Cognitive Modeling of Automation Adaptation in a Time Critical Task

Junya Morita 1*, Kazuhisa Miwa 2, Akihiro Maehigashi 3, Hitoshi Terai 4, Kazuaki Kojima 5 and Frank E. Ritter 6

1 Department of Behavior Informatics, Faculty of Informatics, Shizuoka University, Hamamatsu, Japan, 2 Department of Cognitive and Psychological Sciences, Graduate School of Informatics, Nagoya University, Nagoya, Japan, 3 Center for Research and Development in Admissions, Shizuoka University, Shizuoka, Japan, 4 Department of Information and Computer Sciences, Faculty of Humanity-Oriented Science and Engineering, Kinki University, Fukuoka, Japan, 5 Learning Technology Laboratory, Teikyo University, Tochigi, Japan, 6 College of Information Sciences and Technology, Pennsylvania State University, University Park, PA, United States

This paper presents a cognitive model that simulates an adaptation process to automation in a time-critical task. The paper uses a simple tracking task (which represents vehicle operation) to reveal how the reliance on automation changes as the success probabilities of the automatic and manual mode vary. The model was developed by using a cognitive architecture, ACT-R (Adaptive Control of Thought-Rational). We also introduce two methods of reinforcement learning: the summation of rewards over time and a gating mechanism. The model performs this task through productions that manage perception and motor control. The utility values of these productions are updated based on rewards in every perception-action cycle. A run of this model simulated the overall trends of the behavioral data such as the performance (tracking accuracy), the auto use ratio, and the number of switches between the two modes, suggesting some validity of the assumptions made in our model. This work shows how combining different paradigms of cognitive modeling can lead to practical representations and solutions to automation and trust in automation.

Keywords: automated operation, reinforcement learning, ACT-R, Semi-Markov Decision Process, trust calibration

1. INTRODUCTION

Automation technology, which can partially substitute for human cognitive functions, has made remarkably progress recently. Although the application area of such technology is diverse, one of the recent prominent areas is the automatic operation of vehicles. The operation of ships and aircraft has commonly been automated in our society. For cars, automation of some functions such as speed control (i.e., adaptive cruise control) and braking (anti-lock) have also been used for a long time. In recent years, automatic control of steering has been actively developed due to the rapid progress of sensing and machine learning technologies. However, there are still barriers to the full application of automatic driving (self-driving cars). For a while, it has been assumed that automatic control will be used with driver's monitoring to intervene immediately at any time if the automatic control fails to respond properly (National Highway Traffic Safety Administration, 2016).

When new technologies, not limited to automatic control of vehicles, are introduced, misuse (overreliance) and disuse (underutilization) of the technologies has often become a problem. For sustainable industrial development in our society (United Nations Industrial Development Organization, 2015; Fukuyama, 2018), it is important to understand how humans adapt to new
technologies. Disuse of new technology results in less innovation, while misuse of new technology can cause serious accidents. In the field of human factors, such problems have been repeatedly discussed (e.g., Bainbridge, 1983; Parasuraman and Riley, 1997).

However, previous studies in the field have not fully considered time factors involved in the adaptation process to new technologies. Contrary to some other tasks studied in human factors, vehicle operation is a dynamic continuous process in which the cycle of perception, judgment, and action sequentially repeats. Automated vehicle systems partially substitute for such human operation. In a case where an operator can use automatic operation, s/he repeats the cycle of perception and judgment while observing that an automation system executes the overall cycle. When the operator notices that the auto control has problems, s/he needs to immediately turn off the automation to return to manual control.

We suggest that the above adaptation mechanism to automation technology can be partially explained by reinforcement learning, which updates selection probabilities of actions with rewards from the environment (Sutton and Barto, 1998). In a broader context, this paradigm has already been used to model the interaction between human and automation systems. As will be described in section 2, Gao and Lee (2006) proposed a computational model called Extended Decision Field Theory that simulates how operators adapt to a system that automates plant operation. In their model, a selection probability of using the automation system is dynamically changed through iterated environmental feedback.

However, it is not simple to apply the paradigm of reinforcement learning to a task that involves time-critical decision making. Generally, reinforcement learning has been applied to discrete tasks through Markov Decision Processes (MDP)—like bandit tasks (Sutton and Barto, 1998). Contrary to typically applied fields of reinforcement learning, vehicle operation does not directly fit into a MDP representation. Rather, it can be expressed as a SMDP (Semi-Markov Decision Process). SMDP introduces the concept of a time delay between action selection and state transition, and rewards that can be delivered at different points in time (Duff and Bradtke, 1996; Asada et al., 1999; Sutton et al., 1999; Georgeon and Ritter, 2012; Rasmussen and Eliasmith, 2013).

In this paper, we present a simple task that has some characteristics of continuous vehicle operation with automation and construct a model to reveal what type of mechanisms can simulate human adaptation to automation in a time-critical task like automatic vehicle operation. We specifically try to answer this question integrating a traditional reinforcement learning algorithm with a cognitive architecture. Using a cognitive architecture, we will explore this question using appropriate time constraints on behavior.

