The 4th IARP/IEEE RAS/EURON
Keynote lecture
June 17, 2005 Nagoya

Design of Human Interactions with Smart Machines: Lessons Learned from Aircraft Accidents

Toshiyuki INAGAKI

Department of Risk Engineering
University of Tsukuba
Tsukuba 305-8573 Japan; inagaki@risk.tsukuba.ac.jp

Abstract - *Function allocation needs to be dynamic and situation-adaptive to support humans appropriately. Machines have thus been given various types of intelligence. Smart machines can now sense and analyze situations, decide what must be done, and implement control actions. It is true, however, humans working with such smart machines often suffer negative consequences of automation, such as the out-of-the-loop performance problem, loss of situation awareness, complacency or over-trust, and automation-induced surprises. This paper discusses mismatches between humans and smart machines, and gives some viewpoints that are useful in designing sensible human-machine interactions.*

1. Introduction

Many systems in our society are semi-autonomous, where computers control the processes based on directives given by human operators. The configuration of such human-machine systems is called *human supervisory control* (Sheridan, 1992). Why are these systems semi-autonomous, rather than being fully automated? A most obvious reason is that we cannot foresee in the design phase all possible events that may happen during the expected lifetime of the systems. In early days of automation, designers tried to replace human operators by machines for higher efficiency or reliability. However, their attempt was not very successful (Bainbridge, 1983). Actually, human operators have to be on-site to perform tasks of "completing the system design," adapting the system for situations that the designers did not anticipate (Rasmussen & Goodstein, 1985).

For semi-autonomous systems, it is important to determine what humans do and what machines do. The design decision of assigning functions to human and machine is called *function allocation*. The traditional ways of function allocation are classified into three categories (Rouse, 1991): (1) *comparison allocation* which compares relative capabilities of humans versus machines for each function, and allocates the function to the most capable agent, (2) *leftover allocation* which allocates to machines every function that can be automated, and (3) *economic allocation* that tries to find an allocation ensuring economical efficiency. The traditional strategies described above consider "who does what." Such design decisions yield function allocations that are *static*: viz., once a function is allocated to an agent, the agent is responsible for the function at all times.

Though the static function allocations are easy to implement, human operators may not be very happy with them. The leftover and the economic allocation strategies do not reflect human characteristics, and treat the operators as if they were machine elements. The resulting function allocation can be elusive for the operators, and they may have to adapt to the machines unwillingly. The comparison allocation seems to be nicer for the humans than either the economic or leftover allocations. Even when the operators are allocated only functions in which humans surpass machines, the superiority may not hold at all times and on every occasion. Actually, humans may get tired after long hours of operations, or they may find it difficult to perform the functions under time pressure.

The above discussions imply that "who does what" design decisions are not sufficient, but "who does what and when" considerations are needed, which implies that function allocation must be dynamic. A scheme that modifies function allocation dynamically depending on situations is called an *adaptive function allocation*. Suppose that a human and a machine are to perform assigned functions for some period of time. The operating environment may change as time goes by, or performance of the human may degrade gradually as a result of psychological or physiological reasons. If the total performance or safety is to be strictly maintained, it may be wise to reallocate functions between the human and the machine. The adaptive function allocation assumes criteria to determine whether functions have to be reallocated, how, and when. The criteria reflect various factors, such as changes in the operating environment, loads or demands to operators, and performance of operators. The automation that operates under an adaptive function allocation is called *adaptive automation* (Inagaki, 2003a; Moray, Inagaki, & Itoh, 2000; Parasuraman et al., 1992; Rouse, 1988; Scallen & Hancock, 2001; Scerbo, 1996).

Adaptive automation is expected to improve comfort

and safety of human-machine systems. It is well-known, however, that humans working with highly autonomous systems often suffer negative consequences of automation, such as the out-of-the-loop performance problem, loss of situation awareness, automation surprises (see, e.g., Wickens, 1994; Endsley and Kiris, 1995; Sarter and Woods, 1995a; Sarter, Woods, & Billings, 1997). Adaptive automation may not also be free from those negative consequences. Moreover, some types of adaptive automation may seem to violate the assumption of *human-centered automation* claiming that "the human must be maintained as the final authority over the automation" (Woods, 1989; Billings, 1997).

