The Benefits of Robot Deception in Search and Rescue: Computational Approach for Deceptive Action Selection via Case-based Reasoning

Jaeeun Shim and Ronald C. Arkin, *Fellow, IEEE*

Abstract—**By increasing the use of autonomous rescue robots in search and rescue (SAR), the chance of interaction between rescue robots and human victims also grows. More specifically, when autonomous rescue robots are considered in SAR, it is important for robots to handle sensitively human victims' emotions. Deception can potentially be used effectively by robots to control human victims' fear and shock as used by human rescuers. In this paper, we introduce robotic deception in SAR contexts and present a novel computational approach for an autonomous rescue robot's deceptive action selection mechanism.**

I. Introduction

The use of robots in SAR is rapidly growing and a large body of research has been conducted for developing rescue robots [1]. Several robots were previously used in SAR situations such as unmanned marine vehicles (UMV), unmanned ground vehicles (UGV), and unmanned aerial vehicles (UAV) in the Tohoku earthquake disaster [2], the World Trade Center disaster [3], and others. Most previous rescue robots focus on assisting human rescuers by supporting tasks such as structural inspection, searching, reconnaissance, and delivering supplies.

During the past few years, various novel robot systems and software were also proposed and developed for the DARPA robotics challenges [4]. They were expected to effectively support and work in natural and man-made disasters or emergency-response scenarios by focusing on specific rescue tasks, but not the interactions or relationships with human victims.

We argue that as increases in the use of autonomous robots in SAR occur, the chance of interaction between human victims and robots will increase as well. Thus, it is increasingly important for a rescue robot to interact with human victims naturally and effectively.

Human victims in SAR form a special population that needs to be handled differently from the everyday people typifying general human-robot interaction (HRI) studies. Accordingly, it is important for rescue robots to be capable of interacting with emotionally-sensitive humans in SAR contexts. One important capability for a rescue robot in this specific context is deception, specifically deception for the benefit of human victims.

J. Shim is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, USA (corresponding author to provide e-mail: jaeeun.shim@ gatech.edu).

R. C. Arkin is with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, USA (e-mail: arkin@cc.gatech.edu).

In a crisis, victims' emotional state can seriously affect their safety [5]. When victims' cooperation is required during the SAR tasks, managing their emotions is important. For this reason, rescuers sometimes hide the truth of the situation and act deceptively such as not describing the severity of injuries or the situation to victims accurately [6].

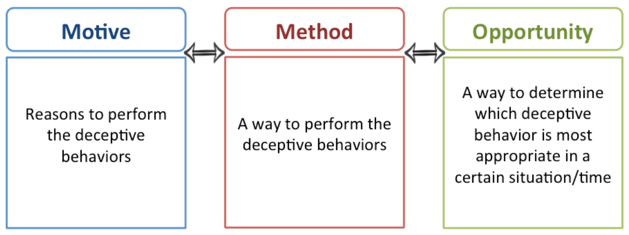
We also argue that a rescue robot can establish a better bond with human victims by having the capabilities of deception. As a goal, we aim for a rescue robot to perform a deceptive behavior only for the purpose of helping the deceived human victim. In our previous research [7], we defined this type of deception as other-oriented robot deception. Briefly, according to DePaulo [8], deception can be defined based on motivation, such as self-oriented and other-oriented deception. Self-oriented deception is used for the deceiver's own advantages. Conversely, other-oriented deception is motivated by the benefits that accrue to the person who is being deceived. We similarly classify a robot's deception capabilities into these two categories. Finally, we can formulate our research hypothesis as "*A rescue robot can perform other-oriented deception to benefit the deceived human victims in SAR situations.*"

To achieve a rescue robot's other-oriented deception, we first develop a computational model for the robot to determine when and how to perform those deceptive behaviors in SAR. For inspiration, we reviewed deception research in criminology [9]. According to the criminological approach, deception can be analyzed by three criteria, which are ***methods, motives***, and ***opportunities*** (Figure 1). *Methods* indicate the way to perform deception, where a novel algorithm for generating deceptive actions is developed. *Motives* indicate whether the current situation warrants the use of other-oriented deception. If so, the robot should determine specifically when the deceptive actions should be performed given an *Opportunity*. Inspired by this approach, a computational model of robot deception was developed for use in SAR.

We presented the deceptive action generation model in our previous research for the ***method*** part [10]. Briefly, we developed a novel algorithm for a robot to generate deceptive actions from the default true action based on two different deception mechanisms in psychology: deception by omission and deception by commission.

In this paper, as a next step, we present the computational models for the motive and opportunity parts. Importantly, this action selection mechanism should be adaptable since modeling deceptive action selection varies significantly by domains and users.

Figure 1. Criminology-inspired computational architecture for a robot's other-oriented deception


Motive
Reasons to perform the deceptive behaviors
Method
A way to perform the deceptive behaviors
Opportunity
A way to determine which deceptive behavior is most appropriate in a certain situation/time

II. Background

A. Human-robot interaction in SAR

As stated above, most SAR robotics research has focused on supporting human rescuers' specific tasks, and only a few studies have been done within the context of the victim. Nowadays, however, interests in HRI are growing rapidly within SAR contexts [1].

Several case studies were performed to establish a theoretical background for discussions of HRI in SAR contexts. Casper et al. [3] performed a practical evaluation by observing human operators' uses of SAR robots during the World Trade Center disaster. From the data collected during the robot-assisted response, they identified and analyzed interesting HRI events and finally made eleven recommendations meant to improve HRI in SAR contexts. Murphy [11] reviewed different practical cases and discussed HRI issues in SAR to formulate a theoretical background.

Most HRI studies in SAR are focused on interaction between human rescuers/operators and robots. For example, Drury et al. [12] developed a human-robot interface and evaluated its ability to support operators performing SAR missions. Yanco et al. [13] also performed usability tests for the two different interfaces and proposed recommendations for a new interface to be used in urban SAR. Another HRI-based study proposed and evaluated an architecture for assisting human-robot teaming in SAR [14]. A recent study analyzed the interfaces and interaction methods used in DARPA robotics challenges [4]. As evidenced here, previous research has generally evaluated the quality of interactions between operators and rescue robots. In contrast to those studies, our research aims to improve relationships between rescue robots and human victims.

