Adaptive Autonomy in Unmanned Ground Vehicles Using Trust Models

Armon Toubman[1,2], Peter-Paul van Maanen[1,2], Mark Hoogendoorn[2]

[1]*Department of Cognitive Systems Engineering, TNO Human Factors*
P.O. Box 23, 3769 ZG Soesterberg, The Netherlands
[2]*Department of Artificial Intelligence, Vrije Universiteit Amsterdam*
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
Email: armon@armontoubman.com, peter-paul.vanmaanen@tno.nl,
m.hoogendoorn@vu.nl

Although autonomous systems are becoming more and more capable of performing tasks as good as humans can, there is still a huge amount of (especially) complex tasks which can much better be performed by humans. When making such decisions however, it might show that in particular situations it is better to let a human perform the task, whereas in other situations an autonomous system might perform better. This could for instance depend upon the current state of the human, for instance measured by means of ambient devices, but also experiences obtained in the past. In this paper, a trust-based approach is developed which aims at judging the current situation and deciding upon the best allocation (human or autonomous system) to perform a certain task. Hereby, an experiment in the context of controlling a set of robots to dismantle bombs has been performed, with focus on multiple types of support. The results show that support by means of simply allocating the task to the most suitable party gives superior performance.

1. Introduction

Nowadays, more and more complex tasks that were originally performed by humans are being automated. This automation has become possible due to the huge advancements in technological development. For instance, in the field of robotics a trend can be seen that moves from robots that were completely controlled by a human to robots that handle complete tasks autonomously (cf. (Parasuraman *et al.*, 2000)). Although the choice for automation might sometimes be clear-cut (e.g. robots mounting windows on car are far more precise than humans) in other cases it might not be so obvious. For instance, it could be the case that an automated system performs far superiorly in straightforward situations, but in more complex situations a human might still perform better. Therefore, different so-called levels of autonomy (LOAs) can be defined ranging from complete control by the human to complete control by the autonomous system (see e.g. (Sheridan and Verplank, 1978)).

Currently, systems have been developed that try to accomplish an adaptive level of autonomy, see e.g. (Parasuraman and Wickens, 2008). Such a system can take the form of an advice system for a human operator, but can also be an autonomous system itself that selects one of the LOAs. A key consideration when selecting one of these LOAs is

the expected performance in a particular situation, both of the human and the automated system. In order to create these expectations, a performance model should be built up. Such a model should take previous experiences in similar situations into account, but can for instance also incorporate a model of the current state of the human operator (e.g. is the operator currently overloaded, or is the operator bored because he or she has nothing to do). In order to feed such a model with information, techniques from the domain of Ambient Intelligence can be deployed.

In this paper, a first step in this direction is made in the field of robot control. Hereby, a computational trust model (see e.g. (Sabater and Sierra, 2005)) is utilized to create a support system for a human operator. This system maintains a trust model for both the human operator and the automated system and derives which is better equipped to handle the current situation. These trust levels are based on a history of experiences. The setting that is investigated does not concern a single but multiple robots that need to be controlled simultaneously, thereby creating a situation where the human operator is simply not able to control all robots manually. In an experiment different forms of support are provided, ranging from displaying the trust level in both the autonomous system and the human operator, providing advice on who should take control of a particular robot, to completely autonomous assignment of control. Of course, hereby the fact that the human operator cannot control more than one robot at the same time is taken into account.

This paper is organized as follows. In section 2, the proposed support model is described. The hypotheses about the application of the support model are listed in section 3, and the method used to test these hypotheses is described in section 4. The results are given in section 5. The paper is concluded with a discussion in section 6.

2. Support Model

A support model is proposed that can be used to aid a supervisor with the supervision and control of multiple robots. The support model consists of two main parts: a set of trust models and an autonomy reasoner. With these parts, the support model is able to offer different types of support. The support model is used in the context of supervisor S monitoring $n > 1$ robots $R_1, \ldots, R_n$. For each robot R_n, the support model has two trust models: one trust model T_{R_n} to predict robot R_n's performance, and one trust model T_{S_n} to predict the supervisor's performance when they are controlling robot R_n. The autonomy reasoner uses the trust values from the trust models to decide which robots are allowed to function autonomously, and which robot's control should be shifted to the supervisor, if any. A graphical overview of the support model is given in figure 1.1(a).

2.1. *Trust models*

The support model uses multiple instantiations of the same trust model to calculate the trust it has in the supervisor and the robots (together: agents) regarding a certain task. The trust model calculates the trust $T_j(s,t)$ that the support model has in trustee $j \in \{S, R_1, \ldots, R_n\}$ at time step t, with $T_j(s,t)$ ranging between 0 and 1. The calculated trust $T_j(s,t)$ represents

a prediction of an agents' performance on the task.

