Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Reduction of Complexity for the Analysis of Human-Machine-Interaction

Dennis Gamrad and Dirk Söffker
Chair of Dynamics and Control
University of Duisburg-Essen
Duisburg, Germany
dennis.gamrad@uni-due.de, soeffker@uni-due.de

***Abstract*—In this contribution, a concept and principal realization of an additional module within a proposed HMI analysis architecture is developed. The main aspect of this module is the reduction of complexity allowing the analysis of a Human-Machine-System. Core of the architecture is an action model, which is methodical founded on Situation-Operator-Modeling. The action model describes the interaction within a Human-Machine-System and is implemented by high-level Petri Nets. From the Petri-Net-model a state space can be generated to analyze the interaction between a human operator and the environment (in general assumed as machine). The definition of the situation representing the considered part of the real world influences the size of the state space significantly. This contribution realizes the implementation of a size-variable situation vector to reduce the complexity of the considered system. The functionality of the extended architecture is illustrated by the interaction of a human operator with an arcade game.**

***Index Terms*—Man-machine systems, Petri nets, State space methods**

I. Introduction

The development of assistance systems (e.g., for human driver, cf. [1], [2]) supporting human operators during supervisory tasks [3] is a field with increasing significance and a wide range of applications. Typical examples are safety critical Human-Machine-Systems like the control and supervision of the operation of nuclear power plants or airplanes. In these scenarios, the human operator's flexibility is in practice often used as a fallback level to handle unexpected dangerous situations. However, the human interaction behavior is also not free of various kinds of errors (cf. [4]). Here, assistance systems analyzing the Human-Machine-Interaction (HMI) could help in critical situations.

In this contribution, a concept and principal realization of an additional module within a proposed HMI analysis architecture is developed. The main aspect of this module is the reduction of complexity allowing the analysis of a Human-Machine-System. Core of the architecture is an action model, which is methodical founded on Situation-Operator-Modeling (SOM). The action model describes the interaction within a Human-Machine-System and is implemented by high-level Petri Nets (HPNs). From the Petri-Net-model a state space can be generated to analyze the interaction between a human operator and the environment (in general assumed as machine). The definition of the situation representing the considered part of the real world influences the size of the state space significantly. This contribution realizes the implementation of a size-variable situation vector to reduce the complexity of the considered system. The functionality of the extended architecture is illustrated by the interaction of a human operator with an arcade game.

II. Simulation and analysis of human behavior within the HMI-context

In [5] and [6], an approach for automated detection of human errors in human interaction with complex dynamical systems is presented by formal representation [5] and related automatic detection [6]. As a basis of the approach, the interaction between a human operator and a technical system is formalized using a Situation-Operator-Modeling (SOM) approach (cf. [5], [7]). The implementation of a SOM-based model of interactions is realized using Coloured Petri Nets (CPNs, cf. [8], [9]). From the CPN model a full state space can be generated, which contains all possible situations of the system. The considered human errors are described by formal query functions which are used to analyze the state space. The whole sequence from modeling to analysis is described in the following and visualized in Fig. 1.

A. *From the real world to the model*

The interaction between cognitive systems, like humans, higher animals, or specific technical systems and the environment can be formalized using the SOM approach [7]. Core of this approach is the assumption that changes of the considered parts of the real world are understood as a sequence of effects described by the items scenes and actions. The item situation, which is in contrast to McCarthy [10] a time-fixed, system-, and problem equivalent one, is used to describe the internal system structure (as part of the real world). A situation consists of characteristics, which are linked via relations. The item operator is used to model actions which change scenes (modeled as situations) in time. An operator transfers a situation to another by changing the characteristics, the relations or both.