B. Deception in Robotics

We intend to add deceptive capabilities to the rescue robots with a goal of providing better outcomes for human victims. Limited research related to robotic deception has been done not only in the SAR context but in other domains in general.

Deceptive behaviors are commonly observed in animals, and several of these deceptive behaviors are applied to robotic systems. For example, inspired by animals, a camouflage soft robot was developed at Harvard University [15]. Carey et al. [16] developed an optimal control mechanism based on motion camouflage of dragonflies. Many animals also use deceptive behaviors to mislead predators or competitors. Squirrel's food protection behavior includes an interesting deception mechanism that was applied to a robotic system [17]. Another study explores the role of deception according to Grafen's dishonesty model in the context of birds' mobbing behavior and was applied to a robotic system successfully [18].

Several robot deception projects have been conducted in HRI contexts. In many cases, deception is used for engaging a user's attention. For example, a deceptive robot referee in a multi-player robotic game shows an increase in users' engagement and enjoyment [19]. A cheating robot in the context of a rock-paper-scissors game also illustrates increased engagement [20]. According to recent work [21], a deceptive robot assistant can also improve the learning efficiency of children.

Deception has been successfully used in a robotic physical therapy system [22]. By giving deceptive visual feedback on the amount of force patients currently exert, patients can perceive the amount of force to be lower than the actual amount. As a result, patients add additional force and gain the benefits during the rehabilitation.

C. Case-based Reasoning in Robotics

This paper describes how we are embodying a computational model that enables a robot to choose a beneficial behavior from either the true or deceptive action set based on other-oriented deception. Since modeling deceptive action selection varies significantly by situation and users, the computational model should be able to adapt the action selection mechanism, and consequently, it uses a learning-based computational approach.

Learning methods can be classified in two ways: eager-learning and lazy-learning models [23]. Eager-learning methods find generalization during training (e.g., reinforcement learning), and therefore, they require more training experiences to converge to an optimal solution. However, this long training time is impractical in a SAR situation. Instead, a lazy-learning method is performed at the instance-query time. In the initial stage, preloaded or previously acquired cases can be used to solve the new problem, and the cases can be adapted or updated through experience. Therefore, even though it may include some initial generality issues, it is more feasible when applied to our computational model and domains. For this reason, we build our computational model based on one of the well-known lazy-learning mechanisms, the case-based reasoning (CBR) mechanism [24], which has already been applied in several robotic systems successfully [25, 26, 27].

In the CBR process, the instances of a situation-solution pair are stored in the memory as a "case," and the CBR cycle can be illustrated in terms of four process stages [28]:

- <u>Case retrieval</u>: when a problem arises, the best matching case is searched to retrieve an approximate solution.
- <u>Case adaptation</u>: the approximate solution from the case retrieval is adapted to the new problem.
- <u>Solution evaluation</u>: either before or after the solution is applied, the adapted solution can be evaluated.
- <u>Casebase updating</u>: based on the verification of the solution, the case may be updated or a new case may be added to the casebase.

III. Computational Model

In our model, a rescue robot should be able to select an appropriate action from a set of true/deceptive actions for a given situation. The model should store the information of situation-action pairs that increases/decreases the human victim's benefits. For this purpose, we develop a computational model for a robot to select and learn the most appropriate true/deceptive action via CBR.

*A. Case **C***

A previously experienced situation-action pair is stored as a case in the memory. In addition, how much the action can benefit a human victim is also contained in the case. Therefore, a situation (s) – action (a) – benefits (r) trio is required to define a case as follows:

$$C = \{\ [s, a, r\ ]\ |\ s \in S,\ a \in A,\ r \in R\}$$

<u>State ***S***</u>

A robot should observe the current SAR situation as an input and then should select and perform an appropriate true/deceptive action as an output. The situational state should express the current state of the human victim during the interaction and also the environment. Therefore, state s is defined as the combination of features, which can represent victim's internal ($f_{m1}, \ldots, f_{mj}$) and environmental ($f_{e1}, \ldots f_{ek}$) conditions. Finally, we can define the set of situational state for our CBR mechanism as shown below:

$$S = \{< f_{m1}, \ldots, f_{mj}, f_{e1}, \ldots f_{ek} >\ |\ f_i = \textit{feature to perceive victim's internal or environmental conditions}\ \}$$

Features can be valued as '*don't care*' if the system determines that they do not play a key role in state discrimination.

<u>Action ***A***</u>

In previous work [10], we developed a novel algorithm to generate a robot's action set that includes the available true and alternative deceptive actions. Briefly, a robot can find and generate the action set based on general/emotional action primitives. A robot action is defined as the combination of different action cues; $a = <g, f, p>$, where g is the bodily gesture cue, f is the facial expression cue, and p is the proximity cue. The primitives of each action cue are illustrated in our previous research [10]. When a robot's true action a_t is determined, it can generate the alternate deceptive actions based on the characteristics of each primitive. This deceptive action generation is possible according to two mechanisms, namely deception by commission and deception by omission. Finally, from the n default/true actions and alternative m deceptive actions generated, the set of actions can be defined as $A = \{a_{t1}, \ldots, a_{tn}, a_{d1}, \ldots, a_{dm}\}$.

