environment. The action set for the human was derived by reasoning about the types of actions available to a police officer, firefighter, victim, citizen, medical staff, and hospital patient in each of the environments (see table 2 for a list of example actions).
Utility functions were created by reasoning about the ordering of each individual's actions in an environment (creating a preference relation over the actions). Once the individual's actions were ordered, an action in the list was selected to be the zero point. The distance from the zero point determined the action's outcome value. The value for action pairs was determined by summing the value for each individual action. For example, if the robot's ordered action list is `alert-guards` and `observe-exhibit` and the partner's ordered action list of actions is `perform-CPR` and `rescue-person` and the utility of `observe-exhibit` and `perform-CPR` are set to zero then utility of each action is `alert-guards`=1, `observe-exhibit`=0, `perform-CPR`=0, and `rescue-person`=-1. The utility of the action pairs would then be (`alert-guard`, `perform-CPR`) =1 (`alert-guard`, `rescue-person`) =0, (`observe-exhibit`, `perform-CPR`) =0, and (`observe-exhibit`, `rescue-person`) =-1.

Table 2. Example actions for different types of individuals.

Partner Type	Example actions	Robot type	Example actions
police officer	perform-CPR, arrest-person, search-home	police aid	alert-guards, alert-security, observe-exhibit
fire fighter	perform-CPR, fight-fire, rescue-person	fire fighter aid	alert-victim, alert-fire, guide-to-victim
accident victim	crawl, limp, moan	medical aid	guide-to-victim, guide-to-triage, alert-medical-staff
hospital patient	get-food, do-art-therapy, watch-TV		
citizen	watch-scene, talk, run-away		
medical staff	stabilize-person, treat-illness, assess-person		

All experiments consisted of a Sony AIBO ERS robot and a human partner (the experimenter) interacting in one of the five different environments (see table 1). Before interacting, the robot and the partner were assigned types from the list in Table 1.

Each interaction began with the assignment of a type for the robot, the partner, and the environment. These types served as input to the algorithm in figure 2. As discussed in section 4, the algorithm then produces an outcome matrix representing the interaction. In all of the experiments, the robot used a *max_own* strategy to select its interactive action. Using the lines depicted in figure 3 to navigate to a location, the robot would then perform the action. The interaction ended once the robot performed the action.

B. Matrix generation

We tested the algorithm in simulation in each of the five different environments (from table 1) and with the robot acting as either a police aid, firefighter aid, or medical aid.
Table 3 presents the actions selected by the robot for each environment and robot type. The actions are generally type and environment specific. For example, the robot performs actions related to its role as a medical aid when it is assigned the role of medical aid. In the case when the robot is

assigned the role of firefighter aid, it selects the same action, `alert-victim`, regardless of the type of environment. This is because the robot's action set for the firefighter aid type remained relatively constant for each environment when acting as this type.

C. *Errors in outcome value*

A common concern about the outcome matrix as a representation for interaction is that it consists of a daunting amount of information about both individuals and the social situation. These experiments address this concern by showing that the outcome matrix often selects the correct action in spite of significant errors in outcome value. Moreover, we show that the performance of the outcome matrix degrades gracefully as the number of errors in the matrix increases.

Table 3. Actions selected in different environments and different robot types. The partner was of type police officer.

	Environment Type				
Robot Type	**Assistive**	**Househd**	**Mus.**	**Prison**	**Search and Rescue**
Police Aid	alert-security	oblert-security	oblert-security	guide-to-alarm	alert-victim
Fire Aid	alert-victim	alert-victim	alert-victim	alert-victim	alert-victim
Med. Aid	guide-to-movemt-therapy	assist-movemt-therapy	oblert-security	alert-guards	guide-to-victim

The data presented in the following subsections was collected from every combination of environment (5), robot type (3) and human type (6). Thus, 90 simulations were run to study each type of outcome matrix error.

One potential type of outcome matrix inaccuracy is an error in the value of an outcome. As mentioned in section 3, outcome values are the scalar numbers produced by the robot's utility function to populate the outcome matrix. Further, these values constitute the basis on which the robot will select an interactive action. A robot using a *max_own* action selection strategy would, for example, select the action resulting in the largest outcome value for itself. It would therefore seem that accurate outcome values are critical for this representation. As we show below, this is not the case.