With n robots, the support model uses $2n$ instantiations of the trust model. The performance of a robot R_n is predicted using a trust model T_{R_n}. Similarly, the performance of the supervisor when controlling each robot is predicted using trust models T_{S_n}.

The trust models use direct experiences and situations as input.

2.1.1. *Direct experiences*

At every time step t, each trust model receives input in the form of either a positive or a negative experience, or no experience at all. Experiences are modelled in the support model as entities without explicit value. To discriminate between positive and negative experiences, each trust model remembers positive and negative experiences by maintaining two sets (*Pos* and *Neg*). From these sets, two new sets (*PosR* and *NegR*) which contain only recent experiences are deduced at each time step:

$$\begin{aligned} &Pos = \{x | x \text{ is a time step at which a positive experience was received}\} \\ &Neg = \{x | x \text{ is a time step at which a negative experience was received}\} \\ &PosR = \{x \in Pos : x > t - \theta_t\} \\ &NegR = \{x \in Neg : x > t - \theta_t\} \end{aligned} \tag{1.1}$$

The parameter θ_t defines after how many time steps an experience is no longer counted as recent. To let experiences with an agent influence the trust in that agent, the value of recent experiences needs to be quantified. This is done by looking at the ratio of positive experiences to negative experiences at each time step:

$$\varepsilon_j(t) = \begin{cases} \frac{\#PosR}{\#PosR + \omega_{neg} \#NegR} & \text{if } \#PosR + \#NegR > 0 \\ 0.5 & \text{otherwise} \end{cases} \tag{1.2}$$

Here, ω_{neg} is a weight that can be used to balance the importance of negative experiences.

2.1.2. *Situation*

Combinations of features of the robots' environment, together with known properties of the robots, can form cues for the robots' expected performance. Such features make up situations. In the trust models, scores are assigned to situations so that situations with a positive outlook add trust, while situations with a negative outlook subtract trust. The specific situations and their scores are defined before any operation as possible combinations of features of j's environment. The defined situations are assigned scores that indicate the outlook offered by the different situations, ranging from negative (a score of 0) to positive (a score of 1). The trust model for j obtains the score of j's situation with a lookup function:

$$\sigma_j(s,t) = \begin{cases} \text{The score of } s \text{ at } t & \text{if } s \text{ is a defined situation} \\ 0.5 & \text{otherwise} \end{cases} \tag{1.3}$$

The t parameter is used to reflect the declining or increasing predictive value of a situation as time passes. For example, if a robot spends too much time in a situation with a positive outlook (compared to the expected task completion time), trust in the robot declines as the amount of time spent might indicate a failure.

2.1.3. *Combination of the input*

Direct experiences with j and j's situation at time step t have to be combined to form the new input I on which the trust in j at time step t is based:

$$I_j(s,t) = \omega_I \sigma_j(s,t) + (1-\omega_I)\varepsilon_j(t) \tag{1.4}$$

To be able to balance the importance of the experiences and the situation in the model, weight ω_I is introduced ($0 \leq \omega_I \leq 1$).

2.1.4. *Final definition*

The final definition of the trust model is as follows:

$$T_j(s,t) = \lambda_T T_j(t-1) + (1-\lambda_T) I_j(s,t) \tag{1.5}$$

A decay factor λ_T ($0 \leq \lambda_T \leq 1$) is added to control how strongly old trust influences new trust.

With $2n$ instantiations of this trust model, the support model is able to predict the performance of each robot, and the performance of the supervisor with each robot. These predictions, in the form of trust values, form the input of the autonomy reasoner, which is described in the next section.

2.2. *Autonomy reasoner*

The autonomy reasoner uses the trust values generated by the trust models to select (at most) one robot at each time step. The selected robot is the most suitable candidate for a shift in control, according to the support model. A number of criteria are applied on the trust values that enter the autonomy reasoner.

First, threshold θ_1 ($0 < \theta_1 < 1$) is defined as the trust value for each T_{R_i} above which robot R_i should not be considered further for a control shift. This can be formalized as the first criterion for the autonomy reasoner:

Criterion 1.1 (Good Enough). *For each robot R_i, if $T_{R_i} > \theta_1$, remove R_i from consideration for selection.*

While the predicted performance of a robot may be low ($\leq \theta_1$), the difference with the predicted performance of the supervisor with that robot might not be large enough to warrant a shift in control. A new threshold θ_2 ($0 < \theta_2 < \theta_1 < 1$) is needed for the maximum

difference in trust in the robot and the supervisor under which R_i is allowed to retain control, when $T_{R_i} \leq \theta_1$. The second criterion can now be defined:

Criterion 1.2 (Added Value). *For each robot R_i, if $T_{S_i} - T_{R_i} < \theta_2$, remove R_i from consideration for selection.*

From the robots that are still being considered, the autonomy reasoner selects the robot that is expected to perform the worst relative to the supervisor:

Criterion 1.3 (Lowest Relative Trust). *Find the distinct* $\arg\max_i(T_{S_i} - T_{R_i})$ *and select robot R_i.*

It is possible that multiple robots are tied for the lowest relative trust, meaning the Lowest Relative Trust Criterion can not select a single robot. In this case, the robot in which the support model has the lowest absolute trust should be selected:

Criterion 1.4 (Lowest Absolute Trust). *Find the distinct* $\arg\min_i(T_{R_i})$ *and select robot R_i.*

Again a tie is possible. In case of a tie with the Lowest Absolute Trust Criterion, a random robot is selected:

Criterion 1.5 (Multiple Ties). *Robot R_i is selected at random, and this selection is maintained until the next time step at which a robot is selected by application of any criterion except the Multiple Ties Criterion.*

If after application of the Added Value Criterion no robots are left for consideration for selection, a shift in control is not deemed necessary by the autonomy reasoner. However, if a shift is required, a selection can be forced by applying the Multiple Ties Criterion on all robots.

A graphical overview of the autonomy reasoner is given in figure 1.1(b).

2.3. *Support types*

Three types of support using the support model are proposed: Trust Overview, Weak Adaptive Autonomy and Strong Adaptive Autonomy.

As the first type of support (Trust Overview), the trust generated by T_{R_i} for each robot i can be presented to the supervisor. The autonomy reasoner is bypassed. The supervisor must make all decisions about the robots' autonomy themselves. The presentation of the trust values can be used as a decision aid.

As the second type of support (Weak Adaptive Autonomy), the support model is used as shown in figure 1.1(a). At each time step t, the trust in all agents is updated and a selection is made by the autonomy reasoner. This selection is presented to the supervisor as the most suitable candidate for a shift of control to the supervisor.

As the third type of support (Strong Adaptive Autonomy), the support model is again used as shown in figure 1.1(a). However, the selected robot is not merely presented to the supervisor. The supervisor is forced to take control of the selected robot. At each time step t, the trust in all agents is updated and a selection is made by the autonomy reasoner. The choice of whether to take control of a robot to the supervisor is taken from the supervisor. Instead, the control of the selected robot is forced on the supervisor by the support model.

3. Hypotheses

Several hypotheses can be formed on the effectiveness of the support types offered by the support model that was described in the previous chapter.

The support model was designed to help the supervisor monitor robots and decide over their autonomy. The support model, using trust models, should be able to more accurately predict the performance of robots than the supervisor. Therefore, the supervisor/robot team is expected to achieve higher performance in their tasks when the supervisor is supported with either Trust Overview (H1a), Weak Adaptive Autonomy (H1b), or Strong Adaptive Autonomy (H1c), compared to when the supervisor does not receive support. Specifically, team performance is expected to be the highest when the supervisor is supported by Strong Adaptive Autonomy (H2). In other words, the support model should be able to make better decisions on the robots' autonomy than the supervisor.

The support model uses trust models to predict the performance of robots. Since trust is a subjective measure, the only baseline suitable for comparison is that of human opinion. The support model's predictions and decisions can be compared to human opinion in two ways.

First, the trust the support model has in the robots and the supervisor can be compared to the trust a human would have, when given the same information as the support model to base their trust on. The expectation of increased team performance can be reduced into the expectation that the trust models used by the support model are better at predicting performance than the method used by humans. Therefore, trust generated by the support model is not expected to resemble trust reported by humans (H3), given the same information as input.

Second, the decisions of the support model on the autonomy of the robots (in the form of the robots selected by the switching mechanism) can be compared to the decisions of the supervisors in the same situations. For higher team performance with support from the support model, supervisors need to agree with the advice from the support model. Meanwhile, supervisors' agreement is expected to be low without support from the support model – if it were high, there would be no added benefit in application of the support model. Therefore, it is hypothesized that supervisors will agree with the advice from the support model (under Trust Overview and Weak Adaptive Autonomy)(H4). When no support is given, supervisors are not expected to agree with the selected robot the support model would have provided.