By taking some aircraft accidents as examples, this paper discusses mismatches between humans and smart machines. This paper also tries to give some viewpoints that are useful in designing sensible human-machine interactions.

2. Human Supersirory Control

There are many modern technical systems that are controlled by machine intelligence (or computers) under human supervision. Nuclear power plants, glass-cockpit aircraft, and computerized manufacturing systems are typical examples of such systems. These systems are neatly represented by a *human supervisory control* model (Sheridan, 1992). The model distinguishes four units as depicted in Fig. 1: 1) human supervisor, 2) human-interactive computer, 3) task-interactive computers, and 4) technical process to be controlled.

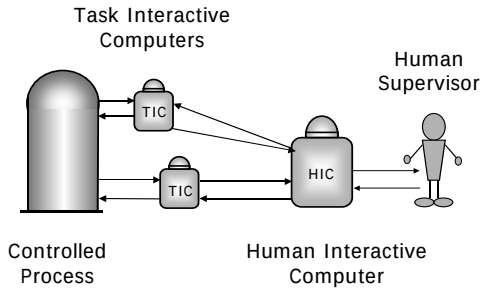

Fig. 1. Human supervisory control

The human supervisor decides what to do and issues commands to a human-interactive computer (HIC) that has capability to communicate with the human supervisor. The HIC understands high-level language to interpret directives given by the human supervisor, provides him or her with system state information in an integrated form, and gives decision aids or alert messages when appropriate. Upon receiving a supervisor's directive, the HIC issues necessary commands to at least one task-interactive computer (TIC). The TIC then performs feedback control by use of its actuators and sensors.

Five phases are distinguished for the human supervisory's control effort (Sheridan, 1992): 1) *Planning* of what needs to be done over some period of time and matching these requirements with available resources, 2) *teaching* the computer what it needs to know to perform its assigned function for that time period, 3) *monitoring* the automatic action to check that everything is proceeding as planned, 4) *intervening* into the automatic action when necessary (such as, in case of an emergency situation or after the completion of a planned task), and finally, 5) *learning* from experience.

3. Mismatches between Humans and Smart Machines

It is recognized that smart machines for aviation automation has been contributing to workload reduction and aircraft safety improvement. Nevertheless, aircraft incidents and accidents still occur. Smart machines can sense, analyze situations, decide what must be done, and implement control actions in highly autonomous manners. It is not easy for humans to keep perfect awareness on mode and intention of smart machines, partly due to difficulty in constructing mental models for various context-specific functions of the automation. Smart machines thus sometimes contribute to incidents or accidents that were not possible in the old days. Let us see some of such examples.

3.1 Mode confusion

Technical systems have been designed to eliminate chance of miscommunication between humans and machines as much as possible. A human operator is requested to push a specific button or write a specific program so that his or her directive may be sent precisely to machine intelligence. Such a design, however, may not always be perfect. The following is a case in which a very clear yet wrong directive was sent to the computer.

Example 1: An Airbus 320 aircraft crashed into mountainous terrain near Strasbourg in 1992. The pilot decided to make an automatic descent using a flight path angle of 3.3° downward, and he gave the computer an input by use of a selector knob on the Flight Control Unit. However, the computer thought that it was ordered to make a descent in the *vertical speed mode* at the rate of 3,300 feet per minute. The misunderstanding occurred because the pilot failed to choose the *flight path angle mode* correctly with a push button for switching between the two modes. Until the very last second, the flight crew did not notice at all that the aircraft was descending at much larger rate than they intended. See, e.g., (Billings,

1997; Sparaco, 1994).

3.2 Loss of situation awareness

The automation (TIC) is *strong* enough to counteract effects caused by an anomaly occurring in the aircraft. However, the automation is sometimes *silent* (Sarter & Woods, 1995b); it does not tell pilots explicitly how hard it is to control the aircraft. Automation with a poker face may cause pilots fail to recognize what is happening. An example may be found in the following in-flight upset incident.

Example 2: A Boeing 747 aircraft dived 32,000 feet near San Francisco in 1985. The rightmost (# 4) engine failed while flying at 41,000 feet on autopilot, and the aircraft began to suffer from an undesirable yaw movement. The autopilot attempted to compensate the yaw movement by lowering the left wing; the rudder could not be used at that time. Pilots were focusing their attentions on the airspeed that was becoming lower. After some unsuccessful trials to increase the airspeed, the captain finally decided to disconnect the autopilot so that he could fly the aircraft manually. Upon the autopilot disconnection, the aircraft rolled to the right, nosed over, and dived steeply until the captain regained control of the aircraft at 9,500 feet. For details, see, e.g., (Billings, 1997).