1) *Errors in magnitude*

An error of magnitude is an error in the utility function in which all outcome values are greater or less than their true value. Consider the example matrix from figure 1. Using a *max_own* action selection strategy the robot would select the `alert-fire` action because the action pair (`alert-fire, fight-fire`) results in the largest outcome for the robot. Even if a utility function error results in an increase of all outcomes by 10, the robot will still select the same action. Moreover, the same is true of any type of systematic error (dividing by a positive value, multiplying by a positive number, etc.) that alters the magnitude of all values but does not alter their overall rank order in terms of outcome. These types of errors will not affect the ability of the robot to select the correct action.

2) *Errors of single outcome values*

Errors in magnitude affect all outcome values. What about errors that do not affect all outcome values? In contrast to an error in magnitude, errors with respect to single outcome values may result in a new ordering of actions. These errors occur when the robot's utility function generates inaccurate values with respect to particular action pairs. To reproduce this type of error, first the utility function produced outcome values for a matrix (for each combination of environment, partner type, and robot type). Next, random outcome values were selected and these values were randomly changed to produce outcome matrix errors. This type of error results in incorrect action if one of two conditions are met: 1) the current maximum outcome value is selected for change, the new value is less then the second largest value, and the second largest value does not occur in the same column as the old maximum value; 2) the selected outcome value does not occur in the same column as the current maximum value and the new value is greater than the current maximum value. Consider again the example from figure 1. The first condition is met when the outcome value for the action pair (`alert-fire, fight-fire`) is randomly selected and changed to a value less than 12 resulting in the robot choosing action `guide-to-victim`. The second condition is met if an outcome value in the column for `guide-to-victim` is selected and changed to a value greater than 15. We examined the impact of these types of errors experimentally. Figure 5 shows our results.

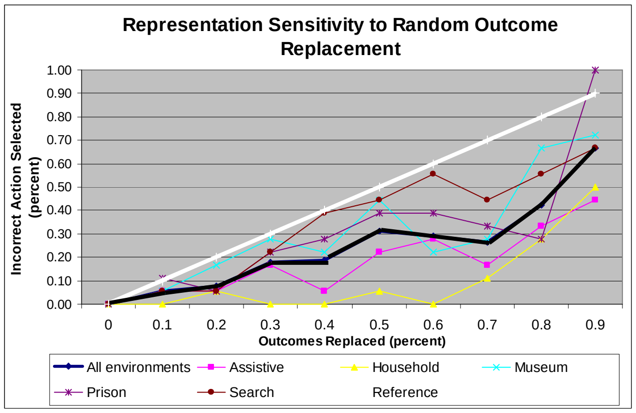


Figure 5. The graph depicts the percent of incorrect actions selected as a function of increasing random outcome replacement. A y-axis value of 1.00 represents total selection of incorrect actions. The bold black line depicts the average incorrect actions selected for all environments. The individual colored lines represent changes in accuracy for each different environment. The bold white line is a baseline for comparison, depicting a linear decrease in accuracy.

The dependent variable for all experiments is the percentage of incorrect actions selected by the robot. The percent of incorrect actions selected was calculated by comparing an outcome matrix with random errors to a matrix with no random errors added. The independent variable for this experiment was the percent of outcome values replaced with error values within the matrix. The bold black line in figure 5 depicts the average result over all five environments. Thinner lines depict the results for individual environments. The bold white line provides a baseline by depicting the expected result if the percent of incorrect actions selected increased at a rate equal to the number of errors added to the matrix. Thus, if the bold black line is below the bold white line, then action selection inaccuracy increases more slowly then the rate of errors introduced. The reason for this is that many random errors do not meet the conditions listed above, and thus do not affect the ability of the matrix to select the correct action. In other words, the outcome matrix representation degrades gracefully with respect to increasing outcome value inaccuracy.

D. Action set errors

In addition to errors involving outcome values, the action set from which the outcome matrix is constructed can be flawed. In this case, valid actions may have been left out of the matrix or actions unsuitable for the environment may have been inserted. The subsections below investigate both of these types of errors.

1) Action deletion errors

An action deletion error occurs when an action, suitable for the robot's environment, has been left out of the matrix. This type of error can occur whenever the robot lacks a good model of its own actions. Even more likely, the matrix may contain omissions with respect to the actions of the robot's partner. The effect of action deletion errors with respect to the partner depends on the action selection strategy. The deletion of any one action only affects the matrix's accuracy when the action that would have otherwise been selected is deleted. The probability that this is not the case is $\frac{m-1}{m}$, where m is the number of columns in the matrix. Hence, action deletion is less likely to affect the matrix when actions are deleted from larger matrices and impact of these deletions will increase approximately linearly as the size of the matrix decreases. We see again, that, on average, the accuracy of the outcome matrix representation degrades gracefully with respect to increasing error.