In the next section, we review research associated with the present study, showing a framework to simulate human behavior in real-time, and research on adaptation to automated systems. After that, we present the task, initial human data, the model, and the simulation results, describing a mechanism that can simulate a human adaptation process to automatic vehicle operation and show how it predicts and explains human performance on the task. Finally, we summarize the results obtained in this study and their implications.

2. RELATED STUDIES

This section presents previous work relating to the method and the tasks of the previous studies that define the theories and models that we use.

2.1. Research About Cognitive Architecture

Many reinforcement learning studies have been conducted so far. Currently, this approach is being used as a framework for end-to-end simulation in tasks such as games and maze search (Silver et al., 2016; Banino et al., 2018). However, these simulations are still unclear in terms of their correspondence with human behavior.

In contrast, cognitive architectures can be regarded as a framework to simulate human behavior with detailed time predictions. Usually, cognitive architectures include modules responsible for perception and action to model the overall behavior in a task. Among several cognitive architectures that have been proposed so far, the present study focuses on ACT-R (Adaptive Control of Thought-Rational; Anderson, 2007; Ritter et al., 2019 for a recent review) because this architecture has a large community and the mechanisms are well-tested. The present study aims to extend such mechanisms to the time-critical domain. Other architectures such as Soar ( Laird, 2012) and EPIC (Kieras and Meyer, 1997) could probably be used to simulate this phenomenon as well.

In the field of human factors, it has been discussed that automation systems cannot replace human cognitive tasks completely. Bainbridge (1983) claimed that even highly automated systems need human operators to monitor system performance and to make decisions of system use. Some researchers have also pointed out that human decision-making of system use is not optimal. Parasuraman and Riley (1997) noted that there are two types of maladaptive uses of automation: misuse (the overreliance of automation), and disuse (the underutilization of automation). Some studies indicated that human users have a bias to use automation (misuse, Singh et al., 1997; Skita et al., 2000; Bahner et al., 2008). On the other hand, other research has indicated that human users have a bias toward
The present task simulates the behavior of participants in a behavioral experiment conducted by Maeghigashi et al. (2013). The task used in the study was a simple tracking task, called the “line-following task.” Figure 2 shows the task environment. In this task, the operators are required to control the horizontal position of the vehicle (red circle) to follow the black line that scrolls down at 24 pixels per second. The screen is updated every 40 ms. If the vehicle does not overlap the line, a warning is presented outside of the window. The line is drawn by randomly combining 48 pixels high line patterns of varied angles (30, 45, 90, 135, and 150 degrees, equally chosen).

The vehicle is controlled by commands of “left,” “straight,” or “right.” If the vehicle receives a left command, the vehicle moves 1 pixel left from the original position. The command is sampled at 48 Hz. If a key-press event is detected, a flag of sending commands is set to on. This flag is off when a key-release event occurs. The vehicle moves 1 pixel per 40 ms. If the vehicle does not overlap the line, a warning is presented. Failures occur at specified rates; the vehicle position. In the experiment, the optimal line and goals are not visible to participants.

In both control modes, commands are not always successfully sent to the vehicle. Failures occur at specified rates; Ca and Cm are not visible to participants.

In both control modes, commands are not always successfully sent to the vehicle. Failures occur at specified rates; Ca and Cm are not visible to participants.

1Therefore, the command rate is not influenced by a key-repeat rate setting in an operating system or keyboard.

3. THE TASK
specify these rates. If $Ca$ or $Cm$ is low, the vehicle controlled by the corresponding mode is lagged, and it becomes hard to follow the line. To conduct the task successfully, operators need to select a suitable mode in each situation. The operators freely change between modes by pressing the space-bar. The experiment conducted by Maehigashi et al. (2013) used the task in 25 conditions where $Ca$ and $Cm$ levels were manipulated (5 levels of $Ca$ ranging from 30 to 70% v.s. 5 levels of $Cm$ ranging from 30 to 70%). In each condition, the participants ($n = 63$) performed the task for 40 s.

The results of this experiment are summarized as Figure 3. The graph on the left shows the performance of the task, which is the ratio of time that the vehicle was able to follow the line during the task. The middle graph indicates the auto use ratio, which represents how long the auto mode was used during the task. The right graph presents the number of switches, which represents how many switches occurred between the two modes during the task. All graphs have the x-axis and y-axis corresponding to $Ca$ and $Cm$ levels, respectively, and it can be observed that the values of the z-axis rise from the front to the back. However, the directions of the x and y axes are different in each graph to make the surfaces more visible. For the performance graph, the lowest level of $Ca$ and $Cm$ is placed in the front. As for the auto use ratio, the highest level of $Cm$ and the lowest level of $Ca$ are arranged in the front. Concerning the number of switches, both the highest $Cm$ and $Ca$ levels are placed at the front of the graph.

That is, we can observe that the performance got higher as both $Ca$ and $Cm$ levels increased. The auto use ratio increased as the level of $Cm$ decreased and the level of $Ca$ increased. The number of switches increased as the values of $Ca$ and $Cm$ were lower. The present study tries to construct a cognitive model that reproduces these trends under the same conditions as the human experiment. Although this task is very simple compared to driving in the real-world, the cycle of perception and actions in real-time and the choice to trust automation (or not) is reproduced. We suggest that the task is appropriate to explore adaptation mechanisms to automation in time-critical fields.

