Proceedings of the 2002 IEEE
Int. Workshop on Robot and Human Interactive Communication
Berlin, Germany, Sept. 25-27, 2002

Mutual Adaptation to Mind Mapping in Human-Agent Interaction

Seiji YAMADA
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda, Tokyo
101-8430, Japan
seiji@nii.ac.jp

Tomohiro YAMAGUCHI
Nara National College of Technology
22 Yata-cho, Yamato-Koriyama, Nara
639-1080, Japan
yamaguch@info.nara-k.ac.jp

Keywords

Mutual adaptation between human and robot, mind-expression mapping, instance-based learning

Abstract

This paper describes a human-agent interaction framework in which a user and a life-like agent mutually acquire the other's mind mapping through a mutual mind reading game. A lot of studies have been done on a life-like agent including humanoid robots. Through development of various life-like agents, an internal state (we call mind) of an agent like emotion, processing load has been recognized to play an important role in making them believable to a user. For establishing effective and natural communication between an agent and a user, they need to read the other's mind from expressions and we call the mapping from expressions to mind states mind mapping. If an agent and a user don't obtain these mind mappings, they can not utilize behaviors which significantly depend on the other's mind. We formalize such mutual mind reading and propose a framework in which a user and a life-like agent mutually acquire mind mappings each other. In our framework, a user plays a mutual mind reading game with an agent and they gradually learn to read the other's mind. Eventually we implement our framework and make experiments to investigate its effectiveness.

1 Introduction

In these several years, a lot of studies have been done on a life-like agent like a software agent[7][6] and a robot agent[11][9]. A typical life-like agent appears on a Web shopping page and assists a user in inputting his/her order. Through the development of various life-like agents, an agent's *mind*, an internal state [1] representing emotion, processing load has been recognized to play a very important role in making them believable to a user[2]. Thus researchers are trying to implement a mind model on an agent for making it more believable[2][10]. However, even if a mind generation mechanism is fully implemented on a life-like agent, there is a significant problem that mind reading between a user and an agent is difficult.

For establishing effective communication between a life-like agent and a user, they need to be able to identify the other's mind from an expression and we call this task *mind reading*. If mind reading is impossible, they can not act human-like behaviors which significantly depend on the other's internal state. For example, a life-like agent should kindly and carefully behave to a depressed or busy user, and intuitively communicate its processing load to a user through a (facial) expression. Though mind reading is always done among human, it between a life-like agent and a user becomes far more difficult. Because design of agent's expressions depends on personal preference, social culture and so on. For example, Fig.1 shows various expressions and corresponding minds of Microsoft® agents, which is known as the most popular like-like software agent. We can easily identify minds from some expressions (Surprised, Congratulate in Fig.1), however minds from some expressions (Confused, Decline, Process in Fig.1) may be hard to be identified. Consequently a life-like agent and a user need to acquire relation between an expression and a mind when they actually encounter. We call such a mapping from an expression to a mind a *mind mapping*. Since this situation occurs independently of whether an life-like agent is implemented on software or a physical robot, human-robot interaction needs to cope with this problem of learning mind mapping mutually.

In this paper, we propose a human-agent interaction framework in which a user and a life-like agent mutually acquire mind mappings each other. They play a mutual mind reading game together and gradually learn mind mappings each other. Instance-based learning is applied to agent's learning. We fully implemented our framework on a PC with a CCD camera and eventually we make experiments for investigating mutual mind reading.

Velásquez[10] proposed a emotion model based on society of mind. His model is for generating human-like emotions using a multi-agent system architecture in which each agent corresponds to a primitive emotion and emotions are emerged as a result of the in-

[1] Theory of Mind has been developed in psychology, and our work is related with it. However we do not deal with a model for describing human mind, rather our term "mind" means a part of computational internal states of an agent and a human like states of processing load, reasoning, attention and so on.

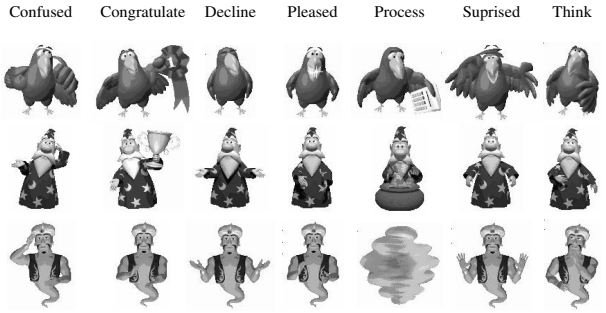

Figure 1 Various expressions of Microsoft agents.

