Avoiding Mode Confusion in Service Robots
The Bremen Autonomous Wheelchair as an Example

Axel Lankenau
alone@tzi.de
Bremen Institute of Safe Systems, University of Bremen,
P.O.-Box 330440, 28334 Bremen, Germany

Keywords: Mode Confusion, Shared Control, Model Checking, Service Robotics

Abstract

This paper introduces a new formal approach to find potential mode confusion situations in shared-control systems such as service and rehabilitation robots. So far, this subject is primarily discussed in the avionics domain. Here, it is motivated why these experiences should be transferred into the service robotics community and how it could be done. The cooperative obstacle avoidance module of the Bremen Autonomous Wheelchair "Rolland" serves as an example to explain the method.

1 Introduction

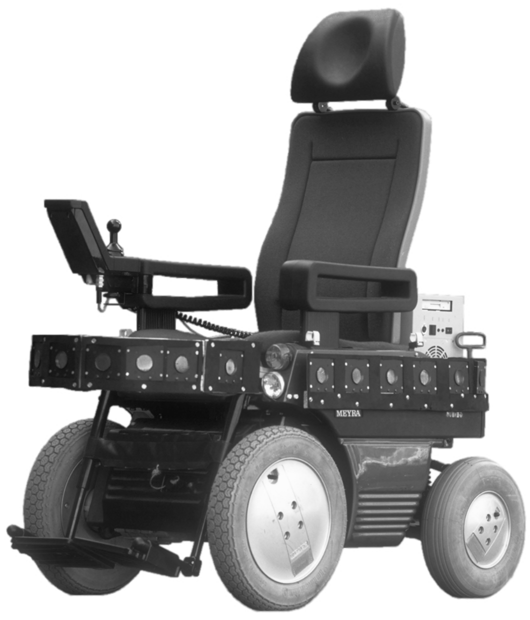
The Bremen Autonomous Wheelchair "Rolland" is a rehabilitation service robot, that realizes intelligent and safe transport for handicapped and elderly people. The system consists of dedicated modules each of which adds certain skills to the platform. The vehicle is a commercially available power wheelchair Genius 1.522 manufactured by the German company Meyra. It has been equipped with a control PC and a ring of sonar proximity sensors (see Fig. 1 *left*).

In contrast to other (autonomous) service robots, Rolland is jointly controlled by its human operator and by a so-called safety module. Depending on the active operation mode, either the user or the automation is in charge of driving the wheelchair. Conflict situations arise if the commands issued by the two control instances contradict each other. How to detect and subsequently avoid such shared control conflicts is the subject of this paper.

2 Mode Confusion

Complex embedded systems usually offer a huge variety of operation modes. In recent publications (e.g., [6, 10]), a variety of causes of problems in shared-control systems have been described. These problems occur if the (controlling) human operator and the (potentially also controlling) technical system have divergent assumptions about the actual system state. Such situations arise, if the mental model of the operator does not match the model of the technical system about the behavior of the whole system.

The following list intends to introduce some categories of mode confusion situations that are especially relevant to the wheelchair application presented later.



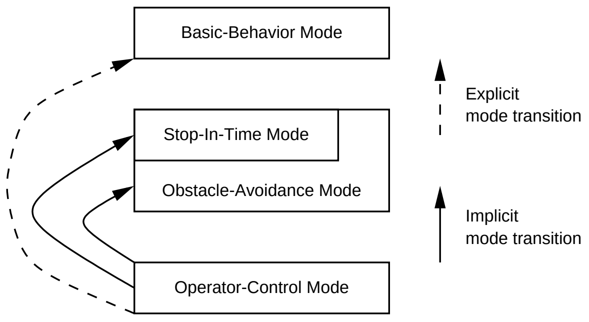


Figure 1: *Left:* The Bremen Autonomous Wheelchair Rolland: A control PC and a ring of sonar sensors for obstacle detection are mounted on a commercially available power wheelchair — *Right:* Hierarchy of its operation modes