4. MODEL

In this section, we will first present the architecture and then the detailed mechanisms of the model will be explained.

4.1. Architecture

This study uses ACT-R to construct a model for the line-following task. This architecture integrates several cognitive modules including a visual module, a motor module, and a production module. The visual module is used to take information from an external environment. The motor module manipulates devices like a keyboard or a mouse in an external environment. These modules have buffers to temporarily hold declarative information called a chunk. The production module integrates the other modules by production rules, which consists of a condition/action pair that is used in sequence with other productions to perform a task. Conditions and actions in production rules are specified as patterns in the buffer contents of modules. Further reviews are available (Anderson et al., 2004; Anderson, 2007; Ritter et al., 2019 also see act.psy.cmu.edu).

Importantly, each event executed by ACT-R’s modules has a parameter of performance time. For example, ACT-R production rules take 50 ms to apply. Events including visual perception and motor actions, such as eye-movements, mouse-movements and key-presses, also have time parameters. These parameters have been developed and validated by psychological studies (Anderson et al., 2004)\(^2\). By using these parameters, ACT-R makes real-time simulations possible.

ACT-R also includes sub-symbolic cognitive processes that modulate the probabilities of firing (applying) production rules. When several rules match to the buffer conditions, a rule must be chosen. This is called conflict resolution. The choice is made based on comparisons of the utility values associated with each matching production rule. More specifically, the probability of firing production $i$ is described by Equation (1).

$$P(i) = \frac{e^{U_i}/s}{\sum_j e^{U_j}/\sqrt{s}}$$

(1)

$s$ is a parameter that determines the variance of the noise distribution according to a logistic distribution; $U_i$ is the utility of the production $i$ that competes with the number of productions with utility values $U_j$. The learning of $U_i$ is controlled by Equation (2).

$$U_i(n) = U_i(n-1) + \alpha[R_i(n) - U_i(n-1)]$$

(2)

$\alpha$ is the learning rate; $R_i(n)$ is the reward value given to production $i$ at time $n$. The learning occurs when a reward is triggered, and all productions that have fired since the last reward are updated. Though the official theory of ACT-R (Anderson, 2007) does not explicitly note it, this learning is the same as the basic reinforcement learning method called Q-learning, which updates the quality of a state-action combination by receiving rewards from the environment (Sutton and Barto, 1998). The relations between the two theories were also discussed by Fu and Anderson (2006).

We considered that the above characteristics (the visual and motor modules to interact with external environments, the real-time simulation, the utility update based on reinforcement learning) are useful for modeling an adaptation process on automatic vehicle operation.

4.2. Simulated Task Environment

By using the ACT-R Graphical (user) Interface (AGI) tool that is part of ACT-R 6 (Anderson, 2007; as a technical\(^3\)This approach was originally developed by Card et al. (1983) based on the finding by Sternberg (1969). The perceptual-motor system of ACT-R was also developed based on EPIC architecture (Kieras and Meyer, 1997).
FIGURE 3 | The results of the human experiment. Error bars represent σ/2 (plots are rotated to show data more clearly).

reference see Bothell, 2007⁴, we developed a simulated task environment with which the constructed model interacts. The simulated environment is the same as the original environment in the keyboard layout, screen update rates, line scrolling speed, vehicle size, line width, and screen size. The auto control mode is implemented with Common Lisp in the simulated task environment. However, unlike the original environment, visible goal positions are set at each corner to allow the model to perceive the path.

4.3. Basic Cycle of the Model

Figure 4 describes the basic cycle of the model. The model uses the production, goal, vision, and motor modules of ACT-R 6, and 11 production rules. These rules consist of a perceptual (the top part of the figure) and motor process (the bottom part of the figure) similar to previous driving models in ACT-R (Salvucci, 2006; Ritter et al., 2007; Salvucci and Taatgen, 2010).

In the perceptual process, the model picks visual information from a visual location buffer that holds location information of objects in the environment. The FindVehicle rule finds the horizontal position of the vehicle and places it into the goal buffer. The FindGoal rule finds the horizontal position of the nearest goal position and places it into the goal buffer. The position information in the goal buffer is used in the subsequent motor process. After the motor process, the information in the goal buffer is cleared to begin the next cycle.

The motor process depends on the current mode. In each mode, there is a rule to switch the current mode (ToAuto/ToManual). These mode-switching rules send a command to release currently pressed keys to the motor module. After finishing the key-release, the PressSpace rule sends a motor command pressing the space-bar, toggling the mode.

To fire the mode-switching rules, the model needs to solve conflict resolution (Equations 1 and 2) with other rules in each situation. In the auto mode, the ToManual rule conflicts with the KeepA rule that just clears the goal buffer. In the manual mode, the ToAuto rule competes with the KeepM, ToLeft, ToRight, LtoS, and RtoS rules. These five rules have different conditions specifying the vehicle and the goal positions, and current move-commands (left, right, straight). The action clauses of the ToLeft, ToRight, LtoS, RtoS rules send a command to hold or release a key to the motor module⁵. The KeepM rule does not have any action clauses relating to the motor module. This rule just clears the goal buffer.