3.3 Automation surprises

The automation (HIC) may "surprise" pilots by doing what the pilots did not order explicitly. Suppose a pilot directed the HIC to do task A. The HIC may think that task B must be done simultaneously, and may perform both tasks. The pilot would then be confused about the aircraft's behavior. They may say, "what is the autopilot doing, and why is it doing that?"

Example 3: An Airbus 330 aircraft crashed at Toulouse in 1994. The accident occurred in a test flight for investigating performance of the autopilot during an engine-out go-around. The pilot commanded the autopilot *on* at 6 seconds after takeoff. The goal of the autopilot was to climb to the 2,000 feet altitude that had been already set. The autopilot calculated at which point it had to activate the altitude acquisition transition mode (ALTSTAR) to achieve a smooth level-off. The calculation was done while both engines (A330 is a two-engine aircraft) were operating perfectly and the aircraft was climbing very fast, at the vertical speed of 6,000 feet/min. Eight seconds after takeoff, the left engine was reduced to idle, to simulate an engine failure. At the same time, the autopilot activated the ALTSTAR mode, but the pilots did not realize the mode change. Under the simulated engine failure condition, the aircraft could climb only at 2,000 feet/min. To achieve the already calculated climb rate (6,000 feet/min), the autopilot continued pitching the aircraft up. Although the pilots realized that something was wrong, they could not understand what the autopilot was doing, and why. Since there was no pitch limit in the ALTSTAR mode, the pitch angle reached 31.6 degrees. At that stage, the captain disconnected the autopilot. It was too late, however, to regain control of the aircraft. For details, see, e.g., (Dornheim, 1995).

3.4 Conflicts of intention

The HIC has a two-way communication link with each TIC in the engineering human supervisory control model. Through a communication link, the HIC sends its directive to an appropriate TIC, and the TIC returns status information back to the HIC. It may be said that the TIC shares situation awareness with the HIC only within the context of the given directive. Note that each TIC may not have a whole picture of the overall system. The lack of a big picture may be responsible for the failure to resolve possible conflicts of intention between humans and automation.

Example 4: An Airbus 300-600R aircraft crashed at Nagoya in 1994. At some point during the final approach, the pilot gave *unintentionally* a Go-Around directive to the computer. The computer started its pitch-up maneuver. However, the pilot decided to descend for landing. The pilot knew that the autopilot was in the Go-Around mode, but he did not follow an appropriate procedure to cancel the mode. The goals of the pilot and the computer were thus quite conflicting. The computer was ordered by the pilot to go around, and it tried to achieve the go-around at any cost. To the computer, the pilot's input force to descend was a *disturbance* that

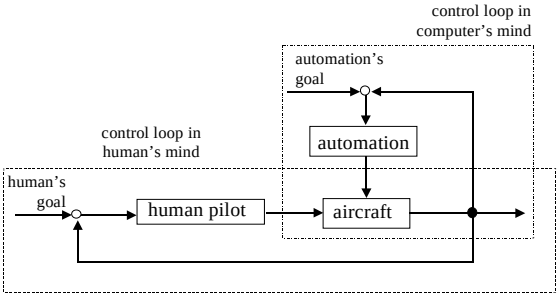

Fig. 2. Control inputs may be disturbances under different goaks or intentions

must be cancelled out by applying a stronger control to the stabilizer for ascending. From the viewpoint of the pilot, the aircraft did not descend smoothly, and he applied a stronger control input to the elevator (Fig. 2). The aircraft was subject to completely contradictory controls by the two agents with opposite intentions. It finally stalled and crashed; see, e.g., (Dornheim, 1995).

4. Design of Artefacts to Assist Humans

Humans perceive the situation, decide what must be done, and implement a control action. In the design of artefacts to assist human operators, it is useful to distinguish the following four classes of functions: 1) Information acquisition, 2) Information analysis, 3) Decision and action selection, and 4) Action implementation.