Interface interpretation errors occur if the human operator takes an inappropriate action because he or she misinterprets the actual system output, often due to bad interface design. As an example, consider the modes for inserting and overwriting text in a word processor. When changing between these modes, often only a small detail of the word processor's user interface is changed. *Inconsistent behavior of the automation* causes problems because the system does not work as expected. A basic requirement is that the automation must work deterministically so that the same input command from the operator in the same state results in the same output. *Indirect mode changes* happen if the automation changes modes autonomously (without previous operator instruction). In such situations, the operator's mental model is prone to lose track of the real state of the system. *Operator authority limit.* If the automation is authorized to override operator commands under certain conditions, the human operator may suffer from this limitation in unforeseen situations. The standard example is the "escape-from-fire" scenario: if the proximity sensors of a wheelchair robot misinterpret dense smoke as obstacles, the automation might hinder the human operator's escape. *Lack of appropriate feedback.* The automation should indicate any important mode transitions that it performs. It is very difficult to define exactly which mode transitions are "important"; indicating too many transitions probably results in a careless operator after a while, whereas confining the indication of mode transitions to the really important ones may cause the operator's mental model to diverge from the actual system behavior sooner or later.

3 Formal Methods and Mode Confusion

Recent research aims at finding solutions to the shared-control problem with the help of formal methods (e.g. [1]). The basic idea is to check the (formal) specification of the system (including the interaction with the user) for potential conflict situations. The challenging part is to model and to formally specify the behavior of the human operator correctly. As a number of plane crashes were caused by mode confusion problems that occurred between the pilot and the automation, there is a huge industrial interest in these techniques.

Two prominent approaches to be mentioned are the specification language SpecTRM-RL by Nancy Leveson [6] and a model checking method proposed by John Rushby [9]. Leveson et al. present a method to detect potential mode confusions in system specifications [6]. They employ the formal language SpecTRM-RL (Specification Tools and Requirements Methodology Requirements Language) to describe the technical system as well as the human operator. Leveson and colleagues identify a list of errors that are typically made by humans in shared-control systems. Then, the black-box behavior of the automation is analyzed in order to find out which situations are error-prone. Finally, suggestions can be made as to how to improve the system design to avoid the mode confusions. Even though the specification language is automatically processable, the authors are convinced that human experts should do the analysis.

Rushby et al. make use of the generally accepted assumption that the user of a technical system develops an individual mental model which describes the behavior of the technical system [9]. Being a kind of internal representation of the behavior of the automation, such a mental model is the basis for decisions about interaction with the system. By inspection of operator instruction material, generally applicable (abstract) mental models can be derived. Rushby and colleagues use an integrated (finite) state machine to specify the operators mental model as well as to describe the technical system. In a subsequent step, they are able to employ a standard model checking tool to search automatically for potential mode confusions.

4 Safe Cooperative Obstacle Avoidance

In order to illustrate the role of mode confusion in rehabilitation robots such as the wheelchair Rolland, its operation modes and the resulting confusion potential are discussed in the following subsections. Finally, the formal approach to avoid mode confusion situations is sketched (cf., section 4.3).

4.1 Operation Modes of the Bremen Autonomous Wheelchair

The user controls the commercial version (no control PC, no sensors) of the power wheelchair with a joystick. The command issued via the joystick determines the speed and the steering angle of the wheelchair. The idea of Rolland's safety layer (cf., [4, 8]) is to wiretap the control line from the joystick to the motor. Only those commands that won't do any harm to the wheelchair and its operator are passed unchanged. If a command turns out to be dangerous (e.g., approaching too close to a wall), the safety layer intervenes and decelerates the vehicle. Thus, this fundamental module ensures safe traveling in that it guarantees that the wheelchair will never actively collide with an obstacle.

Above the safety module, higher-level skills provide additional functionality: obstacle avoidance (i.e., smoothly detouring around objects in the path of the wheelchair), assistance for passing the doorway, behavior-based traveling (wall following, turning on the spot, etc.) and others. These modules have been combined to the *driving assistant* (cf., [5, 7]). It provides the driver with various levels of support for speed control and for steering. Since this module averts collisions with obstacles and facilitates difficult maneuvers such as doorway passage, it is useful for most people confined to a wheelchair.

The following paragraphs present the hierarchy of Rolland's operation modes, classified in accordance to the amount of control left to the human operator (cf., Fig. 1 *right*).