<u>Benefit ***R***</u>

The goal is to find the rescue robot's action that maximizes the benefit for the human victim. Therefore, how much benefit a human victim can obtain from a robot's true/deception action is an essential element in the case. To express those human benefits, we define a measure R such as:

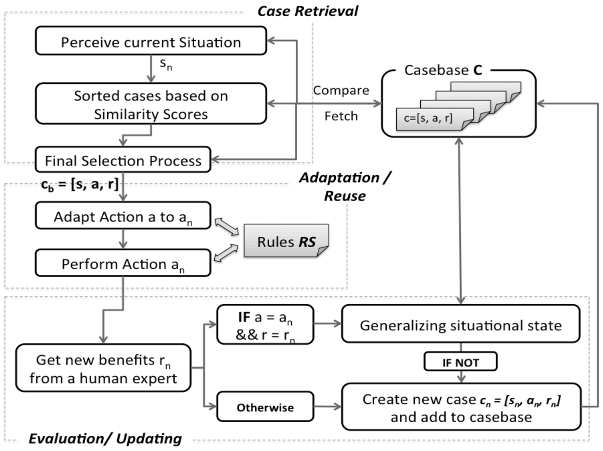
$$R = \{r\ |\ r \in \mathbb{Z}\ \wedge\ R_w \leq\ r\ \leq\ R_b\}.$$

Here, $\mathbb{Z}$ is the set of integers. R_w is the minimum limit for negative benefits and R_b is the maximum limit for positive benefits.

B. Case-based Reasoning Process

Overall, our deceptive action-selection model via CBR can be presented in Figure 2. Details for each step will be explained in this section.

Figure 2. Computational architecture for the motives/opportunities model



Case Retrieval
Perceive current Situation
s_n
Sorted cases based on Similarity Scores
Compare
Fetch
Casebase **C**
c=[s, a, r]
Final Selection Process
c_b = [s, a, r]
Adaptation / Reuse
Adapt Action a to a_n
Perform Action a_n
Rules *RS*
IF a = a_n && r = r_n
Generalizing situational state
IF NOT
Get new benefits r_n from a human expert
Otherwise
Create new case $c_n = [s_n, a_n, r_n]$ and add to casebase
Evaluation/ Updating

<u>***Step 1. Case Retrieval***</u>

In the case retrieval step, the system should find which case(s) has the most similar situational state to the current perceived state. The case retrieval algorithm scores how similar the current situational state is to other cases in the casebase. By calculating this similarity score, the algorithm can select the best-matched case or cases in this step.

<u>*Similarity metric*</u>

To determine the initial matched cases, a similarity score is calculated that measures how much the state in the case is similar to the current situation. For this, we use a syntactic similarity assessment, which provides a global similarity metric based on surface match. For computing this score, a Manhattan distance calculation is used, similar to [27], by calculating the L-1 norm between the two feature vectors of the situational states. Since not all features in the state are equally important, each feature is weighted to indicate its relative importance. $\Phi(s_p,\ s_c)$ is defined as the similarity metric between states s_p and s_c as shown in equation (1) when the state vectors are illustrated as $s_p = <f_{1,p}, f_{2,p}, \ldots., f_{n,p}>$ and $s_c = <f_{1,c}, f_{2,c}, \ldots, f_{n,c}>$.

$$\Phi(s_p,\ s_c) = \frac{\sum_{i \in S} k_i \cdot w_i}{\sum_{i \in S} w_i} \tag{1}$$

Here, k_i is a similarity score for each feature f_i and w_i is a weight for each feature f_i. Here, the feature valued '*don't care*' in s_p indicates that it can be matched with any feature value in state s_c. Therefore, the similarity score for this feature can be the maximum value. Weight w_i should be empirically set by a human expert's domain knowledge. Similarity score k_i is defined as $k_i = \{x\ |\ x \in\ \mathbb{R}\ \wedge\ 0\ \leq x\ \leq 1\}$ and it can be calculated differently based on the characteristics of each feature f_i as shown in Algorithm 1.

Algorithm 1. Calculating similarity score k_i for feature f_i

```
IF feature f_i is 'don't care'
    $k_i = 1$

ELSE IF feature f_i is a numeric datum
    $k_i = 1 - \left(\frac{|f_{i,c} - f_{i,p}|}{v_i}\right)$      where $v_i$ is the range for feature $f_i$

ELSE IF feature f_i is a categorical datum AND
        Cardinality of set $F_i \leq 2$ where $f_i \in F_i$
    $k_i = \begin{cases} 1 & if\ f_{i,p} = f_{i,c} \\ 0 & otherwise \end{cases}$

ELSE IF feature f_i is a categorical datum AND
        Cardinality of set $F_i \geq 3$ where $f_i \in F_i$
    $k_i = M_i(f_{i,p}, f_{i,c})$   where $M_i$ is the $f_i$-specified similarity table
```

As described in Algorithm 1, when feature f_i is a categorical datum, categorical matching can be used. However, if there are more than two categories, there may be shades of similarity; one category might be more like the one of the other categories than another. To specify those shades of similarity we define the feature-specified lookup table M_i based on the characteristics of each feature, and similarity scores for feature f_i can be determined by the corresponding entry in M_i.

After calculating $\Phi(s_p, s_c)$ for each case in the casebase, the cases are sorted from the highest scored case to the lowest one, and the top n cases are selected as the initial matched cases and transferred to the case adaptation step.

Step 2. Case Adaptation and Reuse

Once the initial matched cases are selected, the final selection process is required to find the best-case $c_b = [s, a, r]$ to use. Even though a robot finds the best-case, there is no reason to reuse this case's action if the benefits of the action are poor. Therefore, as a final selection step, a robot should assess the previous benefit and determine whether it should be selected for execution. After selection, the corresponding action should be adapted and passed to the robot to perform the robot behavior. This final action selection process can briefly be explained as follows:

[**Step 1**] Select the top n cases from the sorted casebase as the initial matched cases

[**Step 2**] From the n cases, discriminate the cases that have a positive benefit ($r > 0$)

> [**Step 3-1**] If several cases have a positive benefit, choose one case using the weighted roulette wheel algorithm (the likelihood of a case being chosen are determined based on the relative benefit of that case) and set it as the best matched case $c_b = [s, a, r]$
>
> [**Step 3-2**] If none have a positive benefit, retrieve next n cases and repeat the case selection process from **step 2**

[**Step 4**] From the selected case $c_b = [s, a, r]$, adapt action a to the adapted action a_n (see algorithm 3 below)