Example 5: Traffic alert and collision avoidance system (TCAS) is a family of airborne devices designed to help pilots to avoid a mid-air collision (US Dept. of Transportation & FAA, 2000). Its functionalities are described as follows.

1) Information acquisition: TCAS sends interrogations at 1030 MHz that transponders on nearby aircraft respond to at 1090 MHz. By decoding the replies, the position and altitude of the nearby aircraft can be known.

2) Information analysis: Based on the range, altitude, and bearing of nearby aircraft, TCAS performs *range and altitude tests* to determine whether the aircraft is a *threat* or not.

3) Decision and action selection: When the nearby aircraft is declared a threat, TCAS selects an avoidance maneuver (to climb or descend) that will provide adequate vertical miss distance from the threat. If the threat aircraft is equipped with TCAS, the avoidance maneuver will be coordinated with the threat aircraft.

4) Action implementation: TCAS issues the *resolution advisory* to let the pilot know the appropriate avoidance maneuver. However, TCAS does not perform any avoidance maneuver itself. It is the human pilot who implements the avoidance maneuver.

In the above example, information acquisition and information analysis are highly automated. However, the decision and action selected by TCAS are “advices” to human pilots. In other words, a resolution advisory is not an order to be obeyed unconditionally. The pilot may disregard the advisory when he or she is sure that it is wrong. However, the TCAS issue is not so simple. Recall the mid air crash on July 1, 2002, in which two TCAS-equipped aircraft collided over south Germany (Ladkin, 2003; Learmount, 2002). When a conflict developed between two TCAS-equipped aircraft, the TCAS software determined which aircraft should climb and which should descend. One of the aircraft descended according to the TCAS resolution advisory. The other aircraft also descended, although its TCAS told the pilot to climb, which yielded the mid air collision. It would not be hard to imagine how risky it is to disregard a resolution advisory or to do the direct opposite to it, because resolution advisories are “harmonized” between the two aircraft.

Example 6: The enhanced ground proximity warning system (EGPWS) is designed to help pilots to avoid a ground collision (Bresley & Egilsrud, 1997). Its functionalities are described as follows.

1) Information acquisition: EGPWS collects air data, radio altitude, barometric altitude, and airplane position through some other systems, such as Flight Management System, GPS, the airplane air data system.

2) Information analysis: Receiving the above data, EGPWS determines potential terrain conflict by use of its self-contained worldwide airport and terrain databases. EGPWS displays the terrain in dotted patterns with colors indicating the height of the terrain relative to the current airplane altitude.

3) Decision and action selection: EGPWS continuously computes terrain clearance envelopes ahead of the airplane. If these envelopes conflict with data in the terrain database, EGPWS sets off alerts.

4) Action implementation: EGPWS issues a caution-level alert approximately 40 to 60 seconds before a potential terrain conflict, and sets off a warning-level alert approximately 20 to 30 seconds before a conflict. However, EGPWS does not perform any conflict avoidance maneuver itself.

Although information acquisition and information analysis are also highly automated, EGPWS is not given authority for automatic action implementation.

However, there are cases in which automatic action implementations may be essential, even in aviation, for ensuring systems safety. One such example can be seen in the automatic ground collision avoidance system (Auto-GCAS) for combat aircraft (Scott, 1999). When a collision against the terrain is anticipated, the computer gives a “pull-up” warning. If the pilot takes a collision avoidance maneuver aggressively, then the computer does not step in any further. If the pilot does not respond to the warning, the computer takes control back from the pilot and executes an automatic collision avoidance maneuver. In order to describe or consider such automatic action implementations in an emergency, we need the concept of *trading of authority*.

5. Trading of Authority

Trading of authority refers to the human-computer

collaboration in which either one of the human or the computer is responsible for a function, and an active agent changes alternately from time to time. Trading of authority is an essential notion in adaptive automation, because adaptive automation needs to modify function allocation between humans and machines dynamically in response to changes in situations, human workload, or performance.

A scheme to implement trading of control is called an *automation invocation strategy*. There are some types of automation invocation strategies (Inagaki, 2003a; Parasuraman, et al., 1992; Scerbo, 2001). Among them, the following two classes are important for transportation systems.

1) Critical-event strategies: Automation invocation strategies of this class change function allocations when specific events (called *critical events*) occur in the human-machine system. It is assumed that human workload may become unacceptably high when the critical events occur. If the critical events did not occur during the system operation, allocation of functions would not be altered.