Figure 5 presents a time flow diagram showing the relations between the environmental changes and the model cycles. Each column of the diagram represents events of the environment and the modules of the model. The environment regularly updates the screen every 40 ms. Individual rule firings take 50 ms, but the cycle of the model is not regulated. There are delays due to the interaction with the environment. The processing of the visual location module itself has no delay. The model recognizes the vehicle or the goal position on the screen at the same time as the corresponding rule fires. However, the ACT-R motor module needs preparation and execution time, which depends on the status of the motor module. These delays disadvantage manual control by the ACT-R model compared to the automatic control in the task simulation.

4.4. Learning and Mode Switching

We now explain how the model adapts to the two modes. First, we describe the default mechanism of ACT-R and then our modification of the mechanism.

4.4.1. Default Learning of ACT-R

In studies that use ACT-R reinforcement learning (Lovett and Anderson, 1996; Anderson et al., 2004; Anderson, 2007), a reward parameter is assigned to the specific rules by a modeler. When a rule with a reward value fires, all rules that have fired since the

⁴The latest version of ACT-R is 7.14. We used version 6 because we first implemented the model in that version. The latest version has a richer interface but the basic structures and functions have not largely changed.

⁵The default implementation of ACT-R 6 does not include key-press and key-release functions (but see the extras folder for a new draft implementation of this capability). We used a customized module in which the time parameter of key-punch is used for these actions.
FIGURE 4 | The basic cycle of the model.

FIGURE 5 | Time flow diagram of the ACT-R model components as they perform the task.
last utility update receive the reward. Following this paradigm, this study assigns a reward parameter to the FindVehicle rule (see Figure 4). In every cycle of the model, this rule perceives the vehicle location. Thus, it is reasonable to assume that this rule triggers rewards by checking a relative position between the vehicle and the line to follow.

More specifically, the default framework of ACT-R divides the FindVehicle rules into two variations: the FindVehicleOnline rule that fires when the vehicle is on the line, and the FindVehicleOffline rule that fires when the vehicle is off the line. Then, it sets the reward value triggered by the former to positive, and the reward value triggered by the latter to zero. As a result, when the vehicle is on the line, the utilities of the rules included in the current mode rises, and when the vehicle is off the line, those utilities go down. As the FindVehicleOffline rule keeps firing, the utility of the rules for mode switching (ToManual in the auto mode/ToAuto in the manual mode) becomes higher than the competing rules (KeepA in the auto mode/KeepM, ToLeft, ToRight, LtoS, RtoS in the manual mode), and causes the model to switch to another mode.

However, there are the following two difficulties for simply applying this paradigm to our model:

1. The problem of rewarding.
   As shown in Figure 5, there are motor delays in the ACT-R architecture. Reflecting these delays, the rules in the manual mode fire slightly less often than the automatic mode rules over similar time periods when they are preferred. They thus receive less opportunity for updating rewards.

2. Absence of direct comparisons.
   During the cycle in Figure 4, the utilities of the two modes are not directly compared. The rules keeping the current mode (KeepA/KeepM) receive rewards corresponding to how well the vehicle moves in the current mode. On the other hand, rewards for the mode-switching rules are influenced by the vehicle movement in both of the two modes because the perception-action cycle of mode-switching bridges across the two modes (see Figure 5). Accordingly, mode switching is made only based on the utility for the current mode without considering how well the vehicle moved in the other mode. Adaptation to the proper mode may be possible even if this problem exists. However, it is reasonably assumed that humans can predict future rewards by remembering past experiences.

4.4.2. A Solution to the Problem of Rewarding

The present study uses the reinforcement learning method from SMDP to solve the problem of rewarding. Unlike the MDP situation where rewards are regularly given, in the SMDP situation, it is necessary to appropriately distribute the rewards given irregularly to the rules. According to Sutton and Barto (1998), the expectation value of the reward received at time \( t \) under SMDP is defined by Equation (3).

\[
R(t) = \sum_{k=0}^{T} \gamma^k r_{t+k+1}
\]

\( T \) is the elapsed time since the last reward; \( \gamma \) is the time discount rate; \( r \) is the immediate reward received \( k \) units time ago. Because \( T \) is not fixed in the SMDP theory, the frequency of receiving rewards is possibly different between rules. To solve this, the above equation sums immediate rewards to balance the amount of rewards received over the long term. Following this definition, the present study assumes the travel distance of the vehicle per unit time (speed) as an immediate reward, and the model received the summation of the immediate reward across the interval of reward acquisition.

Specifically, distinguishing the FindVehicleOnlineSMDP rule that fires when the vehicle is on the line from the FindVehicleOfflineSMDP rule that fires when the vehicle is off the line, the reward is calculated as follows.