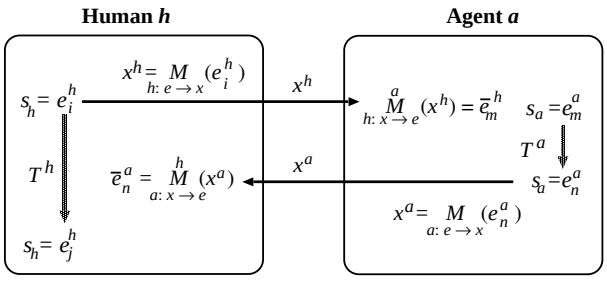

Figure 2 A framework for emotional interactions between a agent and a user.

teractions. However the purpose of his research is to generate emotions and moods like a human, and not to build a framework for interaction between an agent and a user.

Various researches of avatars have been done intensively on interaction with a man[3], and they found out interesting view points about communicative agents. However their purpose is to develop avatars that can naturally communicate with a human and our one is to design interaction between a human and an agent.

A lot of researches on facial expression recognition [5] have been done thus far. We can utilize these techniques to categorize sensed expressions. However our interest is concerned with mutual learning of mind mapping, and our research objectives is quite different from facial expression recognition.

Human robot interaction have been also studied actively. In particular, Ono and Imai proposed a cognitive model to describe how a human reads a robot mind and investigated its validity experimentally[9]. Though their work is excellent and interesting, it has no mutual learning of mind reading like our work.

2 Learning of Mind Mapping

In this section , we formalize our framework to deal with mutual mind reading between a agent and a user. First the following primitives are introduced.

- *Mind state* s_a, s_h: A variable s_a and s_h standing for a state of mind for an agent and a user respectively. A primitive mind is substituted for this variable.
- *Primitive mind* $E^a = \{e_1^a, \cdots\}$, $E^h = \{e_1^h, \cdots\}$: E^a and E^h are sets of m elements of agent's and a user's minds respectively. We can define these primitive minds depending on a particular task.
- *Primitive expression* $X^a = \{x_1^a, \cdots\}$, $X^h = \{x_1^h, \cdots\}$: X^a and X^h are sets of agent's and a user's primitive expressions.
- *Mind mapping* $M_{h:x\to e}^a = \{x_i^h \to e_j^h, \cdots\}$: This means a user's many-to-one mapping from primitive expressions to primitive minds which was learned by an agent. $M_{a:x\to e}^h$ means agent's mind mapping learned by a user.
- *Expression mapping* $M_{h:e\to x}$, $M_{a:e\to x}$: A user's (or an agent's) one-to-many mapping from primitive minds to primitive expressions.
- *Mind transition function* $T^a(c)$, $T^h(c)$: This function determines the next mind of an agent/user depending a context c. This context c may include its current mind, the other's current mind, success rates and so on.

Using the above notations, we describe a framework in which a life-like agent a and a user h interact through expressions as shown in Fig.2.

2.1 What should be learned?

With the framework described in Fig.2, we define learning of a mind mappings and mutual learning of mind mappings in the following.

Learning of a mind mapping: An agent(or a user) acquires the other's mind mapping $M_{h:x\to e}^a$ (or $M_{a:x\to e}^h$).

Mutual learning of mind mappings: An agent and a user mutually acquire the other's mind mapping, $M_{h:x\to e}^a$ and $M_{a:x\to e}^h$.

Since a designer is able to develop an agent by himself in practical situations, we can assume that the following parameters which are concerned with an agent are given. Primitive minds of a user may not be essentially determined by a designer. However we consider that an agent (or its designer) should determine primitive minds of a user because how an agent utilizes them is significantly dependent on the agent's ability.

- Primitive minds of a user and an agent.
- Primitive expressions of an agent.
- A mind transition function of an agent.

Except primitive minds of a user, we give no constrain to a user. A user freely learns an agent's mind mapping. Given the above parameters, the mutual learning of mind mappings is achieved by procedures described in the next subsection.