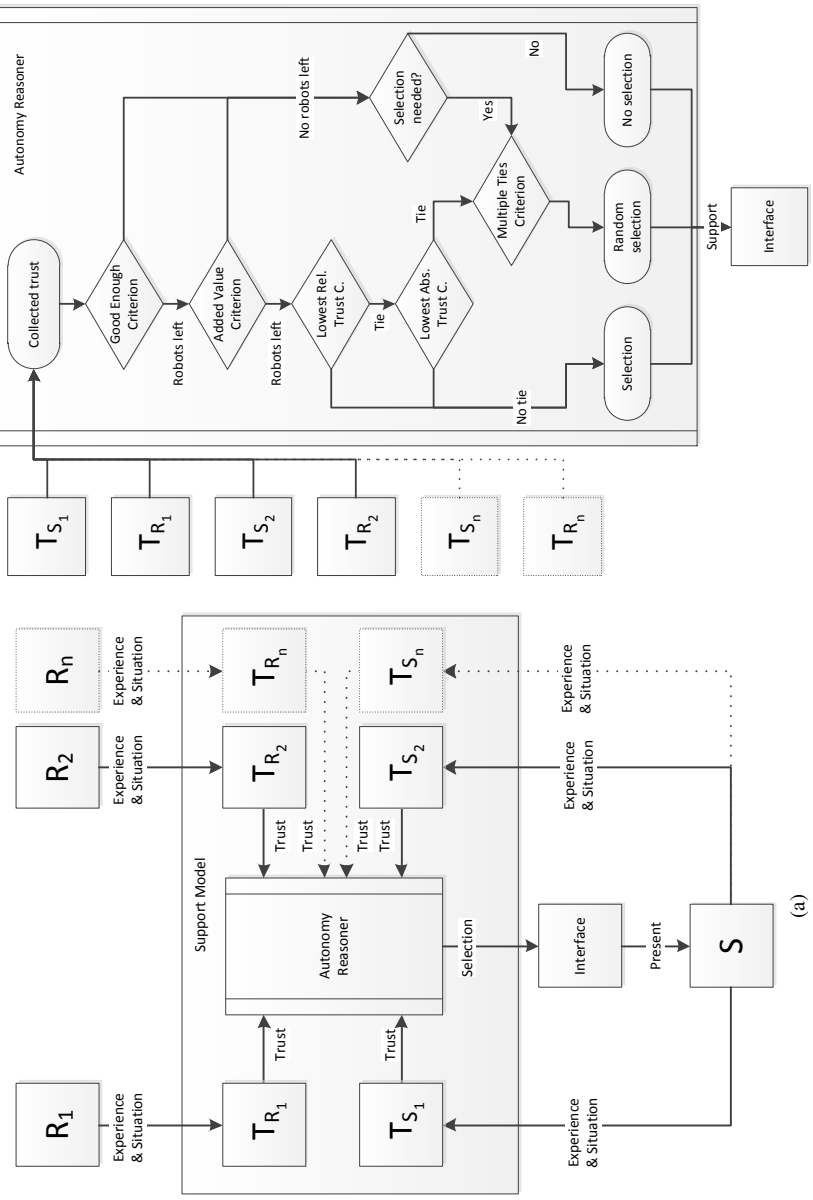


Fig. 1.1. (a) Overview of the support model. (b) Overview of the autonomy reasoner used in the support model.

4. Method

4.1. *Participants*

Thirty-five experienced computer users participated in the experiment ($M = 30.5$ years, $\sigma = 13.4$ years, 17 female). Out of each group of five participants, four were randomly assigned the role of supervisor.

4.2. *Task*

To test the hypotheses from section 3, a task has been designed in which four robots have to disarm as many bombs as possible at a priori unknown locations at the same time in separate virtual mazes. These four robots were supervised by a human supervisor. The robots themselves were operated by humans (as opposed to an AI algorithm, which 1) would have required more effort to program, 2) does not represent future AI capabilities anyway, and 3) the focus of the experiment was on the supervisor's task, not the robot operators' task). The robot operators were not able to communicate with the supervisor. The supervisor monitored the activity of the robots and was able to shift the control of one robot at a time to himself.

To add realism to the task (i.e., a certain degree of uncertainty of properly finding and disarming bombs) there were three types of bombs in each maze: 1) bombs that only the robot operators could see, 2) bombs that only the supervisor could see and 3) bombs that both could see. In this way both parties (robot operator and supervisor) would have their own capabilities and needed each other to take over control at different moments in time to have an optimal performance.

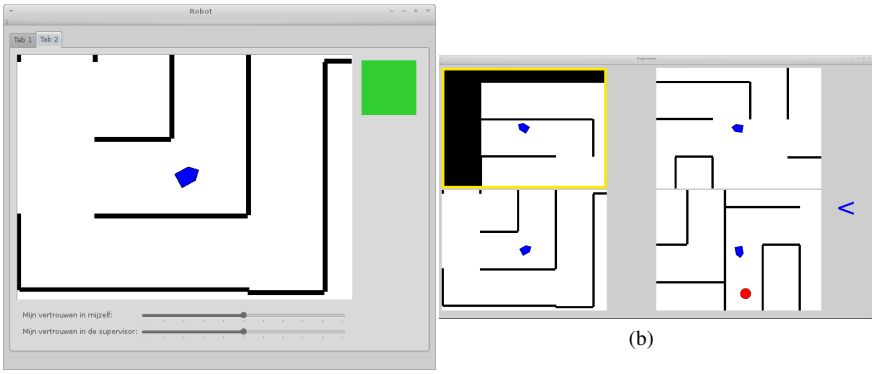
A bomb was disarmed (positive experience) when a robot drove over it when visible to its current controller. When a robot drove over a bomb that was not visible to its current controller, the bomb exploded (negative experience). Both the operator and supervisor were notified whenever one of these events happened. A bomb visible on the interface constituted a positive situation. If no bomb was picked up after 20 seconds of seeing one, the situation became neutral, and after 20 more seconds, the situation became negative until a bomb was picked up.