[**Step 5**] If no case can be found after repeating to the last case, assign the default action (which is the true action a_t) to the adapted action a_n

Algorithm 2. Final Action Selection Process

```
Input:   Sorted casebase C
         Current situation $s_n$
Output:  Final adapted solution action $a_n$
-----------------------------------------------
startIndex = 0
// starting index of casebase when picking top n cases
LOOP:
  nCases = {}             // reset nCases, which stores the top n cases
  FOR i in startIndex to startIndex+n          //top n ranked cases
    IF c[i].r ≥ 0                   // find the cases w/ positive benefit
      nCases.add(c[i]);             // if positive, add to set nCases
  IF |nCases| > 0
    Find the best case from nCases
    using the weighted roulette wheel algorithm
    $c_b$ = selected best-case
    $a_n$ = adaptingAction($c_b$, $s_n$)   // adapting the action (Algo. 3)
    return $a_n$;
  ELSE                              // If none have a positive benefit
    startIndex += n;                // set startIndex to the next n cases
    goto LOOP;                      // retrieve n more cases
$a_n$ = $a_t$;                      /* return the default action as a solution
return $a_n$;                          if no cases can be selected */
```

Table 1. Data structure for the rule

Field	Description
FeatureIndex	Indicates which feature should be observed for the current adaptation rule
Origin	The origin of the adaptation rule (reference)
Activity	Indicates if the rule is currently active
Description	Short, concise description of the rule
AdaptationCondition	Condition for determining whether the adaptation rule should be applied to the current action
AdaptationRule	Formal expression defining the way to adapt the current action

More specifically, we can illustrate this final action selection process as shown in Algorithm 2. After the best-case is determined ($c_b = [s, a, r]$), it may need to be adapted to the current situation. If the current state is exactly matched to the situational state in the best-case, we can directly use action a in the case without adaptation. However, if those two states are slightly different but the case is selected since it has the most similar situational state, then the action may be adapted before application.

According to Kolodner [28], adaptation involves two major steps: 1) figuring out what needs to be adapted and 2) performing the adaptation. Similar to this approach, the algorithm first observes whether there are differences between best case's state s and current state s_n, and if the two states are different each other, an appropriate adaptation should be applied. Differences between states can be discriminated by finding features that specifically have different values (figuring out what needs to be adapted). When the features having different values are determined, the adaptation rules associated with those features are applied (performing the adaptation). In addition, adaptation can be varied for any particular domains or tasks, and a set of adaptation strategies or heuristics can be used for a CBR working system [28]. Therefore, the adaptation process is designed based on predefined adaptation rules. For this, we define the data structures of the rules that encode the adaptation procedures (from experts or literature) that instruct

```
Algorithm 3. Adaptation Process: adaptingAction(c_b, s_n)
Input:   best-case $c_b = [s, a, r]$ where $s = <f_{1,s}, f_{2,s}, …, f_{k,s}>$
         Current state $s_n = <f_{1,sn}, f_{2,sn}, …, f_{k,sn}>$
         Set of rules $RS = \{rs_1, rs_2, rs_3, … , rs_n\}$
Output:  Adapted action $a_n$
// determine whether the current state  is same as the case's state
IF $s == s_n$            // if two states are same,
  $a_n = a$              // set the same action a as an adaptation action
ELSE                     // if two states are different, start adaptation
  D = {}                 // variable to store the set of feature indices
  /* Step 1. Figuring out what needs to be adapted */
  FOR all features in s
    IF $f_{i,s}$ != $f_{i,sn}$   // if the value of feature $f_i$ in case's state s is
                           // different from current state $s_n$, store feature
                           // index i into the set D
      $D \leftarrow D \cup \{i\}$
  /* Step 2. Doing the adaptation */
  FOR all feature index i in D
    FOR all rules $rs_j$ in RS   // among all rules in RS
      IF $rs_j$.featureIndex == i AND
         $rs_j$.adaptationCondition == True
        // find the rule $rs_j$ that contains the feature index i
        // if $rs_j$'s adaptation condition is satisfied, adapt by the rule
        $a_n \leftarrow$ Perform $rs_j$.adaptationRule;
return $a_n$;
```

the robot how to adapt the selected action to the current situation. The data structure of the rules is shown in Table 1.

The adaptation process using the rules is summarized in Algorithm 3. Again, when the state in the selected best-case is exactly same as the current situation, action *a* in the case can be directly used as a solution. However, when the two situations are different, the adaptation process is called. As shown in Algorithm 3, the system first compares all feature values between the case state and the current state and if the two values are different, those feature indices are compared to the element in the set of rules *RS*. In other words, if the discriminated feature index is matched with the element of *FeatureIndex* in certain rule $rs \in RS$, this rule's *AdaptationRule* applies based on *AdaptationCondition*, and an adapted action a_n will be generated.

Case application

From the retrieval and adaptation steps, a robot can determine an appropriate adapted action a_n. Then, this adapted action a_n should be performed by the robot. This step is known as the case application step. When the adapted case is applied (action a_n is determined and performed), we should perceive the changes of human victim's status for the next evaluation/updating steps.

Step 3. Solution Evaluation and Casebase Updating

Cases are initially created by experts or experienced users. Even then, the pair of situational state and action in each case may not always be the best solution. During the evaluation/updating phase, the cases should be revised and updated to the most effective solution. In other words, it should be observed whether a human victim truly receives benefit from a rescue robot's behavior, and the casebase should be updated with the case that can generate the highest payoff for the human victims.

After performing an action (case application), the benefits should be identified to evaluate the action post facto. It can be determined by a human expert afterwards or if possible assessed autonomously by a robot. In our system, a human expert will rate the benefits since it is a more practical and accurate way in the real-world situation. When the human expert determines the benefits, they should be asked to observe the changes of the human victim's condition after performing a rescue robot's adapted action and rate the benefits within the predefined range of benefits R.