The dependent variable for this experiment was the percentage of incorrect actions selected by the robot. The independent variable was the percentage of actions deleted from the matrix. Again, the bold black line in figure 6 depicts the average over all environments and the bold white line provides a baseline for comparison. The graph shows that the number of incorrect actions selected increases at approximately the same rate as the number of actions deleted from the matrix. In other words, the accuracy of an outcome matrix is approximately equal to the number of actions deleted (assuming no other types of errors).

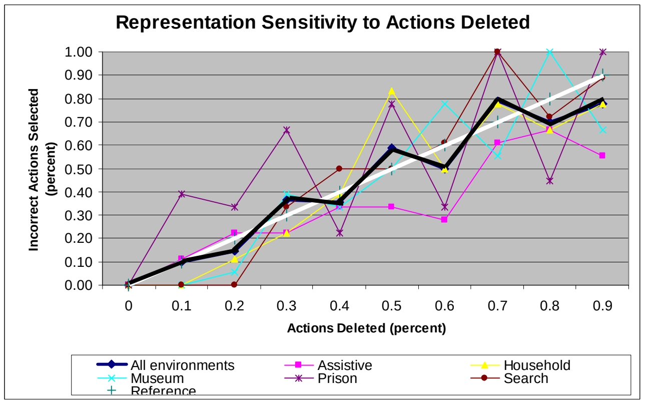


Figure 6. The graph depicts the percent of incorrect actions select as a function of increasing random action deletion. The bold black line depicts the average incorrect actions selected for all environments. The individual colored lines represent changes in accuracy for each different environment. The bold white line is a baseline for comparison, depicting a linear decrease in accuracy.

2) Action insertion errors

An outcome matrix can also contain actions that are not possible given the type of environment. Moreover, because each invalid action results in several invalid outcome values, these types of errors have the potential to flood the outcome matrix with improper outcome values. An action insertion error results in the incorrect selection of an action only if one of the outcome values for the new action is greater then the outcome values for all other actions. In other words, an error only occurs if the incorrectly inserted action adds a new maximum value to the matrix. Because an incorrectly added action may not result in a new maximum value for the matrix, the percentage of incorrect actions selected by the robot is expected to increase less than the rate of the number of insertion errors added.

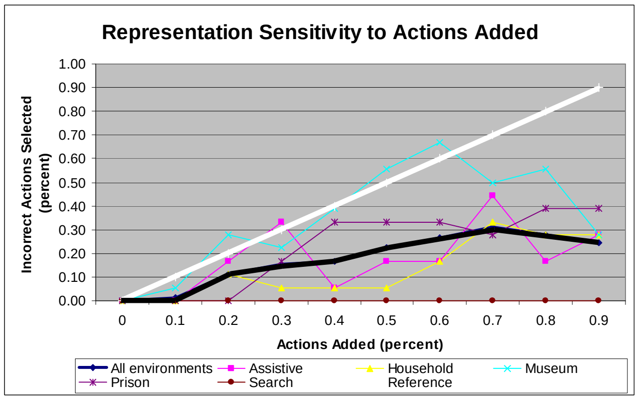


Figure 7. The graph depicts the percent of incorrect actions select as a function of increasing random action insertion. As in the other graphs the bold black line depicts the average incorrect actions selected for all environments, the colored lines represent the changes in accuracy for each different environment, and the bold white line is a baseline for comparison, depicting a linear decrease in accuracy.

Figure 7 confirms this expectation. Here again we see that the bold black line is below the bold white line indicating that action selection accuracy decreases more slowly then the rate that incorrect actions are inserted.

Many of our results assumed the use of the *max_own* action selection strategy. These results also apply if other action selection strategies, such as the *max_other* strategy, are used instead.

VII. SUMMARY AND CONCLUSIONS

This paper has introduced a computational representation for interactions and social situations suitable for implementation on a robot or software agent. We have discussed the composition of this representation, its ability to represent both interactions and social situations, and formal operations related to the representation. Moreover, we have presented a preliminary algorithm for the creation of the representation. This paper has also described the use of an exciting and largely unexplored simulation environment that could improve the repeatability of HRI experiments. Finally, we have experimentally examined the effect of different types of errors on the accuracy of the representation.