2.2 Learning in an agent

Because a user is able to autonomously learn an agent's mind mapping in our framework, we give no restriction to a user within his/her learning. Thus we develop only learning procedures of an agent.

Since primitive minds of a user are given, an agent does not need to acquire them. Also user's primitive expressions are obtained by categorizing captured images with a CCD camera. Hence if a user's primitive mind e^h is estimated when a user's expression x^h is observed, an agent acquires an instance of a user's mind mapping $x^h \rightarrow e^h$. After an agent stores sufficient such instances through interactions with a user, it becomes able to estimate a user's primitive mind from his/her observed primitive expression by instance-based learning[1] like a NN(nearest neighbor) method[4].

Agent Learning procedure

- $c \in C$, $c = (I, S)$: an instance.
- I_c: an attribute vector.
- V: a set of classes v.
- S_c: a sequence of latest n answer pairs. $S_c = [s_1, s_2, \cdots, s_n] = [(v_1, good), (v_2, nogood), \cdots]$

1. A new attribute vector I_{new} is given.
2. Investigate the most similar instance c_{sim} to I_{new} by computing the distance between the attribute vectors.
3. Determine a class $\hat{v} \in V$ using the following equation. Random selection is done for tie-breaking.
$$\hat{v} \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{s \in S_{c_{sim}}} g(v, s)$$
where $g(v, s) = 1$ if $s = (v, good)$, $g(v, s) = -1$ if $s = (v, nogood)$, and $g(v, s) = 0$ if no $(v, \_)$. If no instance in an initial period, determine $\hat{v}$ at random.
4. Indicate $\hat{v}$ to a user, and he/she answers YES or NO to $\hat{v}$.
5. If the answer is YES, add $(\hat{v}, good)$ into S of c_{sim}, and remove the oldest s from S if necessary.
6. If the answer is NO, add $(\hat{v}, nogood)$ into S of c_{sim} respectively, and remove the oldest s from S if necessary. Also if the distance between c_{sim} and I_{new} is over a threshold α, add a new instance $(I_{new}, [(\hat{v}, nogood)])$ to C.

When an agent guess a user's mind and shows it to a user in a later mutual mind reading game, he/she answers by "Yes" or "No" to the estimated mind. Thus an agent needs to utilize "No" answer which is not generally employed for instance-based learning. Since the number of classes (user's primitive minds) is usually over three, we can not determine which class the "No" answer is a positive instance to. Thus we modified a simple instance-based learning algorithm IBL2[1] to be able to deal with a "No" answer. When a "No" answer is given to an estimated primitive mind, an agent stores it as a new instance having negative evaluation to the estimated class. To deal with such negative evaluation, an agent assigns a set of recent evaluations and estimated minds to an instance and determines its class by a majority vote. Detail procedures of agent learning are shown in the following. In all the later experiments, we set parameters as $n = 2$, $\alpha = 900$ empirically.

2.3 Success rate and finish condition of learning

The success rate $r(e)$ for a primitive mind e is computed by the following equation. This success rate is also utilized to evaluate user's learning. The average value R of all $r(e)$ is used to indicate the progress of learning. $r(e) = \frac{\text{The number of success answer pairs in } S_e}{|S_e|}$ Finish condition for learning of an agent and a user is described as $R = 1$. This means recognitions of all primitive minds become complete when the condition is satisfied.

3 Mutual Mind Reading Game

A primal objective of a mutual mind reading game is to collect instances for agent's instance-based learning efficiently and broadly. An instance is a pair of an observed user's expression and an estimated primitive mind. In this paper, a game in which a player estimates the other's mind state through the (facial) expression to compete for the accuracy is called a *mutual mind reading game*. A problem of this game is that user's cognitive load becomes high. To solve this, this game is designed so that a user may enjoy it to play a part in collecting training data actively, and as results, the user's cognitive load becomes low.

Another objective of a mutual mind reading game is concerned with trust and motivation[8][6]. We consider that it is not a good idea to give a user an agent which fully learned a user's mind mapping from the start. On this matter, Schneiderman argued that such a sophisticated agent would give a user a feeling of loss of control and understanding and the user does not try to do modeling the agent[8]. Thus we believe that a user is effectively motivated through a mutual mind reading game.