The SOM approach can also be used to formalize human errors [7]. According to Dörner [4], human errors are classified with respect to the interactions of humans in complex dynamical systems. Coming from psychology, word models are typically used to describe and distinguish different

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

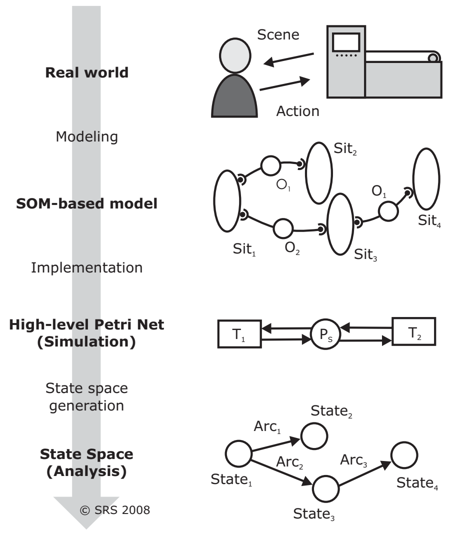

Fig. 1. From modeling to analysis of interaction

human errors, which are divided into four main clusters: goal elaboration, decision processing, control of the actions, and errors due to internal cognitive organization problems. The term 'human error' is used in psychology (e.g., [4]). In general, the term 'error' describes differences to a defined behavior etc. Here, it is also used to describe behaviors which are not optimal (regarding a certain quantity) and lead to undesirable situations respectively. Nevertheless, it has to be mentioned that these kinds of behaviors could also be useful strategies in certain situations.

As an example, the human error 'rigidity' is described and visualized with SOM notation in Fig. 2. The situations (s_i) including the characteristics (a_i, b_i) and relations (r_i) are represented by gray ellipses and the operators (o_i) are represented by white circles. In the situation trajectory depicted in the lower part of the figure, the desired situation s_2 is not reached as planned in the upper part, due to external effects and disturbances. Instead, a different and unexpected situation s_{2a} is resulting. Hence, o_2 will not lead to the desired goal due to the changed situation s_{2a}. The human error 'rigidity' includes that the known and previously planned operator o_2 is nevertheless realized inconsiderately by the human operator, although the assumptions for its application no longer hold.

B. From the model to the simulation

The introduced SOM technique provides a qualitative modeling approach starting from a symbolic notation to structure and investigate human interaction with technical systems. In order to analyze complex systems with many degrees of freedom, computer-based representations of SOM models have to be built and simulated. The computer-based simulation of SOM-based models has formerly been realized in the form of both textual programming languages [11] and high-level Petri Net (HPN) formalisms [12]. The latter approach based on

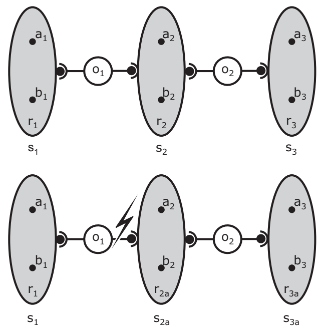

Fig. 2. SOM-based formalization of the human error 'rigidity' [5], [7]

HPNs combines the advantages of a graphical representation with the benefits of formal executable model. According to [13], HPNs are defined as a class of net formalisms, which extends the classical Petri Net formalism originally introduced by Petri [14] (consisting of places, transitions, arcs, and black, uniform tokens) mainly in that HPNs allow several different tokens (such as numerical values, data structures, or complete software objects) at the same time on the same place and often support net hierarchization.

Between SOM and HPNs certain structural similarities exist. They are both inherently bipartite formalisms consisting of an active element representing the system functionality and a passive element representing the system state. In this respect, a natural and intuitive correspondence exists between the notion of operators in SOM and transitions in HPNs. The meanings of the term situation in SOM and the term place (in combination with one or several tokens) in HPNs are also closely related.

Finally, some Petri Net software, such as CPN Tools provided with a well developed repertoire of basic graph theoretic analysis techniques, are existing. These are useful to prove individual properties on SOM-based models and allow analyzing interaction described by SOM-based approaches. The capability of generating and analyzing model state spaces is the main reason for choosing CPN Tools as the basis of the implementation in this work.

C. From the simulation to the analysis

State spaces of CPN models are discrete, directed graphs (digraphs). The nodes of the graph represent different states (markings), which the model may obtain, starting from a certain initial state and given the models constraints. The arcs of the graph represent possible transitions between these states. Adopting an appropriate implementation technique and net patterns to model SOM in CPN Tools as presented in [12], the states are interpreted as SOM situations while the state transitions are associated with SOM operators.