Updating casebase

Based on the new benefit r_n, which is provided by a human expert, the CBR system should update the cases if necessary. According to Kolodner [28], the casebase can be updated by 1) *generalizing* the cases in the casebase, or 2) *storing* the new case. Similar to this approach, when the current state s_n, the adapted action a_n, and the new benefit r_n are perceived, they can be used to generalize existing cases in the casebase. Otherwise, it will be stored as the new case for future reference.

First, the system should determine whether generalization is possible with s_n, a_n, and r_n. In our system, states will be generalized when the cases have the same action as a_n and benefit as r_n. Since states are represented as different features, the system can generalize the state by finding/merging the features that may not play key roles during the calculation of similarity scores. For the generalization process, those unnecessary features will be minimized to a '*don't care*' value.

This feature reduction can be solved using the minimization algorithm of the algebraic variables. When all the features are boolean variables, the '*don't care*' feature can be determined by the Quine–McCluskey (Q-M) algorithm [29]. Let us assume that we have the state representation such as $s = < f_1, f_2, f_3 >$ and each feature is defined as $f_1, f_2, f_3 \in \{1, 0\}$. When we have two existing states s_1 = <1, 1, 1> and s_2 = <1, 1, 0> and the new state s_n = <1, 0, 1>, those three states can be generalized using an algebraic minimization algorithm. First, states s_1, s_2, and s_n can be represented as $f_1 f_2 f_3$, $f_1 f_2 \overline{f_3}$, $f_1 \overline{f_2} f_3$ using a canonical form. Via the Q-M algorithm [29], the states can be generalized to the canonical sum of products expression by summing the minterms such as $f_1 f_2 f_3 + f_1 f_2 \overline{f_3} + f_1 \overline{f_2} f_3 = f_1 f_2 + f_1 \overline{f_2} f_3$. Since '*don't-care*' terms are left out of the minterms, we finally get the generalized states such as <1, 1, *don't care*> and <1, 0, 1> from minterms $f_1 f_2$ and $f_1 \overline{f_2} f_3$.

Multi-valued features can be handled by the same mechanism via the extended Q-M algorithm [30]. Again, let us assume the state $s = < f_1, f_2, f_3 >$ and features are defined as $f_1 \in \{1, 2, 3\}$, $f_2 \in \{a, b, c\}$, $f_3 \in \{x, y\}$. When we have three existing states s_1 = <1, a, x>, s_2 = <1, b, y>, s_3 = <1, a, y> and the new state s_n = <1, c, y>, the states can be generalized by the extended Q-M algorithm. First, the canonical forms of states s_1, s_2, s_3, and s_n will be $f_1^1 f_2^a f_3^x$, $f_1^1 f_2^b f_3^y$, $f_1^1 f_2^a f_3^y$, $f_1^1 f_2^c f_3^y$, and those four states can be minimized such as $f_1^1 f_2^a f_3^x + f_1^1 f_2^b f_3^y + f_1^1 f_2^a f_3^y + f_1^1 f_2^c f_3^y$

```
Algorithm 4. Casebase Updating Strategy
Input: Current State $s_n$
       Adapted Action $a_n$
-------------------------------------------------------------
Determine the new benefit $r_n$ from a human expert

/* Step 1.  Generalizing the cases */
candidateCases = {};  // cases that are potentially generalizable
FOR each case c in Casebase C
    IF c.r == $r_n$ && c.a == $a_n$  // if the case has the same action and
                        // benefits as $a_n$ and $r_n$, it is potentially generalizable
        candidateCases.add(c);  // add to the candidateCases

// Generalize states via minimization algorithm [29, 30]
allMinterms =   // get minterms via feature reduction algorithm
    ExtendedQ-M (canonical forms of  candidateCases' states,
                 canonical form of $s_n$ );
FOR each minterm$_i$ in allMinterms
    $s_{generalizeed}$ = extract from minterm$_i$ by adding 'don't care' terms
    Remove all cases in candidateCases
    Add generalized case [ $s_{generalizeed}$, $a_n$, $r_n$]

/* Step 2.  Storing the new case */
IF no cases are generalized
    Create new case $c_n$ = [$s_n$, $a_n$, $r_n$]     // new case created
    Add $c_n$ to the case base                     // updated
```

= $f_1^1 f_2^a f_3^x + f_1^1 f_3^y$. Therefore, we finally have the generalized two states, <1, *a*, x> and <1, *don't care*, y>.

Algorithm 4 describes the entire process of the updating strategy in pseudocode. For the ***generalization*** process, the system will first select all cases that have the same action as the adapted action a_n and the same benefit as the new benefit r_n. The states from the selected cases and the current state s_n are then generalized by reducing the features. This feature reduction is performed using the minimization algorithm [29, 30]. After the reducible features are determined, those features are valued as '*don't care*,' and the cases are generalized. If none of the cases are merged and generalized in the previous step, a new case $c_n = [s_n, a_n, r_n]$ is created and ***stored*** in the casebase for future use. By using this updating strategy, the casebase can maintain the best situation-action pairs that can provide the largest benefit to human victims.

IV. Exemplar Scenario

We proposed the deceptive action-selection model via CBR in section 2. In this section, we will review our model with a specific SAR example; the scene of a fire.

Case Representation

We first define the case as $c = [s, a, r]$, where s is a situational state, a is an action, and r is the benefit. State s in our example can be simply defined as $s = < f_{m1}, f_{m2}, f_{m3}, f_{m4}, f_{e1}, f_{e2} >$ where features indicate victim's internal and external conditions as shown in Table 2.