- The reward calculation of FindVehicleOnlineSMDP
  \[
  R_t = \text{MaxVehicleSpeed} \times \text{DurationFromReward}
  \]

- The reward calculation of FindVehicleOfflineSMDP
  \[
  R_t = \text{abs}(vposi_t - vposi_{t-1})
  \]

In Equation (4), the maximum vehicle speed (MaxVehicleSpeed) is set to 48 pixels/s. The duration from the last reward (DurationFromReward) is estimated by using the temporal module of ACT-R, which is used to simulate the estimation of subjective time (Taatgen et al., 2004). In Equation (5), the vehicle position at the last reward (vposi_{t-1}) is stored in the goal buffer and compared to the current vehicle position (vposi_t).

Thus, these equations represent approximations of the travel distance of the vehicle from the previous reward to the current reward. The model cannot grasp the exact travel distance of the vehicle, because the vehicle position is recognized only when the FindVehicle rule fires.

4.4.3. A Solution to the Problem of Conflict Resolution

To date, several researchers have investigated applications of reinforcement learning to complex tasks that are broken into several subtasks (Singh, 1992; Sutton et al., 1999; Doya et al., 2002). Gating is a mechanism used for such tasks. At each point of the task, the agent selects a subtask, and switches to use a corresponding module that independently learns the utility of the primitive action.

Subtasks in the present study correspond to auto and manual operation. To select these modes, a gating mechanism is applied before selecting the primitive rule. As represented in the EDFT model by Gao and Lee (2006), adaptation to an automated system is regarded as a problem of balancing between self-confidence (SC) of manual operation and trust (T) of an automation system. Therefore, in this study, we assumed that the gating mechanism compares SC and T to select the auto and the manual modes. In this research, these utility values are represented as competition between two rules that can be kept in ACT-R’s production module. In our model, SC can be interpreted as a utility of a rule representing the manual mode. On the other hand, T can be regarded as a utility of a rule representing the auto mode. Thus, KeepM and KeepA correspond to SC and T, respectively. In this
paper, we call this mechanism, what might be seen as a type of buffer holding a specific type of information, the “gate module”.

The gate module controls the gate that transitions the subtask. In our model, the transition between modes becomes possible only when the gate is opened. This mechanism can be implemented by including the following conditions in the ToAuto and ToManual rules.

- The condition of the gate module in ToAuto: \( SC < T \)
- The condition of the gate module in ToManual: \( SC > T \)

In other words, prior to comparing utilities of primitive rules, \( SC \) held by the gate module is compared with \( T \). These values change only when the corresponding mode is selected. The value of \( SC \) decreases when the manual mode is selected and the vehicle is off the line. When the auto mode is selected and the vehicle is off the line, the value of \( T \) decreases. When the order relation between \( SC \) and \( T \) changes, the gate opens, and it becomes possible to fire the rule to switch the mode. By providing such a mechanism, the model is able to make decisions based not only on the utility within the current mode, but also with the mode used in the past.

5. MODEL SIMULATIONS

We now present two simulations of the model, demonstrating its performance on the two conditions in Maeghigashi et al. (2013) after presenting a test of our implementation of the line-following task.

5.1. Simulation 1: Base-Level Simulation

Before presenting a simulation involving the two modes, we conducted a simulation of Experiment 1 by Maeghigashi et al. (2013) to confirm the correspondence of the base performance of the auto and manual modes in the two implementations.

5.1.1. Method

Maeghigashi et al. (2013) compared the performance of the manual mode with the auto mode for each corresponding \( Ca \) and \( Cm \) level. In their experiment 1, the participants conducted the task without using the auto control mode (Data-Manual: \( n = 65 \)). We ran the model more than the number of participants because we wanted firm predictions from the model (Ritter et al., 2011).

Similarly, we ran the model with the initial control mode set to manual and removed the ToAuto rule from the model (Model-Manual: \( n = 1,000 \)). We also compared baseline auto performance between the original environment implemented in Java (Java-Auto: \( n = 65 \)) and the simulated environment implemented in ACT-R’s interface, AGI (CL-Auto: \( n = 1,000 \)). This comparison was performed to make sure our replicated task environment followed Maeghigashi et al. (2013)’s original environment.

5.1.2. Results

Figure 6 indicates the performances of the four conditions in each \( Ca \) and \( Cm \) level, showing the ratio of time that the vehicle is on the line. From this figure, it can be observed that the performance of all four lines increases with higher \( Ca/Cm \) levels, consistent with the manipulations of capability. In addition, we can confirm that the auto controls are better than the manual controls in both the experimental data and the simulation. This result illustrates the manual disadvantages in this task. Although the performance of the model is relatively lower than that of the data, the correlations between the experiment and the simulation are high (Auto: \( r^2 = 0.993, RMSE = 0.022 \), Manual: \( r^2 = 0.999, RMSE = 0.014 \)). These results indicate that we succeeded in implementing a firm base for the simulation involving the choice of the two modes.

5.2. Simulation 2: Simulation With the Two Modes

This simulation is conducted to simulate Experiment 2 by Maeghigashi et al. (2013) that specifies the automation use ratio.