2) Measurement-based strategies: Automation invocation strategies of this class adjust function allocation dynamically by evaluating moment-to-moment workload or total system performance. It is necessary to develop *custom tailored* algorithms if the system is to be compatible with individual operators. Individual differences in human operator capabilities will also influence the response to multiple task demands.

6. Viewpoints for Sensible Human-Machine Collaborations

Human-machine collaborations may be heavily dependent on the transportation modes. Some viewpoints may be necessary for identifying functionalities of operator assistance systems. Let us note here two of those viewpoints: 1) quality of human operators and 2) time-criticality (Inagaki, 2003b; 2005).

Quality of human operators varies depending on modes of transportation. For non-professional operators, such as private car drivers, it is not wise to assume that they have high level of knowledge and skills, or thorough and continual training, which implies that required driver assistance functionalities may be quite different from those for professional operators, such as airline pilots or train drivers.

Time-criticality also differs appreciably depending on transportation modes. Suppose a warning has been set off. If it was a resolution advisory of TCAS, the estimated time to closest point of approach must be 15 to 35 seconds, and pilots are supposed to respond to the RA within 5 seconds. If it was a warning-level alert of EGPWS, it must have been issued 20 to 30 seconds before a potential terrain conflict. If it was a collision warning on the car, it may have been given just a few seconds prior to a possible collision.

Noting the above two points, let us discuss how we should design functionalities for assisting human operators appropriately and context-dependent manner. Discussions may be made on two aspects: Enhancement of situation awareness, and design of authority.

6.1 Enhancement of situation awareness

Human interface design is a central issue for enhancing situation awareness, avoiding automation surprises, establishing appropriate trust in automation. The implemented human interface must enable the human to: 1) Recognize intention of the automation, 2) Understand why the automation thinks so, 3) Share the situation awareness with the automation, and 4) Show limits of functional abilities of the automation.

The enhancement of situation awareness matches well with the *human-centered automation* concept, in which *human locus of control* is claimed. However, enhancement of situation awareness does not always assure systems safety. Non-professional operators may not be able to cope with the given situation. Even professional operators, they may not respond to the situation appropriately. Thus, “design of authority” becomes an issue to be considered with a great care.

6.2 Design of authority

Consider the following example in which the computer gave no active help to humans in an emergency, just because it was not asked explicitly by the humans to do so.

Example 7: A controlled flight into terrain (CFIT) accident of a Boeing 757 aircraft occurred in 1995 near Cali, Colombia. The original plan of the southbound night flight to Cali was to fly to the point locating 8 nautical miles south of the airport, and then make a U-turn for the northbound landing on Runway 1. The flight was far behind schedule. When the Cali approach controller offered a straight-in landing to Runway 19, the pilots accepted to make up for lost time. Since their altitude was too high for the new flight path, the pilots extended the speed brakes to expedite the descent. Meanwhile, the pilots became unaware where they were flying, because they supplied an inappropriate command to the flight management computer based on their misunderstanding of a clearance issued by the Cali approach controller. After a couple of unnecessary turns,

the aircraft flew into a mountainous area unintentionally. The aircraft was still descending. When the ground proximity warning system issued a “Pull Up” warning, the pilot responded to it aggressively by pulling up his control column and applying the maximum thrust. However, the pilot failed to stow the speed brakes, and the aircraft crashed into a mountain. Although the pilot was right in his immediate response to the warning, his situation awareness was imperfect in the sense he failed to notice that the speed brakes were in the wrong position; see, e.g., (Dornheim, 1996).

The CFIT accident could have been avoided if there had been an automatic mechanism to retract the speed brake if it had not yet been stowed when the maximum thrust was applied. It is almost impossible to imagine a situation where one would apply the speed brake and maximum thrust at the same time. When automation detects such a *contradiction*, it seems reasonable to allow the automation to adjust the configuration automatically so that the new configuration may fit well to the human’s latest intention. The human may not have enough time to do several things, including detecting and recovering their own errors.

The above discussions lead to the argument, “the automation may be given the right to take an automatic action for maintaining system safety, even when a directive may not have been given explicitly by the operator at that time, if the he or she approved beforehand such automatic life-saving actions in emergency.