Control of a robot was required in order to see bombs. Also a shift of control to the supervisor required 5 seconds to process. These two measures were needed to prevent the supervisor from micro-managing the robots too easily (i.e., quickly take over control, disarm a bomb and release control whenever a bomb is seen by the supervisor).

Each maze contained the same number of bombs. However, the number of bombs of each type differed per robot in each trial. This caused each robot to have a different performance, giving the supervisor reason to pay close attention to the performance of each robot. The less bombs visible to a robot, the more difficult that robot's task is: it will find less bombs to disarm and have a higher chance of causing explosions. The same holds for the supervisor. Four combinations of bombs were used: one with most bombs visible to the robot operator, one setup with most bombs visible to the supervisor, one setup with

Table 1.1. Overview of the bomb combinations.

Combination	Visible by robot operator	Visible by supervisor	Visible by both
1	+	−	−
2	−	+	−
3	−	−	+
4	□	□	□



(a) (b)

Fig. 1.2. (a) Robot operator's interface. (b) Supervisor's interface.

most bombs visible to both, and one setup with an equal division. An overview of the bomb setups that were used can be found in table 1.1. The bomb combinations were balanced between the robots in the four rounds under each supervisor. The balancing scheme is shown in table 1.2.

The interface for the robot operators is shown in figure 1.2(a). It shows the robot's immediate surroundings. The operator was able to steer the robot through its maze using the WASD keys. Bombs were displayed as a red circle. When the robot drove over a visible bomb, a green indicator would appear for a short time, accompanied by a sound. When the robot drove over an invisible bomb, a red indicator would appear, accompanied by a different sound. When the robot was controlled by the supervisor, the same indicators and sounds were used for these events.

The interface contained a large colored indicator that showed who was in control of the robot: the robot operator or the supervisor. The indicator would switch between green (the operator controls the robot) and red (the supervisor controls the robot). Control switches were also accompanied by a clear sound.

The supervisor's interface let the supervisor monitor the activity of the robots and take control of one robot at a time. Support from the support model was also given through this interface. The different implementations of these types of support (as explained in section 2.3 is further described in section 4.3.1. The interface is shown in figure 1.2(b).

Table 1.2. Bomb combination latin square.

Trial number under same supervisor	Bomb combination for robot 1	2	3	4
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

Table 1.3. Experimental design.

Supervisor	Conditions			
1	NS	TO	WAA	SAA
2	TO	SAA	NS	WAA
3	WAA	NS	SAA	TO
4	SAA	WAA	TO	NS

4.3. *Design*

A 4×1 within subjects design was used. The single independent variable used in the design was the type of support offered by the support model. This resulted in four conditions, named after the support types: NS (No Support), TO (Trust Overview), WAA (Weak Adaptive Autonomy), and SAA (Strong Adaptive Autonomy).

The order in which the conditions were presented to the supervisors were balanced with a latin square, as shown in table 1.3.

4.3.1. *Independent variables*

The single independent variable that was manipulated in the experiment was the type of support offered by the support model as explained in section 2.3. Figure 1.2(b) shows the 'Weak Adaptive Autonomy' condition, which shows all components used in all conditions. Details specific to each support type are given below.

No Support Under this setting, the supervisor received no support from the support model. The supervisor had to measure each robot's performance manually by paying attention to the red and green indicators that accompanied disarmaments and explosions. The supervisor was able to take and give back control of robots at will.

Trust Overview Under this setting, the level of trust the trust models of the support model had in each robot was presented to the supervisor as a colored border around the window with each robot's activity. The color of each border represented the level of trust the support model had in that robot. The borders blended gradually between red (low trust), yellow (medium trust) and green (high trust). The supervisor was able to take and give back control of robots at will. The supervisor was instructed to consider the colored borders as performance predictors.

Weak Adaptive Autonomy Under this setting, the support model selected a robot which should be taken over by the supervisor, and presented the selection to the supervisor.

The indicator for the chosen robot was a colored border around the window with the chosen robot's activity. The color of the border represented the level of trust the support

model had in that robot. The border blended gradually between red (low trust), yellow (medium trust) and green (high trust). The supervisor was instructed to consider the robot that was selected by the support model. as the support model expected this robot to increase team performance the most with a control shift.

The autonomy reasoner in the support model was configured to always select a robot. This was done to keep supervisors involved in the operation. The trials were short, creating the possibility that the supervisor would not receive advice during a whole trial. Because the autonomy reasoner always had to select a robot, it was chosen to include the trust in the selected robot in the presentation of the selection (see section 2.3).