A set of actions contains the appropriate true and deceptive actions for each state. A true/default action is specifically defined for each state based on the literature or expert perspectives. Then, from the true action, the deceptive actions are generated by our deceptive action generation mechanism [10]. In this scenario, the true action a_t can be defined as a_t = [v_t, <egp_t, f_t, p_t>]. Here, v_t is the verbal cue, which explains the current status to the human victim. Nonverbal cues < egp_t, f_t, p_t > represent the emotional gesture

Table 2. Data structure of state S

	Description
f_{m1}	{x | x = 1 if $10 < r_1 < 30$, otherwise x = 0} : Normal or abnormal respiration : r_1 from respiratory sensor (breaths per minute, *bpm*)
f_{m2}	{x | x = 1 if $60 < r_2 < 100$, otherwise x = 0} : Normal or abnormal ranges of pulse : r_2 from pulse rate sensor (pulse beats per minute, *pBPM*)
f_{m3}	{x | x = 1 if $60 < r_3 < 100$, otherwise x = 0} : Normal or abnormal range of heartbeat : r_3 from heart rate sensor (heart beats per minute, *hBPM*)
f_{m4}	{x | x ∈ {anger, disgust, fear, happiness, sadness, and surprise}} : Current emotional state from speech and pitch detection
f_{e1}	{x | x=1 if $14 < r_5 < 32$, otherwise x = 0} : Normal or abnormal range of room temperature : r_5 from digital temperature sensor (Celsius, *°C*)
f_{e2}	{x | x=1 if r_6 == true, otherwise x = 0} : Gas detected or not, r_6 from CO-gas detection sensor

primitive, facial expression, and proximity, respectively, where these values are matched to the current environmental status. After determining true action a_t, the set of possible deceptive actions are generated through the deception by omission and commission mechanisms. For example, situation s = < 1, 1, 1, anger, 1, 1> can have a true action such as a_t = [v_t, <egp_{pos}, f_{pos}, p_{pos}>] since the environmental features f_{e1} and f_{e2} determine the positive state (e = *pos*). Then, the deceptive actions are generated such as a_{d1} = [v_d, <egp_{neg}, f_{neg}, p_{neg} >] using deception by omission and a_{d2} = [v_d, <egp_{null}, f_{null}, p_{neg} >] using the deception by commission mechanism. Finally, the exemplar situation can have a set of actions such as {a_t, a_{d1}, a_{d2}}. For each state, a set of actions is generated via the deceptive action generation mechanism. In addition, the measure of benefits is defined as $R = \{r \mid r \in \mathbb{Z} \land -3 \leq r \leq 3\}$ via the triage process [31].

Initial Casebase

The initial casebase is determined manually and the benefits are filled by the experts. They are asked to rate the benefits of the action in the range of R by predicting how much the human's state will be improved or worsened in each case. Table 3 is the initial casebase in our exemplar scenario.

Table 3. Initial Casebase

CaseID	State						Action	Benefit
	f_{m1}	f_{m2}	f_{m3}	f_{m4}	f_{e1}	f_{e2}		
1	1	1	1	anger	1	1	a_t	-1
2	1	1	1	disgust	1	1	a_{d1}	+1
3	1	1	1	fear	1	1	a_{d2}	+2
4	1	1	1	happiness	1	1	a_t	+3
5	1	1	0	sadness	1	1	a_{d1}	0
6	1	0	0	anger	0	0	a_{d2}	-3
7	1	0	0	disgust	0	0	a_t	+1
8	0	0	0	fear	0	0	a_{d1}	-2
9	0	0	0	happiness	1	0	a_{d2}	-3
10	0	0	0	sadness	0	1	a_t	-1

Case Retrieval

Next, a robot should perceive the actual current situation and compare it to the previously stored cases in the casebase. Further assume that a robot rescuer perceives the current state such as s_n = < 1, 0, 0, fear, 1, 1 >. Then, as the first step of case retrieval, the similarity scores of each case in the casebase, should be calculated by Algorithm 1. In this example, similarity score k_i for each feature f_i is determined by the following algorithm.

```
For each feature $f_i$ in $i = m_1, m_2, m_3, m_4, e_1, e_2$
    IF $i == m_4$              // for emotional feature $f_{m4}$
        Find $k_i$ from similarity lookup table $\mathbf{M}_{emotion}(f_{i,sp}, f_{i,sc})$
    ELSE                       // for all other features
        IF $f_{i,sp} == f_{i,sc}$   THEN   $k_i = 1$
        ELSE                   THEN   $k_i = 0$
```

Table 4. Similarity score lookup table $\mathcal{M}_{\mathbf{emotion}}$

	anger	fear	disgust	sadness	happiness	surprise
anger	**1**	**0.5**	**1**	**0.5**	**0**	**0**
fear	**0.5**	**1**	**0.5**	**0.5**	**0**	**0**
disgust	**1**	**0.5**	**1**	**0.5**	**0**	**0**
sadness	**0.5**	**0.5**	**0.5**	**1**	**0**	**0**
happiness	**0**	**0**	**0**	**0**	**1**	**0.5**
surprise	**0**	**0**	**0**	**0**	**0.5**	**1**

Here, the similarity lookup table $\mathcal{M}_{\mathbf{emotion}}$ is generated based on Emotion Annotation and Representation Language (EARL) classification [32] as shown in Table 4.

In addition, features should have different weights based on their relative importance. According to [33], in the context of a fire, rescuers primarily check the risk of the scene based on the temperature. Furthermore, the main purpose of deceptive behaviors in a crisis situation is managing the human victim's emotions [6]. Consequently, we empirically set the weight vector such as w = <1, 1, 1, 2, 2, 1>. Now, we can get the similarity scores for each case and also the rank of similarity as Table 5. With the similarity scores, we find one best-case via the final selection process (Algorithm 2).

Table 5. Calculating similarity scores and sorting

CaseID	State S						similarity	Rank
	f_{m1}	f_{m2}	f_{m3}	f_{m4}	f_{e1}	f_{e2}		
2	1	1	1	disgust	1	1	0.75	1
3	1	1	1	fear	1	1	0.75	1
5	1	1	0	sadness	1	1	0.75	1
1	1	1	1	anger	1	1	0.625	2
6	1	0	0	anger	0	0	0.625	2
7	1	0	0	disgust	0	0	0.625	2
4	1	1	1	happiness	1	1	0.5	3
8	0	0	0	fear	0	0	0.5	3
9	0	0	0	happiness	1	0	0.5	3
10	0	0	0	sadness	0	1	0.5	3

Adaptation and Case Application

From the final selection process (Algorithm 2), the system determines case 3 as the best case such as $c_b = [s, a, r] = [<1, 1, 1, \text{fear}, 1, 1>, a_{d2}, +2]$. After, it should be adapted to the current situation. To apply the adaptation algorithm, we first must have a set of predefined adaptation rules *RS*. In our example, we predefine the set of rules as shown in Table 6.