One useful tool in discussing the authority issue is the concept of the *level of automation* (LOA). Table 1 gives an expanded version in which a new LOA comes between levels 6 and 7 in the original list by Sheridan (1992). The added level, called the level 6.5, has been firstly introduced by the author (Inagaki, Itoh, & Moray, 1997) to avoid automation surprises that may be induced by automatic actions, when the actions are indispensable to assure systems safety in emergencies.

Table 1. Scales of levels of automation

1.	The computer offers no assistance; human must do it all.
2.	The computer offers a complete set of action alternatives, and
3.	narrows the selection down to a few, or
4.	suggests one, and
5.	executes that suggestion if the human approves, or
6.	allows the human a restricted time to veto before automatic execution, or
6.5	executes automatically upon telling the human what it is going to do, or
7.	executes automatically, then necessarily informs humans,
8.	informs him after execution only if he asks,
9.	informs him after execution if it, the computer, decides to.
10.	The computer decides everything and acts autonomously, ignoring the human.

Inagaki (1999, 2003a, 2003b) argues that the final authority for decision and control may be traded flexibly and dynamically between humans and automation (which is called the *situation-adaptive autonomy principle*), when systems safety is a factor. His argument is based on a series of mathematical results showing that the LOA positioned at level 6 or higher may be necessary to attain safety of technical processes within the framework of a human supervisory control; see, (Inagaki & Johannsen, 1992). The results imply that the conventional human-centered automation principle is not always right from the viewpoint of systems safety.

Example 8. Suppose an engine fails while an aircraft is making its takeoff roll. The pilot must decide whether to continue the climb-out (Go) or to abort the takeoff (No Go). The standard decision rule upon an engine failure is stated as follows: (a) Reject the takeoff, if the aircraft speed is below V1, and (b) continue the takeoff, if V1 has already been achieved. The critical speed V1 is called the “takeoff decision speed” at which the pilot must apply the first retarding means in case of No Go. Inagaki (1997, 2000) proved that the Go/No Go decision should be neither fully automated nor left always to a human. More concretely, (a) the human pilot must be in authority when the aircraft speed is far below V1; (b) the computer must be in authority if the aircraft is almost at V1 and if there is a possibility that the human pilot may hesitate to make decisions when he or she fails to decide whether the engine is faulty or not; and (c) when the aircraft speed is between (a) and (b), the selection of the agent in charge depends on the situation.

Note that the Auto-GCAS is another example of the situation-adaptive autonomy in aviation.

Example 9. When a collision against the terrain is anticipated, the automatic ground collision avoidance system (Auto-GCAS) gives a “pull-up” warning. If the pilot takes a collision avoidance maneuver aggressively, then the Auto-GCAS does not step in any further (i.e., the LOA stays at level 4). If the pilot does not respond to the warning, the Auto-GCAS takes control back from the pilot and executes an automatic collision avoidance action, in which the LOA goes up to level 6 (Scott, 1999).

In Examples 8 and 9, critical-event strategies are adopted for automation invocation (or, trading of authority from human to automation). If it is possible to define a critical event and if technology is available to detect the event, adaptive automation can be constructed easily with the use of critical-event strategies.

7. Concluding Remarks

Intelligent machines have contributed appreciably to human workload reduction, improvement of efficiency and assurrance of safety in various human-machine systems. However, advanced technologies have induced certain types of accidents or incidents that were not possible in the old days.

Adaptive function allocation (or, situation-adaptive autonomy) offers flexible design for human-computer interactions. This very flexibility, however, may bring inconveniences or undesired results when LOA for trading of authority was selected inappropriately. As a matter of fact, some LOAs for measurement-based automation invocation strategies can induce automation surprises; see, e.g., (Inagaki, 2005). Note also that an appropriate LOA can differ depending on transportation modes, and that there is no single adaptive automation that is effective to all the modes. For detailed analyses and discussions on the design of human interactions with adaptive cruise control (ACC) systems or lane keeping support (LKS) systems for advanced automobiles, see, e.g., (Inagaki, 2003c; Inagaki 2005; Inagaki & Kunioka, 2002; Inagaki & Furukawa, 2004; Inagaki, Furukawa, & Itoh, 2005; Furukawa et al. 2003). For discussions on intelligence for mutual understanding of the intent of agents (human supervisor, HIC, and TICs) in the human supervisory control framework, see, (Inagaki & Stahre, 2004).