Procedures of a mutual mind reading game are given in the following. Note that an agent tells its correct mind to a user when his/her answer is incorrect, but a user does not do so to reduce his/her cognitive load.

1. An expression of an agent is displayed to a user in GUI.
2. A user guesses agent's mind from seeing the expression, and tells the mind to an agent by clicking a button.
3. An agent replies "Yes" (the guess is correct) or "No" (the guess is incorrect) with the correct mind as judgment against the other's guess.

Figure 3 Environment of human-agent interaction.

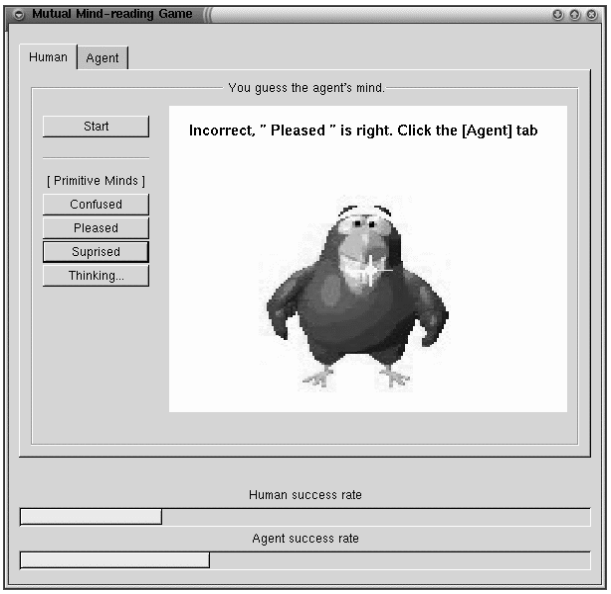

Figure 4 Human guesses agent's mind.

4. An agent sees an expression of a user by a CCD camera.

5. An agent guesses user's mind from the captured expression, and shows the mind to a user through GUI.

6. A user replies "Yes" (the guess is correct) or "No" (the guess is incorrect) as judgment against the other's guess.

7. The above procedures are repeated until a finish condition of mutual learning (described in 2.3) is satisfied.

4 Implementation

We fully implemented our framework. A system consists of a laptop computer (SONY VAIO-SR9G/K) and a CCD color camera (Creative Media: WebCam Plus) with USB. The resolution of the camera is 720×680 (8bit color). We used VineLinux2.1, C and GTK+. Also Video4Linux API was employed for image capture programming. An experimental environment is shown in Fig.3.

In a phase of agent's learning, an agent sequentially captures images of user expressions per 500ms, and obtains a stable expression. This stable expression means continuous four images with distance less than a threshold. We experimentally set the threshold as 250. When a stable expression is obtained, the head image is used as a captured image. This mechanism allows a user to control the timing to present his/her expression to an agent.

Captured image is transformed into an image with 40×30 with 8bit grey scale for an instance. Since computational cost depends on the size of an image, we used such a small grey image. Thus an instance is described by a vector with 256 values of 1200 dimensions. The similarity between instances is defined the Euclid distance.

We do not employ any feature detection for describing an instance. Because large computational cost makes system response slow and neither the best features nor the best detection method for any (facial) expression recognition has been developed. In stead, we consider that a user adaptively forms his/her expressions so that an agent can recognize them. This is user's adaptation to an agent, and agent's learning is agent's adaptation to a user.

Fig.4 shows a snapshot of GUI when a user guesses agent's mind. When a user clicks the "Start" button, an agent shows its expression. Then a user guesses agent's mind, and clicks one of "Primitive Mind" buttons. If a user clicks the button, an agent tells the judgment with the correct mind like a message in Fig.4. Also two progress bars are shown for indicating average success rates R (described in 2.3) of a user and an agent. A user can understand the degrees of learning progresses by seeing the progress bars. A game finishes when both of two progress bars reaches to the right edges.

Fig.5 shows interface where an agent guesses user's mind. When a user clicks a "Start agent's recognition" button, an agent begins to capture user's images. After a stable expression is captured, the four images are shown the window. Also stored instances are indicated with labels and the distance between them and a captured image. Using the most similar instance, an agent guesses a user's mind and tells it to a user like Fig.5. A user answers to it by clicking "Yes" or "No" buttons.