Strong Adaptive Autonomy Under this setting, the support model selected a robot which should be taken over by the supervisor. The supervisor was forced to comply with this decision. The selected robot was indicated with a blue arrow.

4.3.2. *Dependent variables*

Team performance Three measures were used to calculate the performance of the supervisor-robots team on the task.

The first measure was the number of bombs that was disarmed (hits). The number of disarmed bombs should be maximized. The higher the amount of bombs disarmed by the team, the higher the team's performance was.

The second measure was the number of bombs that was set off (misses). The number of bombs set off should be minimized. The lower the amount of bombs set off by the team, the higher the team's performance was.

The third measure combined the previous two measures. The z-scores of both the number of bombs disarmed and the number of bombs set off were calculated per condition. The z-score of the number of bombs set off was then subtracted from the number of bombs disarmed, resulting in a normalized measure of performance per condition. The third measure can be seen as a more objective measure than the first and second measures. Averaging the performance values of all trials per condition resulted in mean performance values for each condition.

Similarity of trust Trust was obtained in two forms: trust generated by the trust models and trust estimates from the robot operators:

Each trial, the trust models in the support model generated eight sets of trust values: trust in each of the four robot operators and four times trust in the supervisor.

Also during each trial, robot operator had to indicate their trust in themselves and in their supervisor using on-screen slider controls (as shown at the bottom of figure 1.2(a)). The slider controls had eleven-point scales. The participants were reminded every thirty seconds with spoken text to update the sliders if they felt their trust had changed. This resulted in eight pairs of trust datasets from each trial. The root mean square deviation (RMSD) was calculated for the pairs of these datasets with the same trustee (the robot

operator or the supervisor). This yielded two sets of RMSDs, one set for each trustee.

Agreement This shows how often the supervisor agreed with the support model by looking at which robots the supervisor took control of and what the support model had advised to take over (depending on the type of support).

5. Results

5.1. *Removal of outliers*

Out of the data from 28 supervisors, the data from 10 supervisors was removed because the number of disarmed or exploded bombs was higher or lower than two standard deviations from the mean performance.

5.2. *Team performance*

Team performance was calculated using three measures: the number of disarmed bombs (hits), the number of exploded bombs (misses), and a normalized score.

The first measure was the number of disarmed bombs. Details of the four conditions are given in table 1.4. The mean number of disarmed bombs per condition is shown in figure 1.3(a). A repeated measures ANOVA showed no significant effect of condition on performance, $F(3,51) = 2.5$, $p = 0.0695$. Paired sample t-tests indicated a significantly higher performance under SAA compared to NS. This was not the case with TO and WAA. The highest mean number of disarmed bombs was achieved under SAA.

The second measure was the number of exploded bombs. Details of the four conditions are given in table 1.5. The mean number of exploded bombs per condition is shown in figure 1.3(b). A repeated measures ANOVA showed a significant effect of condition on performance, $F(3,51) = 4.27$, $p = 0.0092$. Paired sample t-tests indicated no significantly higher performance under any of the conditions with support from the support model. The lowest mean number of exploded bombs was achieved under SAA.

The third measure was the normalized score. Details of the four conditions are given in table 1.6. The mean number of exploded bombs per condition is shown in figure 1.3(c). A repeated measures ANOVA showed a significant effect of condition on performance, $F(3,51) = 5.12$, $p = 0.0036$. Paired sample t-tests indicated a significantly higher performance under SAA compared to NS. This was not the case with TO and WAA. The highest mean normalized score was achieved under SAA.

The number of hits under SAA was significantly higher than the number of hits under No Support (NS), while the application of Trust Overview (TO) and Weak Adaptive Autonomy (WAA) did not significantly improve performance. Based on the results, hypotheses H1a and H1b are rejected, while hypothesis H1c is accepted. Furthermore, using each of the three performance measures, the highest performance was achieved under Strong Adaptive Autonomy (SAA). For this reason, hypothesis H2 is accepted.

Table 1.4. Statistical reports on team performance as the number of disarmed bombs per condition.

			$\mu > \mu_{NS}$		
Condition	μ	σ	t	df	Sig.
NS	56.6111	7.9049	-	-	-
TO	55.8333	8.4523	-0.5416	17	0.7024
WAA	57.2778	6.4790	0.3816	17	0.3537
SAA	60.6667	9.2036	2.2886	17	0.0176

Table 1.5. Statistical reports on team performance as the number of exploded bombs per condition.

			$\mu < \mu_{NS}$		
Condition	μ	σ	t	df	Sig.
NS	9.4444	2.7912	-	-	-
TO	10.5000	2.9754	1.1703	17	0.8710
WAA	12.2778	2.9267	2.5543	17	0.9897
SAA	9.2222	2.6022	-0.2827	17	0.3904

Table 1.6. Statistical reports on team performance as normalized scores per condition.