4.1.1 Operator-Control Mode

In the *operator-control mode*, the wheelchair only monitors the commands issued by the human operator via the joystick. If there is no obstacle close to the wheelchair, the safety module does *not* alter these commands. If there is an obstacle dangerously close to the wheelchair, the automation performs a transition into the *stop-in-time mode*. The notion "dangerously" refers to a situation in which there is an object in the surroundings of the wheelchair that would be hit, if the vehicle was not decelerated to a standstill immediately.

4.1.2 Stop-In-Time Mode

In the *stop-in-time* mode, the safety module overrules every command given by the user and sets the target speed to zero. In addition, it freezes the steering angle to the current value at the moment the stop-in-time mode was invoked. The underlying idea is the fact that you can travel faster in cluttered environments if you ensure that the steering angle remains constant during an emergency brake. Note that in this mode the human operator cannot control the wheelchair in any kind of way. As the driving comfort significantly suffers from frequent activations of the stop-in-time mode, it is basically enclosed within the so-called *obstacle-avoidance* mode.

4.1.3 Obstacle-Avoidance Mode

The obstacle-avoidance mode ensures that emergency braking maneuvers are avoided whenever possible. If in this mode, the wheelchair smoothly decelerates the velocity when approaching an obstacle. During this deceleration, the user is still in control of the steering angle. If the automation realizes that the projected path of the vehicle is free again, it again accelerates up to the speed indicated by the human operator via the joystick. In addition to the smooth speed control, this mode causes the wheelchair to detour around obstacles in its projected path: If there is an object close to the wheelchair *and* the user indicates a travel direction that points to the left or to the right of the object, the automation reduces the speed and changes the steering angle accordingly.

Whereas the transitions between the modes presented so far are implicit (i.e., the technical system autonomously changes operation modes without any indication to or feedback from the user, there is an additional *basic-behavior* mode that has explicitly to be chosen by the driver.

4.1.4 Basic-Behavior Mode

If the system carries out a basic behavior, such as wall following or turning on the spot, it ignores the position of the joystick completely. The only way of intervening during the execution of the autonomous behaviors is to interrupt the behavior with a special command. After the automation successfully carried out such a behavior, the user regains control within the operator-control mode.

4.2 Solving a Mode Confusion Problem: Obstacle Avoidance

Due to some drawbacks of earlier non-cooperative implementations [4, 8], a cooperative obstacle avoidance method was developed [7]. It realizes an intelligent shared-control behavior by projecting the anticipated path of the wheelchair into a local obstacle occupancy grid map. Depending on the side on which the projected path indicated with the joystick passes the obstacle, the algorithm decides how to change the steering angle in order to best serve the user. If instead, the driver directly steers toward an obstacle, the algorithm infers that he or she wants to approach the object and does not alter the steering angle. As a result, both requirements are fulfilled: obstacles are smoothly detoured if desired, but they can be directly approached if need be.

As depicted in Fig. 1 *right*, the transition to the obstacle-avoidance mode is an implicit one, i.e., the mode is not invoked by the user by purpose. Thus, the driver probably does not adapt to the new situation after an obstacle has been detoured, because he or she did not notice that the operation mode changed from operator-control to obstacle-avoidance. It is very likely that the user would not react immediately after the avoidance maneuver and steer back to the original path. Instead, he or she would probably not change the joystick commando. As a consequence, the wheelchair would follow the wrong track.

This mode confusion problem motivates an additional feature of the new obstacle-avoidance algorithm of Rolland: It is able to steer back to the projection of the original path after the obstacle was passed. If the user does not adapt to the new situation, i.e., he or she does not change the joystick position after a detouring maneuver, the algorithm does interpret the command in the frame of reference that was actual when the maneuver began. As a consequence, it is able to navigate through a corridor full of people or static obstacles by simply pointing forward with the joystick. If there is an object that has to be detoured, the user keeps the joystick in an unchanged position and thereby enables the obstacle-avoidance algorithm to steer back to the projection of the original path.