5.2.1. Method

In Experiment 2, participants could use the auto control mode. The results were shown in Figure 3. In this simulation, to examine the adaptation process in this task, we manipulated the methods of rewarding and conflict resolution as follows.

- Rewarding
  - Default: The model receives a reward according to the method in section 4.4.1.
  - SMDP: The model receives a reward according to the method in section 4.4.2.

Although we used the same algorithm and parameters as the original experiment, slight differences occurred in the automatic modes used in the two task implementations due to random factors such as the random number seed and the actual course segments chosen.
Conflict resolution

- Default: The model switches mode according to the method in section 4.3.
- Gating: The model switches mode according to the method in section 4.4.3.

Combining the methods of rewarding and conflict resolution, four models [Default-Default, Gating-Default, Default-SMDP, Gating-SMDP] were prepared. Each model was run 500 times under the same conditions as the participants. All models have the default learning rate ($\alpha = 0.2$), the default noise level ($\epsilon$ in Equation 1 and $\text{egs}$ in the ACT-R system) added to the expected utility value of each rule.

5.2.2. Results

The results of the simulation are shown in Figures 7–9. Similar to Figure 3, these show the performance (the ratio of time that the vehicle was able to follow the line), the auto use ratio, and the number of mode switches, respectively. All figures contain the results of the four models with the green (dark) planes while the human participant data initially presented in Figure 3 are indicated with the meshed plane (the mesh being close to the planes indicates a good fit). Most of the models in the figures replicated the trends observed in the behavioral data: the increase of performance as $Ca$ and $Cm$ increase, the increase of the auto use ratio with as the $Cm$ level decrease, the decrease of the auto use ratio as the $Ca$ level increases, and the decrease of the number of switches as the $Ca$ and $Cm$ levels increase. However, the models without the SMDP rewarding failed to replicate the influence of the $Cm$ level in the auto use ratio; these models were not influenced by the capability of the manual mode.

To quantify the fit between the model and the data, we computed $R^2$ and the Root-Mean Square Errors (RMSE) as shown in Table 1. Although the degree of the fit varies between the three behavioral indices, we can observe the best fit of the model includes the two new model features (Gating-SMDP) for all the behavioral indices. This result generally supports the validity of our model. Especially in the performance and the automation use, the model with the two solutions achieved a high fit to the data. However as shown in the number of switches (Figure 9), there are still differences between the model and the participants; the model made more switches compared to the participants.

6. CONCLUSION

The present study simulated adaptations of automated vehicle operation in a simple tracking task. We assumed that a general paradigm of reinforcement learning can be applied to this problem. Based on this assumption, we used the ACT-R architecture to explore mechanisms to simulate the human participant behavior data. The results of the simulation show overall correspondence with the experimental data, suggesting the validity or at least usefulness of our assumptions. In this final section, we presents implications and limitations of the study.
6.1. Implications

As an implication from the current study, we first claim the advantage of connecting an abstract theory (EDFT) to a cognitive process model (ACT-R). Figure 10 summarizes the process of our model (Gating-SMDP) in the framework of the EDFT theory. The utility module of ACT-R corresponds to Belief and Trust in the EDFT theory. However, unlike the previous model, our model has knowledge to execute the task and performs the task. Our model also does not perceive Cal/Cm directly. Randomized course conditions influence the performance (success/failure) of the task. Moreover, complex perceptual/motor factors are involved in manual mode performance. Therefore, the model’s reliance on the automation interacts with the performance of the task in our model, which in turn influences reliance on the automation. As Bainbridge (1983) noted, to understand decision making about the use of automation, one needs to consider monitoring the performance of the auto and manual performance. This work is a first step to include performance factors into modeling the use of automation though there are still many limitations to applying it to actual systems in the real world.

The present work can also be seen as a contribution to both the field of reinforcement learning and ACT-R cognitive modeling. ACT-R has so far been used to simulate many psychological experiments. Because of this background, this architecture made it possible to produce a simulation that can be directly compared with the participant data. In our study, the perceptual and motor modules of ACT-R were used to represent the time constraints of the task (as shown in Figure 5). However, to simulate the human behavior, we needed to extend the reinforcement learning implemented in ACT-R. These extensions included the summation of rewards over time and the gating mechanism. Without these previously developed for SMDP (Sutton et al., 1999), we could not have achieved as good a fit to the data (Figures 7–9).

Combining these different modeling approaches allows us to predict human behavior in time-critical tasks. Despite the long history of this architecture, studies using reinforcement learning in ACT-R have not been so common. In this community, instance-based learning (IBL), which uses declarative knowledge as past problem-solving experiences, is more popular than reinforcement learning (Lebiere et al., 2007, 2009). However, because of the long time delay in retrieving declarative knowledge, IBL is not suited to model decision making in time-critical tasks, rather it is limited to modeling cognitive processes in discrete-time step tasks (MDP). On the other hand, reinforcement learning in SMDP has been mainly developed in the field of control engineering and robotics (e.g., Asada et al., 1999; Sutton et al., 1999; Doya et al., 2002; Elfwing et al., 2004), which was not directly aimed to make predictions of human behavior.