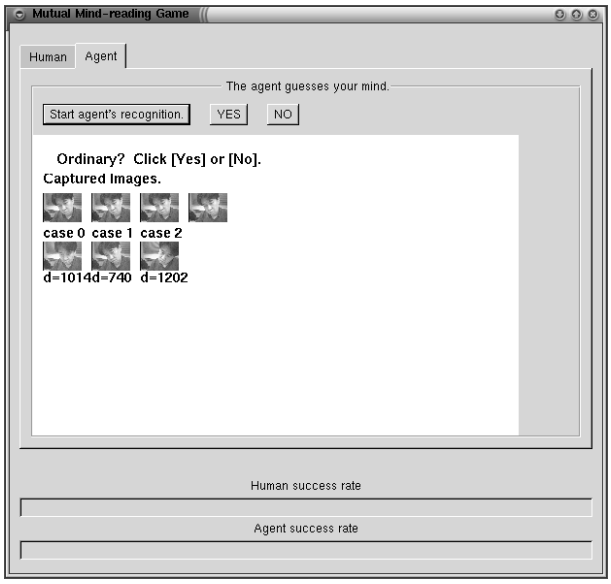

Figure 5 An agent guesses a user's mind.

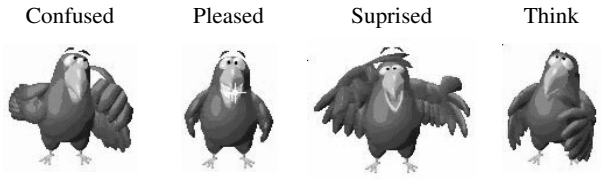

Figure 6 Four expressions used for experiments.

5 Experiments

5.1 Experimental methods

We made experiments to verify mutual learning between a user and an agent, and to investigate its characteristics.

Through all the experiments, we employed eight subjects consisting of five graduate students, three staff majoring Computer Science at Tokyo Institute of Technology.

We used four primitive minds and primitive expressions for an agent shown in Fig.6 and three primitive minds "Ordinary", "Thinking", "Decline" for a user. As the primitive minds increase, mutual learning becomes harder. We empirically consider the number of these primitive minds is valid for practical experiments.

Before experiments, we briefly gave subjects the following instructions. However we did not explain detail procedures of agent's learning, success rates and meanings of captured images, instance images in Fig.5.

- Rules of a mutual mind reading game.
- Explanation on GUI: meanings of two progress bars, buttons and tabs.
- Advise to affect user's expressions: it is effective to slightly rotate, tilt a head and touch a face. Due to agent's ability, fine expressions on a face is hard to be recognized.
- Three primitive minds "Ordinary", "Thinking", "Decline" for a user.

Also we set an agent's mind transition function $T^a(c)$ described in section 2 as the simplest one: random transition. This means an agent's mind changes into next mind randomly independently of context. We will improve this function later to make human learning more efficient.

Under the above conditions, each of eight subjects played a mutual mind reading game once with an agent and we investigated transitions of user's and agent's success rates, success rates for each primitive mind, the number of interactions until learning finished and real time taken for a game. We counted an interaction by a pair of agent's guess and user's guess in a game.

5.2 Observing mutual learning

Fig.7 shows representative results for success rates of a user and an agent. The two success rates gradually increases as interactions progressed, finally both of them converged to 1 and the game finished. Thus we are able to observe mutual learning of mind mappings (described in 2.1) in the experimental results. Since a user and an agent sometimes failed to guess the other's mind, increases of two success rates are not monotonic. In all the experiments, we observed such mutual learning of mind mappings between a user and an agent. A single game took about 5–15 minutes, and most subjects seemed to enjoy experiments.

The stored instances for three subjects s-1, s-2, s-4 are shown in Fig.8. In contrast that only three instances which were minimum to categorize three user's minds were stored for s-2, over five instances were stored for s-1 and s-4. For all the subjects, the numbers of stored instances have significant dispersion. Seeing the expressions in the instances in Fig.8, most of them were done by tilting a head or touching a face. Instruction to affect expressions might excessively restrict user's expressions.