			$\mu > \mu_{NS}$		
Condition	μ	σ	t	df	Sig.
NS	0.1818	1.2929	-	-	-
TO	-0.2633	1.4930	-1.5267	17	0.9274
WAA	-0.6735	1.3227	-2.0131	17	0.9699
SAA	0.7551	1.5399	2.0781	17	0.0266

The increased performance under SAA shows that the support model can make better decisions on autonomy than human supervisors. This is supported by the performance under TO and WAA. The high performance under SAA shows that the advice given by the support model under TO and WAA was useful for basing decisions on.

5.3. *Similarity of trust*

As described in section 4.3.2, two times eight sets of trust values were collected each trial.

Two one-sample t-tests were conducted, comparing each set of RMSDs to a mean of 0. There was a significant difference for both the trust in the robot ($p = 0$, $a = 0.05$) and the trust in the supervisor ($p = 0$, $a = 0.05$).

Neither the trust of the supervisor and the trust models in the supervisor nor the trust in the robot operators correlated between the trustors. Therefore, hypothesis H3 is accepted.

5.4. *Agreement*

Each trial yielded two sets of data on the control of the robots: one set showing which robots the support model had selected, and one set showing which robots the supervisor actually took control of.

The agreement between the supervisor and the support model was calculated as Cohen's kappa for each trial except trials under Strong Adaptive Autonomy. This resulted in mean kappas for No Support ($\mu_\kappa = 0.0622$, $\sigma_\kappa = 0.1371$), Trust Overview ($\mu_\kappa = 0.1055$, $\sigma_\kappa = 0.1126$), and Weak Adaptive Autonomy ($\mu_\kappa = 0.1216$, $\sigma_\kappa = 0.1424$).

The agreement of the supervisor with the support model under NS, TO and WAA was low by any standard. Therefore, hypothesis H4 is rejected. Apparently, the method used by the supervisors to make decisions on the robots' autonomy was very different from the method used by the support model. A high agreement of the supervisors with the support model would show that the support model used a method comparable to the method used by the supervisors. However, this is clearly not the case.

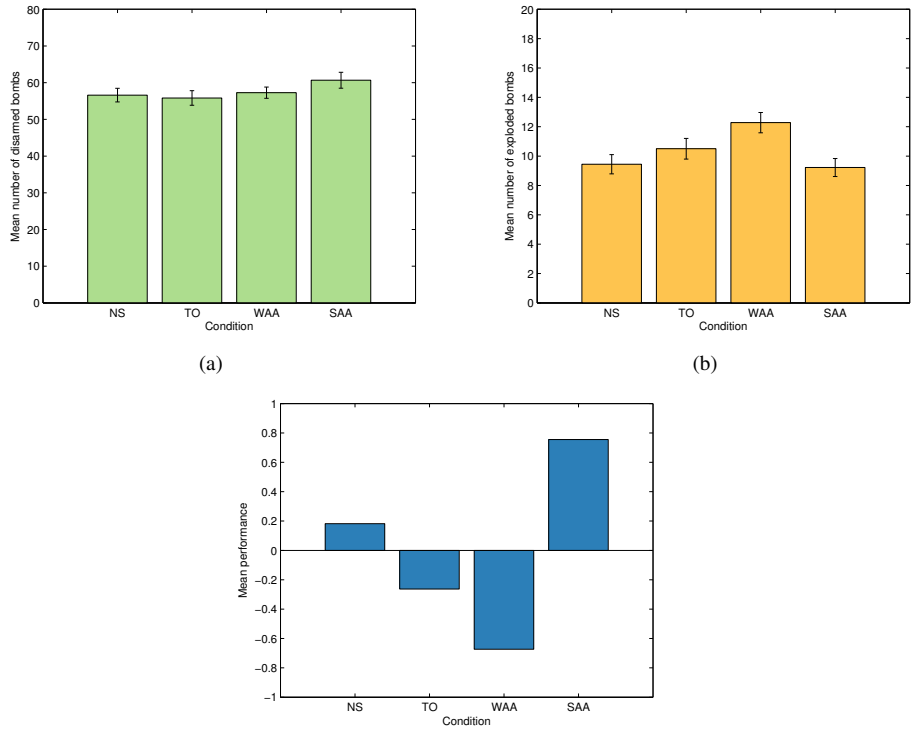


Fig. 1.3. (a) Mean number of disarmed bombs per condition. (b) Mean number of exploded bombs per condition. (c) Mean normalized team scores per condition.

6. Discussion

The main goal of this study was to find out if support from a support model that uses trust models would increase team performance in the case of a human supervisor monitoring multiple robots. Trust models have previously been used to provide support in different settings, but results have varied (Van Maanen *et al.*, 2007, 2011). In this study, a clear positive effect was found.

The most interesting result is that team performance was the highest under SAA. Apparently, the support model made better decisions on the LOAs of the robots than the human supervisors did. This result is coherent with the low agreement that was found between the supervisors and the support model, and also with the low correlation in trust.