In this paper, we have shown that outcome matrices are capable of representing interactions with a variety of different human partners and afford the robot guidance in selecting interactive actions. We demonstrated the use of an algorithm for creating outcome matrices in several different environments and with different robot types. Additionally our results show that, on average, outcome matrices degrade gradually to errors in outcome value, action insertion errors, and action deletion errors. Moreover, we have shown that outcome matrices are unaffected by errors that do not alter the action preference relation. These results are important because they affect the development of algorithms that will produce these outcome matrices from interactions. Further, because interaction often entails a large amount of uncertainty, the results are critical as they show that the information that composes an outcome matrix does **not** need to be perfectly accurate in order for the matrix to be of use.

Nevertheless, our work leaves many questions open. Although we present a general algorithm for creating outcome matrices, our algorithm assumes a great deal of knowledge on the part of the robot. To be precise, we assume the robot maintains a utility function for itself and its partner as well as knowledge relating to which actions are possible in a given environment. We are currently developing algorithms that will allow the robot to learn the information assumed in this work. Future work will also allow the robot to use and reason about its and its partner's type. Together, we expect these advances to allow the robot to learn from and reason about its human partners—one small step towards artificial social intelligence.

ACKNOWLEDGMENT

The author would like to thanks Zsolt Kira and Ron Arkin for their helpful comments.

REFERENCES

[1] R. W. Byrne and A. Whiten, "Machiavellian intelligence," in *Machiavellian Intelligence II: Extensions and Evaluations*, A. Whiten and R. W. Byrne, eds., Cambridge University Press, Cambridge, 1997, pp. 1-23.

[2] N. K. Humphrey, "The social function of intellect," in *Growing Points in Ethology*, P. P. G. Bateson and R. A. Hinde, eds., 1976, pp. 303-317.

[3] R. Bar-On, D. Tranel, N. L. Denburg, and A. Bechara, "Exploring the neurlogical substrate of emotional and social intelligence," *Brain*, vol. 126, 2003, pp. 1790-1800.

[4] T. J. Bergman, J. C. Beehner, D. L. Cheney, and R. M. Seyfarth, "Hierarchical Classification by Rank and Kinship in Baboons," *Science*, vol. 302, 2003, pp. 1234-1236.

[5] R. A. Hinde, "Can Nonhuman Primates Help Us Understand Human Behavior?" in *Primate Societies*, B. B. Smuts, D. L. Cheney, R. M. Seyfarth, R. W. Wrangham, and T. T. Struhsaker, eds., University of Chicago Press, Chicago, 1987, pp. 413-420.

[6] D. O. Sears, L. A. Peplau, and S. E. Taylor, *Social Psychology*. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

[7] Osborne, M. J., & Rubinstein, A. *A course in game theory*. Cambridge, MA: MIT Press, 1994

[8] H. H. Kelley and J. W. Thibaut, *Interpersonal Relations: A Theory of Interdependence*. John Wiley & Sons, New York, NY, 1978.

[9] Gibbons, R.. *Game theory for applied economists*.Princeton, NJ: Princeton University Press, 1992

[10] Crandall, J. W., & Goodrich, M. A. Multiagent learning during on-going human-machine interactions: The role of reputation, *In AAAI Spring Symposium: Interaction between Humans and Autonomous Systems over Extended Operation*. Stanford, CA., 2004.

[11] T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, 2003, pp. 143-166.

[12] Premack, D. G. and Woodruff, G. "Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences, 1,* 515-526, 1978.

[13] Situation. In *Encarta world English dictionary, north American edition*, 2007.

[14] Axelrod, R. *The evolution of cooperation*. New York: Basic Books, 1984.

[15] C. E. Rusbult and P. A. M. Van Lange, "Interdependence, Interaction and Relationship," *Annual Review of Psychology,* vol. 54, 2003, pp. 351-375.

[16] A. R. Wagner and R. C. Arkin, A Framework for Situation-based Social Interaction, in *Proceedings of the 15th International Symposium on Robot and Human Interactive Communication (Ro-Man 2006)* Hatfield, United Kingdom, 2006, pp. 351-375.

[17] Stefano Carpin, Jijun Wang, Michael Lewis, Andreas Birk, Adam Jacoff: High Fidelity Tools for Rescue Robotics: Results and Perspectives. RoboCup 2005: 301-311

[18] J. Wang, M. Lewis, S. Hughes, M. Koes, and S. Carpin, "Validating usarsim for use in hri research," in *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting (HFES'05)*, 2005, pp. 457–461.

[19] Zaratti, M., M. Fratarcangeli, and L. Iocchi, *A 3D Simulator of Multiple Legged Robots based on USARSim*, in *Proc. of RoboCup Symposium 2006*.