References

Bainbridge, L.(1983). Ironies of automation. *Automatica*, 19, 775-779.

Billings, C. E. (1997). *Aviation Automation – The Search for a Human-Centered Approach*. LEA.

Bresley, B. & Egilsrud, J. (1997). Enhanced Ground Proximity Warning System, *Safety Bird*, 33, 12-22.

Dornheim, M. (1995). Dramatic incidents highlight mode problems in cockpits. *Aviation Week and Space Technology*, 57-59, Jan 30.

Dornheim, M. (1996). Recovered FMC memory puts new spin on Cali accident. *Aviation Week and Space Technology*, 58-61, Sep 9.

Endsley, M.R. & Kiris, E.O. (1995). The out-of-the-loop performance problem and the level of control in automation. *Human Factors*, 37(2), 3181-394.

Furukawa, H., Inagaki, T., Shiraishi, Y., & Watanabe, T. (2003). Mode Awareness of a Dual-Mode Adaptive Cruise Control System. *Proc. IEEE SMC Conference*, 832-837.

Hollnagel, E. (2003). Prolegomenon to cognitive task design. In E. Hollnagel (Ed.) *Handbook of Cognitive Task Design*, 3-15, LEA.

Hughes, D. (1995a). Incidents reveal mode confusion. *Aviation Week and Space Technology*, 56, Jan 30.

Hughes, D. (1995b). Studies highlight automation 'surprises'. *Aviation Week and Space Technology*, 48-49, Feb 6.

Inagaki, T. (1997). To go no not to go: Decision under time-criticality and situation-adaptive autonomy for takeoff safety. *Proc. IASTED International Conference on Applied Modelling and Simulation*,144-147.

Inagaki, T. (1999). Situation-adaptive autonomy: Trading control of authority in human-machine systems. In M.W. Scerbo & M. Mouloua (Eds.), *Automation Technology and Human Performance*, 154-159, Erlbaum.

Inagaki, T. (2000). Situation-adaptive autonomy for time-critical takeoff decisions. *International Journal of Modelling and Simulation*, 20(2), 175-180.

Inagaki, T. (2003a). Adaptive automation: Sharing and trading of control. In E. Hollnagel (Ed.) *Handbook of Cognitive Task Design*, 147-169, LEA.

Inagaki, T. (2003b). Automation and the cost of authority," *International Journal of Industrial Ergonomics*, 31(3), 169-174.

Inagaki, T. (2003c). New challenges on vehicle automation: Human trust in and reliance on adaptive cruise control systems. *Proc. IEA 2003 (CD-ROM)*, 4 pages.

Inagaki, T. (2005). Design of human-machine interactions for enhancing comfort and safety. Proc. 6th AAET (Automation, Assistance and Embedded Real Time Platforms for Transportation) Conference, Braunschweig, 22-39.

Inagaki, T. & Johannsen, G. (1992). Human-computer interaction and cooperation for supervisory control of large-complex systems. *Proc. EUROCAST '91: Second International Workshop on Computer Aided Systems Theory*, 281-294.

Inagaki, T., & Kunioka, T. (2002). Possible automation surprises in the low-speed range adaptive cruise control system. *IASTED Int'l Conference on Applied Modelling and Simulation*, 335-340.

Inagaki, T., & Stahre, J. (2004). Human supervision and control in engineering and music: Similarities, dissimilarities, and their implications. *Proceedings of the IEEE*, 92(4), 589-600.

Inagaki, T., & Furukawa, H. (2004). Computer simulation for the design of authority in the adaptive cruise control systems under possibility of driver's over-trust in automation. *Proc. IEEE SMC Conferece*, 3932-3937.

Inagaki, T., Moray, N., & Itoh, M. (1997). Trust and time-criticality: Their effects on the situation-adaptive autonomy. *Proc. International Symposium on Artificial Intelligence, Robotics, and Intellectual Human Activity Support for Nuclear*

Applications, 93-103.

Inagaki, T., Takae, Y., & Moray, N. (1999). Automation and human interface for takeoff safety. *Proc. 10th International Symposium on Aviation Psychology*, 402-407.