Now, we should adapt the action. According to Algorithm 3, the differences between the best-case state and the current state should first be discriminated. In our example, s and s_n are different in features f_{m2} and f_{m3}. Then, it should be adapted from the set of appropriate predefined rules *RS*. Rule 1 should be chosen, since this rule's feature indices include *m2* and *m3*. Then, since the current situation as $s_n = < 1, 0, 0, \text{fear}, 1, 1 >$ satisfies the adaptation condition ($f_{m2,n} == 0$ && $f_{m3,n} == 0$) of rule 1, the adaptation rule will apply and the adapted action $a_n = a_t$ will be used as a solution.

Table 6. Rules for adaptation

Rule 1. Extreme condition of human victim

If vital features are different and the current values are false, we should consider victim's life-threatening status and adapt the action.

```
<rule> rs_extreme_internal_condition
   <feature Index> m1 | m2 | m3 </feature Index >
   <origin> Lois' Article [6], IAFC's 10 Rules [33] </origin>
   <activity> True </activity >
   <description> Perform the true action when human victims
    are in an extreme condition. </description>
   <adaptationCondition> (f_m1,n == 0 && f_m2,n == 0) ||
   (f_m1,n == 0 && f_m3,n == 0)  ||  (f_m2,n == 0 && f_m3,n == 0)  ||
   (f_m1,n == 0 && f_m2,n == 0 && f_m3,n == 0) </adaptationCondition >
   <adaptationRule> a_n = a_t </adaptationRule >
</rule>
```

Rule 2. Risky environmental condition

When the features for the environmental state are different and the feature values from the current situation are false, we should consider it is a risky situation and adapt the action.

```
<rule> rs_risky_external_condition
   <feature Index> e1 && e2 </feature Index >
   <origin> Lois' Article [6], IAFC's 10 Rules [33] </origin>
   <activity> True </activity >
   <description> Perform the true action if the environment is
    very risky.</description>
   <adaptationCondition> f_e1n == 0 && f_e2n == 0
   </adaptationCondition>
   <adaptationRule > a_n = a_t </adaptationRule >
</rule>
```

Rule 3. Contradictions of emotional states

If the victim's emotional status is totally different in f_{m4} and f_{m4n}, it is determined as emotional contradiction, and the corresponding action should be adapted. Therefore, if f_{m4} and f_{m4n} are in the different categories (negative/positive), the adaptation will be performed by regenerating the neutral gesture primitive and facial expression.

```
<rule> rs_avoid_contradiction
   <feature Index> m4 </feature Index >
   <origin>  Lois' Article [6], Shim's Article [10] </origin>
   <activity> True </activity >
   <description> Emotions are contradictory. </description>
   <adaptationCondition> (f_m4 ∈ E_negative && f_m4n ∈ E_positive) ||
    (f_m4 ∈ E_positive && f_m4n ∈ E_negative) </adaptationCondition>
   <adaptationRule > a_n = <egp_n, f_n, p> </ adaptationRule >
</rule>
```

Evaluation and Case Update

After the case application, the adapted action should be evaluated to update the case. A human expert is asked to evaluate the change of victim's state and rate the benefits gained, if any. After getting the new benefit r_n (+2 in this example), the update algorithm should be applied. In this step, by following Algorithm 4, the system can determine the current situation s_n is not used to generalize the cases in the casebase, and so the new case $c_n = [s_n, a_n, r_n]$ can be created with the current situational state, adapted action and the new benefits. Finally, the new case c_n will be added to the casebase as shown in Table 7. The newly updated

Table 7. Final casebase with the newly updated case

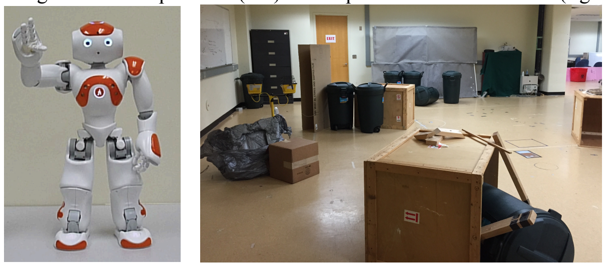
CaseID	State S						Action	Benefit
	f_{m1}	f_{m2}	f_{m3}	f_{m4}	f_{e1}	f_{e2}		
1	1	1	1	anger	1	1	a_t	-1
2	1	1	1	disgust	1	1	a_{d1}	+1
⋮				⋮			⋮	
9	0	0	0	happiness	1	0	a_{d2}	-3
10	0	0	0	sadness	0	1	a_t	-1
11	1	0	0	fear	1	1	a_t	+2

casebase is maintained and reused when the robot faces a new search and rescue situation in the future. Through these experiences, the robot can gradually increase the accuracy and effectiveness of its true/deceptive behaviors over time.

V. Conclusion

With the increasing use of autonomous robots in SAR, improving the interaction between human victims and rescue robots is critical. Similar to human cases, deception can be one efficient and essential capability for rescue robots in HRI. In this paper, we present an interesting and important research question in developing the use of robot deception in the SAR context; *Can deceptive capabilities of rescue robots benefit human victims?* We argued that a robot's deceptive capabilities can control a victim's emotional state and consequently establish a strong bond between victims and robots that lead to better cooperation. For this purpose, we introduced the computational model for a rescue robot's other-oriented deception inspired by criminological law. For the motive/opportunity parts, we proposed a deceptive action selection model using the CBR mechanism. Currently, we apply this model to a rescue robot system and are readying to conduct HRI studies (Figure 3) to evaluate the computational model and prove our research hypothesis.

Figure 3. Robot platform (left) and Experimental Environment (right)



References

[1] R. Murphy, *Disaster Robotics*, MIT press, 2014.