One explanation for the higher performance under SAA is computational power. While the support model can monitor the robots and the supervisor without error, human supervisors have a limited attention capacity, meaning they may have missed or wrongly attributed events such as disarmaments of bombs, explosions, and changed situations. The wrong attribution of events could be explained with Weiner's attribution theory (Weiner, 1985). Supervisors may have attributed their own successes to skill, while attributing the robots' successes to luck. The support model made objective observations of the robots and the supervisors, leading to better decisions.

The low agreement of the supervisor with the support model's advice under the TO and WAA conditions can be regarded as under-reliance. Under-reliance remains a problem in the field of human-machine interaction (Parasuraman and Wickens, 2008). Several factors could have contributed to the supervisors' under-reliance on the support model. Among these factors are an inadequate understanding of the method used by the support model to select robots, and the inability to access the raw information which the support model uses as input.

It should be noted that the configuration of the support model, which forced it to continuously select robots to prevent underload, may have influenced the supervisors' reliance on the advice under WAA. The selection of robots which were not in actual need of a control shift based on their predicted performance, may have been perceived by the supervisors as false alarms. In short, agreement under WAA may have been higher if the continuous selection of robots was disabled.

Some comments can be made on the inner workings of the support model. One such comment is about the used paradigm that positive experiences increase trust, and negative experiences decrease trust. Falcone and Castelfranchi (2004) call this view naive and not useful for artificial systems, because it is unable to attribute successes and failures to their proper causes. They note, however, that such a view cannot be avoided if the trust is modelled as a simple number. In the trust models used in the support model, trust was indeed modelled as a simple number. By using multiple trust models, the support model did avoid the naive view on attribution.

Another comment that can be made is that the values of the support model's parameters used in the implementation were chosen manually. Tuning the parameters may lead to even

higher performance using the support model. Parameter tuning could be done offline before operation, but future research may also bring effective online parameter tuning techniques. This way, the support model and the support it provides could be made more adaptive to unknown and changing environments. The autonomy reasoner could also be made adaptive, for example with the exploration and exploitation method described in (Hoogendoorn *et al.*, 2010).

The proposed support model showed promising results, especially when it was allowed to make all LOA decisions. The support in the form of advice can be improved. Future research should point out if the support model can be used effectively in different settings.

References

Falcone, R. and Castelfranchi, C. (2004). Trust Dynamics: How Trust Is Influenced by Direct Experiences and by Trust Itself, in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '04 (IEEE Computer Society, Washington, DC, USA), ISBN 1-58113-864-4, pp. 740–747, doi:10.1109/AAMAS.2004.286, URL `http://dx.doi.org/10.1109/AAMAS.2004.286`.

Hoogendoorn, M., Jaffry, S. W. and Treur, J. (2010). Exploration and Exploitation in Adaptive Trust-Based Decision Making in Dynamic Environments, *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on* **2**, pp. 256–260, doi:10.1109/WI-IAT.2010.199, URL `http://dx.doi.org/10.1109/WI-IAT.2010.199`.

van Maanen, P.-P., Klos, T. and van Dongen, K. (2007). Aiding Human Reliance Decision Making Using Computational Models of Trust, , pp. 372–376doi:10.1109/WI-IATW.2007.108, URL `http://dx.doi.org/10.1109/WI-IATW.2007.108`.

van Maanen, P.-P., Wisse, F., van Diggelen, J. and Beun, R.-J. (2011). Effects of Reliance Support on Team Performance by Advising and Adaptive Autonomy, in *Proceedings of the 2011 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-2011)* (IEEE Computer Society Press).

Parasuraman, R., Sheridan, T. B. and Wickens, C. D. (2000). A Model for Types and Levels of Human Interaction with Automation, *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* **30**, 3, doi:10.1109/3468.844354, URL `http://dx.doi.org/10.1109/3468.844354`.

Parasuraman, R. and Wickens, C. D. (2008). Humans: Still Vital After All These Years of Automation, *Human Factors: The Journal of the Human Factors and Ergonomics Society* **50**, 3, pp. 511–520, doi:10.1518/001872008X312198, URL `http://dx.doi.org/10.1518/001872008X312198`.

Sabater, J. and Sierra, C. (2005). Review on Computational Trust and Reputation Models, *Artif. Intell. Rev.* **24**, pp. 33–60, URL `http://portal.acm.org/citation.cfm?id=1057866`.

Sheridan, T. B. and Verplank, W. L. (1978). Human and computer control of undersea teleoperators (Man-Machine Systems Laboratory Report), .

Weiner, B. (1985). An Attributional Theory of Achievement Motivation and Emotion, *Psychological Review* **92**, 4, pp. 548–573, URL `http://view.ncbi.nlm.nih.gov/pubmed/3903815`.