Contrary to the above previous research, our gating module enables a fast decision making process with a human-comparable time constraint represented in ACT-R. In addition, we hypothesize that this mechanism relates to topics of prediction in the sense of the free energy principle (Friston, 2010), which is a general brain theory integrating information theory, Bayesian inference, and reinforcement learning. In this framework, reinforcement learning in complex tasks is modeled with belief-based schemes (Friston et al., 2016), which calculates a future prediction based on a belief of the world constructed by past experience. Although our current model does not explicitly have a prediction process, the gating module plays a role to hold a belief of the world constructed from past experience. In other words, we consider that the gating module is necessary to include as a prediction process in reinforcement learning in ACT-R.

We also assume that by implementing the prediction process, we can extend our model to include more subtle emotional processes such as disappointment (Joffily and Coricelli, 2013) caused by a large prediction error. Such an emotional process based on a prediction possibly could...
improve the fit of our model especially in the number of switches.

In sum, our study has not only the practical merit of presenting modeling techniques for adaptations on automated vehicle operation but also the theoretical merit emerged from combining ACT-R with reinforcement learning theories.

6.2. Limitations and Future Study

We can also consider other several directions for future studies. The first direction is expanding the task to complex real-world situations. In this study, we chose an abstract simple task to be tractable in the default ACT-R architecture. However, to apply the findings to real-world situations, we need more realistic models of continuous control of steering and speed. To model such factors, we will need to add some functions converting the discrete action of production rules to continuous physical controls. We consider that such modeling can be accomplished by adding motor components in ACT-R like a previous model of driving (Salvucci, 2006).

It is also desirable to explore further factors influencing automation adaptation. In the real world, this problem is not purely a cognitive problem, but it includes social and ethical issues. Although it might not be easy to answer such a problem by cognitive modeling alone, we can accumulate understanding on the individual cognitive process to approach the problem. The other assumed factors influencing automation adaptation include mental workload (Rajaonah et al., 2008; Maehigashi et al., 2014), mental models (Beggiato and Krems, 2013), and the understandability of the system (Maehigashi et al., 2018). Such factors seem to be compatible with a cognitive architecture like ACT-R because it provides a way to represent mental models in declarative memory, which is utilized in IBL. The other possible factors involved in this problem such as anxiety or opportunism can also be modeled referring to recent studies of emotion representations in ACT-R (Dancy et al., 2015; Juvina et al., 2018; Van Vugt et al., 2018). We consider that combining such various levels of mental functions possibly provides a novel explanation of the complex nature of reliance on automation.

**DATA AVAILABILITY STATEMENT**

The raw data and model supporting the conclusions of this article will be made available by the authors, without undue reservation. The demo movie and the source code of the model are available at https://youtu.be/KI3vIuZ2-QQ and https://github.com/j-morita-shizuoka/line-following-tak/, respectively.

**AUTHOR CONTRIBUTIONS**

JM constructed the model and summarized the paper. KM led the project. AM and HT corrected the behavioral data. KK implemented the experimental environment. FR corrected the model and the papers. All authors contributed to the article and approved the submitted version.

**ACKNOWLEDGMENTS**

This manuscript was written by adding the novel simulation results and discussions from the authors’ past conference proceedings (Morita et al., 2011a,b, 2014), with further revisions and editing, and with feedback from two reviewers. Comments from David Schwartz helped improve this paper.

**SUPPLEMENTARY MATERIAL**

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fpsyg.2020.02149/full#supplementary-material
REFERENCES

Anderson, J. R. (2007). How Can the Human Mind Occur in the Physical Universe? New York, NY: Oxford University Press. doi:10.1093/acprof:oso/9780195324293.001.0001

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. (2004). An integrated theory of the mind. Psychol. Rev. 111, 1036–1060. doi:10.1037/0033-295X.111.4.1036

Asada, M., Uchibe, E., and Hosoda, K. (1999). Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. Artif. Intell. 110, 275–292. doi:10.1016/S0004-3702(99)00026-0

Bahner, J. E., Huper, A. D., and Manley, D. (2008). Misuse of automated decision aids: complacency, automation bias and the impact of training experience. Int. J. Hum. Comput. Stud. 66, 688–699. doi:10.1016/j.ijhcs.2008.06.001

Bainbridge, L. (1983). Ironies of automation. Automattica 19, 775–779. doi:10.1002/1096-1080(1983)9:4<0046::AID-ROF5>3.0.CO;2-9

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., et al. (2018). “Neural models of mental state and mental state self-observation,” in Proceedings of the 15th Conference on Cognitive Modeling (Sendai), 3164–3169. doi:10.1109/IROS.2014.6947590

Beck, H. P., McKinney, J. B., Dzindolet, M. T., and Pierce, L. G. (2009). Effects of psychological stressor on cognition. Comput. Math. Organ. Theory 15, 121–147. doi:10.1007/s10588-014-9178-1

Beggiato, M., and Krems, J. F. (2013). The evolution of mental model, trust, self-confidence, and the allocation of control in route planning. J. Exp. Psychol. Appl. 19, 1347–1369. doi:10.1037/a0030399

Bennett, D. (2007). “Modeling human-automation interaction in a unified cognitive architecture,” in Proceedings of the 36th Annual Conference of the Cognitive Science Society (Quebec, QC), 2609–2614.