Since our work is in an early stage, there are some limitations and open problems. In these experiments, the number of primitive minds were relatively small. Thus we can utilize a simpler method that we directly show a user a table like Fig.1 to remember mind mapping. However a user intends to feel "loss of control and understanding" in such a situation as Schneiderman claims. Thus we consider our approach of a game may outperform such a simple approach. While our method is applicable to a large number of primitive minds, mutual learning becomes very slow and we need additional methods to improve it.

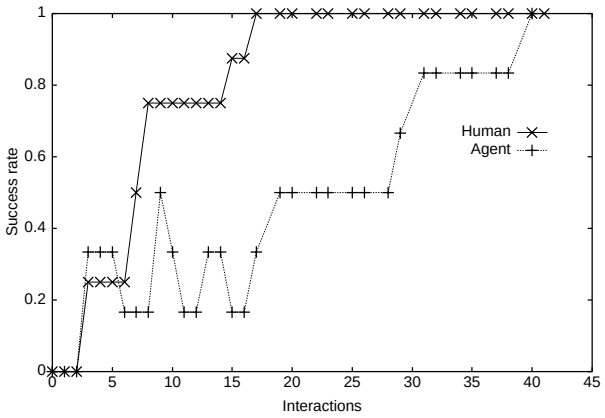

Figure 7 Representative mutual learning.

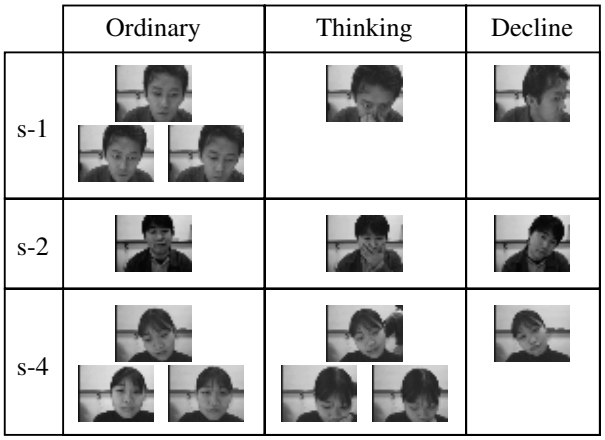

Figure 8 Stored instances of three subjects.

6 Conclusion

We proposed a human-agent interaction framework in which a user and a life-like agent mutually acquire their mind mappings through a mutual mind reading game. For describing mind interactions between a life-like agent and a user, we defined elements of our framework and developed agent's learning procedures by using an instance-based learning method. Then, to acquire the mind mapping each other, we developed a mutual mind reading game in which a user and a life-like agent try to recognize the other's mind from the other's expression. We implemented our framework and made preliminary experiments by employing subjects. As results, we found out mutual learning between a user and a agent through a mutual mind reading game and some characteristics of mutual learning.

A life-like agent in our framework is not necessary to be a software agent, may be a physical robot agent like a pet robot. Actually we are currently developing a system in which a user can interact with a SONY pet robot AIBO with a CCD camera. In such a case, an expression of an agent may be a gesture rather than a facial expression, and we need image processing like segmentation of a face region. We consider the human-robot interaction gives us much interesting phenomena between a human and a robot agent, and we try to modify our framework to deal with them effectively.

References

[1] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] J. Bates. The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125, 1994.

[3] J. Cassell. Embodied conversational agents: Representation and intelligence in user interface. *AI Magazine*, 22(4):67–83, 2001.

[4] B. V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, 1991.

[5] H. Kobayashi and F. Hara. Recognition of six basic facial expressions and their strength by neural network. In *IEEE International Workshop on Robot and Human Communication*, pages 381–386, 1992.

[6] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):30–40, July 1994.

[7] MS agent Web page. http://msdn.microsoft.com/msagent/.

[8] B. A. Myers, A. Cypher, D. Maulsby, D. C. Smith, and B. Shneiderman. Demonstrational interfaces. In *Proceedings of 1991 Conference on Human Factors and Computing Systems*, pages 393–396, 1991.

[9] T. Ono and M. Imai. Reading a robot's mind: A model of utterance understanding based on the theory of mind mechanism. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 142–148, 2000.

[10] J. D. Velásquez. Modeling emotions and other motivations in synthetic agents. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 10–15, 1997.

[11] J. D. Velásquez. An emotion-based approach to robotics. In *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 235–240, 1999.