[2] F. Matsuno, N. Sato, K. Kon, H. Igarashi, T. Kimura, and R. Murphy, Utilization of robot systems in disaster sites of the great eastern japan earthquake, *Field & Service Robotics (FSR)*, 2014.

[3] J. Casper and R. Murphy, Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center, *Proc. IEEE Systems, Man and Cybernetics Conference*, 2003.

[4] H. A. Yanco, A. Norton, W. Ober, D. Shane, A. Skinner, and J. Vice, Analysis of Human-robot Interaction at the DARPA Robotics Challenge Trials, *Journal of Field Robotics* 32, no. 3 (2015): 420-444.

[5] J. Whalen and H. Zimmerman, Observations on the display and management of emotion in naturally occurring activities: The case of "hysteria" in calls to 911, *Social Psychology Quarterly*, 1998.

[6] J. Lois, Managing emotions, intimacy, and relationships in a volunteer search and rescue group, *Journal of Contemporary Ethnography*, 30, 131-179, 2001

[7] J. Shim and R. C. Arkin, A taxonomy of robot deception and its benefits in HRI, *Proc. IEEE Systems, Man and Cybernetics Conference*, 2013.

[8] B. M. DePaulo, D. A. Kashy, S. E. Kirkendol, M. M. Wyer, and J. A. Epstein, Lying in everyday life. *Journal of personality and social psychology*, 1996, 70(5): 979–995.

[9] E. Adar, D. S. Tan, and J. Teevan. Benevolent deception in human computer interaction. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1863-1872, 2013.

[10] J. Shim and R. C. Arkin, Other-Oriented robot Deception: A Computational Approach for Deceptive Action Generation to Benefit the Mark, *Proc. IEEE International Conference on Robotics and Biomimetics*, 2014.

[11] R. Murphy, Human-robot interaction in rescue robotics, *Proc. IEEE Systems, Man and Cybernetics Conference*, 2004

[12] J. Drury, L. Riek, A. Christiansen, Z. Eyler-Walker, A. Maggi, and D. Smith, Evaluating Human-Robot Interaction in a Search-and-Rescue Context, *In Proceedings of the Performance Metrics for Intelligent System (PerMIS) Workshop*, 2003.

[13] H. A. Yanco, M. Baker, R. Casey, B. Keyes, P. Thoren, J. L. Drury, D. Few, C. Nielsen, and D. Bruemmer, Analysis of human-robot interaction for urban search and rescue. *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics*, National Institute of Standards and Technology (NIST), 2006

[14] I. Nourbakhsh, K. Sycara, M. Koes, M. Young, and S. Burion, Human-robot teaming for search and rescue, *IEEE Pervasive Computing: Mobile and Ubiquitous Systems*, pp. 72-78, 2005.

[15] S. A. Morin, R. F. Shepherd, S. W. Kwok, A. A. Stokes, A. Nemiroski, and G. M. Whitesides, Camouflage and Display for Soft Machines, *Science* 337(6096):828–832, 2012.

[16] N. Carey, J. Ford, and J. Chahl, Biologically inspired guidance for motion camouflage, *in* 5th *Asian Control Conference*, 2004.

[17] J. Shim, and R. C. Arkin, Biologically-inspired deceptive behavior for a robot. *12th International Conference on Simulation of Adaptive Behavior*, 2012.

[18] J. Davis and R. Arkin, Mobbing behavior and deceit and its role in bio-inspired autonomous robotic agents, *International Conference on Swarm Intelligence*, pp. 276–283, 2012.

[19] M. Vazquez, A. May, A. Steinfeld, and W.-H. Chen, A deceptive robot referee in a multiplayer gaming environment, *in International Conference on Collaboration Technologies and Systems (CTS)*, 2011.

[20] E. Short, J. Hart, M. Vu, and B. Scassellati, Nofair!!: an interaction with a cheating robot, *in Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, 2010.

[21] S. Matsuzoe and F. Tanaka, How smartly should robots behave?: Comparative investigation on the learning ability of a care-receiving robot, *IEEE RO-MAN*, 2012, pp. 339–344.

[22] B. Brewer, R. Klatzky, and Y. Matsuoka, Visual-feedback distortion in a robotic rehabilitation environment, *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1739–1751, 2006.

[23] D. Chen and B. Burrell, Case-based reasoning system and artificial neural networks: A review, *Neural Computing & Applications*, vol. 10, no. 3, pp. 264-276, 2001.

[24] A. Aamodt and E. Plaza, Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *AI Communications*. 7(1): p. 39-59, 1994.

[25] M. Likhachev and R. C. Arkin, Spatio-temporal case-based reasoning for behavioral selection, *In ICRA*, pp. 1627–1634, 2001.

[26] L. Moshkina, Y. Endo, and R. C, Arkin, Usability evaluation of an automated mission repair mechanism for mobile robot mission specification. *In Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, 2006.

[27] L. Moshkina, S. Park, R. C. Arkin, J. K. Lee, and H. Jung, TAME: Time-Varying Affective Response for Humanoid Robots, *International Journal of Social Robotics*, August 2011, Volume 3, Issue 3, pp 207-221, 2011.

[28] J. L. Kolodner, An introduction to case-based reasoning, *Artificial Intelligence Review*, vol. 6, no.1, pp.3-34, 1992.

[29] E. J. McCluskey, *Minimization of Boolean function*, Bell system Tech. Journal, vol.35, No.5, pp. 1417-1444, 1956.

[30] A. Mishchenko, R. K. Brayton, and T. Sasao, *Exploring multi-valued minimization using binary methods*, 12th International Workshop on Logic and Synthesis, Laguna Beach, California, USA, 2003.

[31] http://www.remm.nlm.gov/

[32] *HUMAINE Emotion Annotation and Representation Language*, Emotion-research.net. Retrieved June 30, 2006.

[33] *The 10 rules of engagement for structural fire fighting and the acceptability of risk*, International Association of Fire Chiefs, 2001.