Boralevi, D., Maehigashi, A., Miwa, K., Terai, H., Kojima, K., and Morita, J. (2013). Experimental investigation of calibration and resolution in human-automation system interaction. IEICE Trans. Fund. Electron. Comput. Sci. 98-A, 1625–1636. doi:10.1587/transfun.E96.A.1625

Carroll, J. M., and Doane, R. M. (2007). “The empirical role of automation in predictive decision making,” in Proceedings of the 2007 Conference of the Cognitive Science Society (Madison, WI). doi:10.1007/978-3-319-91122-9_36

Cleary, T., and Rice, G. (2010). “Making the right choice: choosing between static and dynamic responses,” in Human-in-the-Loop Simulations: Methods and Practice, eds L. Rothrock and S. Narayanan (London: Springer), 97–116. doi:10.1007/978-0-85729-883-6_5

Dancy, C. L., Ritter, F. E., Berry, K. A., and Klein, L. C. (2015). Using a reinforcement learning method to model human-robot interaction. Int. J. Hum. Comput. Interact. 31, 168–217. doi:10.1016/cog.1996.0016

Dasso, M., and Krems, J. F. (2013). The evolution of mental model, trust, self-confidence, and the allocation of control in route planning. J. Exp. Psychol. Appl. 19, 1347–1369. doi:10.1037/a0030399

Doya, K., Samejima, K., Katagiri, K., and Kawato, M. (2002). Multiple model-based reinforcement learning. Neural Comput. 14, 1347–1369. doi:10.1162/089976602753712972

Friston, K. (2010). The free-energy principle: a unified brain theory? Nat. Rev. Neurosci. 11, 127–138. doi:10.1038/nrn2787

Friston, K., FitzGerald, T., Rigoli, F., Schwartenbeck, P., O’Doherty, J., and Pezzulo, G. (2016). Active inference and learning. Neurosci. Biobehav. Rev. 68, 862–879. doi:10.1016/j.neubiorev.2016.05.022

Fu, W.-T., and Anderson, J. R. (2006). From recurrent choice to skill learning: a reinforcement-learning model. J. Exp. Psychol. 135:184. doi:10.1037/0096-3455.135.2.184

Fukuyama, M. (2018). Society 5.0: Aiming for a New Human-Centered Society. Japan SPOTLIGHT, 47–50.
Ritter, F. E., Tehranchi, F., and Oury, J. D. (2019). ACT-R: A cognitive architecture for modelling cognition. *Wiley Interdiscip. Rev. Cogn. Sci.* 10:e1488. doi: 10.1002/wcs.1488

Salvucci, D. D. (2006). Modeling driver behavior in a cognitive architecture. *Hum. Factors* 48, 362–380. doi: 10.1518/00187200677724417

Salvucci, D. D., and Taatgen, N. A. (2010). *The Multitasking Mind*. New York, NY: Oxford University Press.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 484–503. doi: 10.1038/nature16961

Singh, I. L., Molloy, R., and Parasuraman, R. (1997). Automation-induced monitoring inefficiency: role of display location. *Int. J. Hum. Comput. Stud.* 46, 17–30. doi: 10.1006/ijhc.1996.0081

Singh, S. (1992). Transfer of learning by composing solutions of elemental sequential tasks. *Mach. Learn.* 8, 323–339. doi: 10.1007/BF00992700

Skita, L. J., Moiser, K., and Burdick, M. D. (2000). Accountability and automation bias. *Int. J. Hum. Comput. Stud.* 52, 701–717. doi: 10.1006/ijhc.1999.0349

Sternberg, S. (1969). Memory-scanning: mental processes revealed by reaction-time experiments. *Am. Sci.* 57, 421–457.

Sutton, R. S., and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: The MIT Press. doi: 10.1109/TNN.1998.712192

Sutton, R. S., Precup, D., and Singh, S. (1999). Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artif. Intell.* 112, 181–211. doi: 10.1016/S0004-3702(99)00052-1

Taatgen, N., van Rijn, H., and Anderson, J. R. (2004). “Time perception: beyond simple interval estimation,” in *Proceedings of the 6th International Conference on Cognitive Modeling* (Pittsburgh, PA), 296–301.

Taatgen, N. A. (2005). Modeling parallelization and speed improvement in skill acquisition: from dual tasks to complex dynamic skills. *Cogn. Sci.* 29, 421–455. doi: 10.1207/s15316709cog0000_23

United Nations Industrial Development Organization (2015). *Introduction to UNIDO - Inclusive and Sustainable Industrial Development.*

Van Vugt, M. K., van der Velde, M., and Investigators, E.-M. (2018). How does rumination impact cognition? A first mechanistic model. *Top. Cogn. Sci.* 10, 175–191. doi: 10.1111/tops.12318

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Morita, Miwa, Maehigashi, Terai, Kojima and Ritter. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.