4.3 Idea of the Model Checking Approach

To best fit the intention of this paper, the approach is prototypically introduced on the basis of the cooperative obstacle avoidance module presented above. While traveling, the wheelchair is initially controlled by the human operator. Depending on the environment, the automation may seize control. Both, the automation as well as the user's mental model of this automation, can be described as finite state machines. The states represent operation modes at a certain abstraction level, the transitions represent valid mode changes. Similar to the work of Rushby et al. (cf., [9]), a standard model checking tool (here: FDR by Formal Systems Ltd., [2]) is used to detect mode confusion potential.

Both state machines are modeled as two distinct sets of processes in Hoare's CSP language [3]. CSP-processes engage in events (e.g., communication with other processes, or signals) and proceed from one state to another. The processes comprise disjoint sets of states (modeled as subprocesses), disjoint sets of internal communication channels and variables (e.g., the user's perception of the current speed of the wheelchair may differ from that of the automation), and a shared set of external observables, such as the information whether or not the wheelchair is in a standstill.

The FDR model checking tool is used to carry out refinement checks in the so-called *failures* model. Here, each process is represented by the set of finite event sequences it can perform (its *traces*) and by its *refusals* (sets of events the process cannot engage in after having performed a certain trace). If a process P refines a process Q in the failures model, every behavior of P can also be observed for Q with regard to the traces as well as to the refusal sets. If the process U representing the user's mental model of the automation and the process A representing the implementation of the automation are equal in the failures model, the user will never lose track of the automation behavior *and* the automation will always accept any action of the user that he or she thinks to be adequate in a specific situation.

5 Results and Future Work

So far, the mode confusion analysis has shown that the human operator cannot track the behavior of the automation if the obstacle avoidance module tries to steer back to the original path after an avoidance maneuver. To avoid such mode awareness problems in the future, the human-machine interface will be improved. By means of a speech module, the wheelchair will indicate mode changes. As a consequence, confusing situations in which, for instance, the driving assistant tries to circumvent an obstacle that cannot be seen by the human operator will occur less often. The formal approach briefly sketched in section 4.3 will be generalized to suit a wide range of application domains similar to the one presented here.

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft through the priority program "Spatial Cognition".

References

[1] Butler, R. et al. (1998). *A Formal Methods Approach to the Analysis of Mode Confusion*. In: Proc. of the 17th Digital Avionics Systems Conference. Bellevue, Washington.

[2] Formal Systems (Europe) Ltd. (2000). *Failures Divergence Refinement - FDR2 User Manual*. Oxford, UK.

[3] Hoare, C.A. (1985). *Communicating Sequential Processes*. Prentice-Hall International. Englewood Cliffs, New Jersey.

[4] Lankenau, A. et al. (1998). *Safety in Robotics: The Bremen Autonomous Wheelchair*. In: Proc. of AMC'98, Fifth Int. Workshop on Advanced Motion Control. Coimbra, Portugal. 524-529.

[5] Lankenau, A., Röfer, T. (2001). *The Bremen Autonomous Wheelchair – A Versatile and Safe Mobility Assistant*. In: Intelligent Wheelchairs in Europe, Special Issue of the IEEE Robotics and Automation Magazine. To appear in March 2001.

[6] Leveson, N. et al. (1997). *Analyzing Software Specifications for Mode Confusion Potential*. In: Workshop on Human Error and System Development (Proc.). Glasgow, UK.

[7] Röfer, T., Lankenau, A. (1999). *Ensuring Safe Obstacle Avoidance in a Shared-Control System*. In: J.M. Fuertes (Ed.): Proc. on the 7th Int. Conf. on Emergent Technologies and Factory Automation. 1405-1414.

[8] Röfer, T., Lankenau, A. (2000). *Architecture and Applications of the Bremen Autonomous Wheelchair*. In Wang, P. (Ed.): Information Sciences Journal 126:1-4. Elsevier Science BV. 1-20.

[9] Rushby, J. et al. (1999). *An Automated Method to Detect Potential Mode Confusions*. In: Proc. of the 18th AIAA/IEEE Digital Avionics Systems Conference. St. Louis, USA.

[10] Sarter, N., Woods, D. (1995). *How in the World Did We Ever Get into That Mode? Mode Error and Awareness in Supervisory Control*. In: Human Factors. Vol. 37