State spaces can be generated automatically in CPN Tools. Depending on the properties of the model and available computational resources, the resulting state spaces may be complete (representing all reachable states) or only partial (representing a subset of all reachable states, e.g., within a certain search depth with regard to the initial condition). Accordingly, it becomes possible to automatically determine a (partial or complete) set of reachable SOM situations and calculate possible SOM-operator-sequences between these situations.

Applying the calculated space of reachable situations is a helpful approach for the investigation of human errors in Human-Machine-System (HMS) interaction, since it allows to relate the observed actions of the human operator to the (known) set of reachable situations and executable operators of the system. On the basis of the state space, it becomes possible to formally determine desirable (goal) and non-desirable (unsafe) states/situations which can be reached by the HMS and then check if the human operator's actions tend to achieve (come closer to) a certain goal situation. It is also possible to observe if an action sequence serves to pursue one single goal consistently or iterates/jumps between various competing goals etc.

III. Detection of human errors based on state-space analysis

The automatic detection of human errors within the interaction of the system can be realized through formal state space query functions programmed in CPN Tools. In order to make the functions reusable for different kinds of Human-Machine-Systems, the queries should be built in a manner that the structure of the error detection is generic and remains independent of the specific system. Only the concrete meanings of desirable/non-desirable goal situations in a certain application context are system-specific and have to be exchanged. As an example, the definitions and state-space-based-detection of the human errors 'rigidity' and 'thematic vagabonding' [4] are described.

The human error 'rigidity' describes a human behavior, in which a human operator adhere rigidly to a previous planned strategy although due to external effects a change would be necessary and more efficient respectively. Through the formalization of 'rigidity' with formal query functions, the automated detection is possible. Therefore, a set of possible final goal situations is calculated using the full state space. From these situations and the observation of user actions, a set of user goal situations (those final goal situations to which the user actions are directed) can be derived. Both the possible final goal situations and the user goal situations are updated after every observed action. The detection of 'rigidity' itself is based on two conditions. The first condition is fulfilled, if a user action is not directed to a final goal situation and the second condition is fulfilled, if the user action was directed to a previous user goal situation. This implicates the occurrence of a user independent action, which was not detected or ignored by the user corresponding to the definition of 'rigidity'.

If human operators change their behavior several times to solve different problems, without finishing them, the human error 'thematic vagabonding' occurs. The automatic detection based on state space analysis can also be realized by the calculation of user goals. For the detection of 'thematic vagabonding', also two conditions have to be fulfilled. If the behavior of a human operator is directed to a certain goal situation and if the previous user goal was different to the current one, not reached, and still possible, the first condition is fulfilled. The second condition is fulfilled if the first condition is fulfilled more than once.

IV. HMI Analysis Architecture

The approach presented in Section II was already realized and tested successfully for different human errors by a simulation environment named HMI Analysis Architecture. In Fig. 3, the connections between real world, modeling, and analysis are visualized. Here, the interaction between a human operator and a (real or simulated) process is modeled using the SOM approach. This action model is implemented with HPNs. From the Petri Net describing the interaction of the real world a state space is generated. An analysis model uses the state space as well as the actions of the real world (modeled as operators) as inputs to evaluate the behavior of the human.

As an example, the interaction between a human operator (gamer) and a computer (arcade game [15]) was chosen. The arcade game enables the design of custom scenarios and can be replaced by other simulated or real technical processes. The modeling and analysis of the interaction is realized by the software CPN Tools.

A. Arcade game application example

In this contribution, the considered specific process is an arcade style game (like 'Boulder Dash' or 'Sokoban') [15] providing a graphical interface to a human operator and a TCP/IP communication with CPN Tools. Within this synthetic environment an agent has to be controlled within a grid-based environment. The agent is able to perform six different kinds of actions ('move up', 'move down', 'move left', 'move right', 'snap field', and 'drop element'). The environment consists of static and dynamic elements with different behaviors combinable to complex scenarios using the integrated level editor. In general, the task of the human operator includes first picking up a certain number of 'emeralds' and then finishing the level by leaving the scenario through an exit door.