Inagaki, T., Furukawa, H., & Itoh, M. (2005). Human interaction with adaptive automation: Strategies for trading of control under possibility of over-trust and complacency. (to appear in *Proc. HCII 2005*)

Ladkin, P.B. (2002). ACAS and the south German midair.(http://www.rvs.uni-bielefeld.de/publications/Reports/), 2002.

Learmount, D. (2003). Questions hang over collision. *Flight International*, 8, 2003 July 9-15.

Moray, N., Inagaki, T., & Itoh, M. (2000). Adaptive automation, trust, and self-confidence in fault management of time-critical tasks. *Journal of Experimental Psychology: Applied*, 6(1), 44-58.

Parasuraman, R., Bhari, T., Deaton, J.E., Morrison, J.G., & Barnes, M. (1992). *Theory and Design of Adaptive Automation in Aviation Systems*. Progress Report No. NAWCADWAR-92033-60, Naval Air Development Center Aircraft Division.

Rasmussen, J., & L.P. Goodstein (1985). Decision support in supervisory control. *Proc. IFAC Man-Machine Systems*, 79-90.

Rouse, W.B. (1988). Adaptive aiding for human/computer control. *Human Factors*, 30(4), 431-443.

Rouse, W.B. (1991). *Design for Success: A Human Centered Approach to Designing Successful Products and Systems*, Wiley.

Sarter, N.B. & Woods, D.D. (1995a). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors*, 37(1), 5-19.

Sarter, N.B. & Woods, D.D. (1995b). *Strong, Silent, and Out-of-the-Loop*, CSEL 95-TR-01, The Ohio State University.

Sarter, N.B., Woods, D.D., & Billings, C.E. (1997). Automation surprises. In G. Salvendy (Ed.). *Handbook of Human Factors and Ergonomics*, 2nd Edition, Wiley, 1926-1943.

Scallen, S.F. & Hancock, P.A.(2001). Implementing adaptive function allocation. *International Journal of Aviation Psychology*, 11(2), 197-221.

Scerbo, M. W. (1996). Theoretical perspectives on adaptive automation. In R. Parasuraman & M. Mouloua (Eds.). *Automation and Human Performance*, LEA, 37-63.

Scerbo, M.W. et al. (2001). *The Efficacy of Psychophysiological Measures for Implementing Adaptive Technology*. NASA/TP-2001-211018.

Scott, W.B. (1999). Automatic GCAS: 'You can't fly any lower.' *Aviation Week and Space Technology*, 76-79.

Sheridan, T. B. (1992). *Telerobotics, Automation, and Human Supervisory Control*. MIT Press.

Sparaco, P. (1994). Human factors cited in French A320 crash. *Aviation Week and Space Technology*, 30-31, 1994 Jan 3.

US Dept of Transportation & FAA (2000). *Introduction to TCAS II Version 7*.

Wickens, C.D. (1994). Designing for situation awareness and trust in automation. *Proc. IFAC Integrated Systems Engineering*, 77-82.

Woods, D. (1989). The effects of automation on human's role: Experience from non-aviation industries. In S. Norman & H. Orlady (Eds.). *Flight Deck Automation: Promises and Realities*, NASA CP-10036, 61-85.

Toshiyuki Inagaki received BS, MS, and Doctor's degrees in systems engineering from Kyoto University in 1974, 1976, and 1979, respectively. From 1979 to 1980 he was a Research Associate at the University of Houston, USA. In 1980, he joined the University of Tsukuba, where he has been Professor since 1994, and is Chair of Department of Risk Engineering, University of Tsukuba. He is currently Leader of the MEXT supported special research project, "Situation and Intention Recognition for Risk Finding and Avoidance: Human-Centered Technology for Transportation Safety."

From 1990 to 1991 he was at the University of Kassel, Germany, as a Research Fellow of the Alexander von Humboldt Foundation. From 1996 to 2002, he conducted an international and interdisciplinary research project on human-centered automation at the Center for Tsukuba Advanced Research Alliance.

His research interests include adaptive automation, human trust in automated systems, analysis and evaluation of human interface design, reasoning and decision making with imperfect information under time stress.

Dr. Inagaki is a Senior Member of IEEE, and was Chair of IEEE Reliability Society Japan Chapter in 2003-2004. He received the best paper awards from the ISCIE in 1994 and from the Human Interface Society in 2001.