In the current state of development, the arcade game was chosen due to its simple handling by editing levels and custom elements. Nevertheless, the structure of the experimental environment and the developed functions for error detection are not specific to the arcade game. This allows the replacement of the arcade game by other simulated or real technical processes using the same concept and error detection mechanisms.

B. Modeling of arcade game interactions

The possible interactions within the arcade game are modeled using HPN-patterns for SOM-based models. In Fig. 4, a part of the action model representing a subset of possible action consisting of the moving actions of the agent, the dropping and explosion of bombs, the falling of stones, and the moving actions of light- and dark-colored monsters is illustrated. All possible actions represented by transitions are connected to the same place. With increasing number of actions, the net can be designed more clearly by the usage of substitution transitions

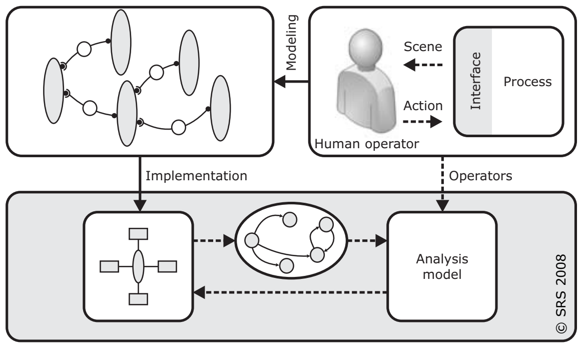

Fig. 3. HMI Analysis Architecture

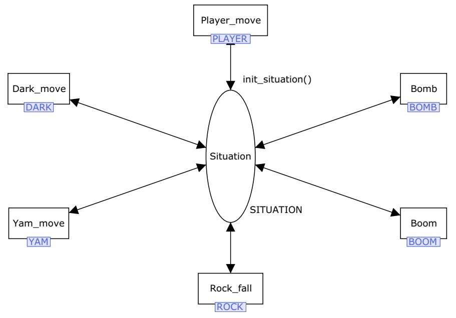

Fig. 4. Modeling of interactions in CPN Tools

and fusion places. The functions of the transitions are detailed on subpages on a lower hierarchical level.

The situation is modeled by a place consisting of one token, which is a set of data types. Here, the characteristics of the situation include ‘x-position of the agent’, ‘y-position of the agent’, ‘vitality of the agent’, ‘x-positions of light-colored monsters’, ‘y-positions of light-colored monsters’, ‘directions of light-colored monsters’, ‘x-positions of dark-colored monsters’, ‘y-positions of dark-colored monsters’, ‘directions of dark-colored monsters’, ‘positions of collectible emeralds’, ‘number of needed points’, etc. In the whole model, the actions of the agent as well as each action of the independent acting monsters are represented by transitions, which are linked to the place. The moving actions of the agent and the moving actions of all light-colored and dark-colored monsters respectively are represented by one transition, which combines several equal operators. Through the firing of a transition the corresponding operator is performed and the situation and token on the place respectively is changed.

V. State-space-based analysis of human interaction in dynamic environments

As described in the previous sections, state spaces can help to analyze the interaction of a human with environments, like technical systems or other humans. Due to the fact that the state space, which contains all possible situations (until a certain search depth), depends on the SOM-based model the definition of the situation has a significant impact on the size of the state space and its application. Here, each dynamic element of the environment influences the measured scene and multiplies the number of possible situations leading to a large state space. These large state spaces can not be analyzed online and also the offline analysis is not possible within an acceptable time.

To reduce the calculation time, the analysis based on multiple partial state spaces was proposed in [6]. Instead of attempting to generate one full state space a priori with all possible system states starting from the initial system state (before actually starting a simulation), multiple partial state spaces of limited size are computed a posteriori starting from observed intermediate states which actually occurred during simulations. Regarding the definition and detection of human errors based on incomplete state spaces (when not all reachable situations and situation trajectories are known) the modeling and the analysis is also influenced. The goal directed behavior can no longer be evaluated with respect to a set of known final goal situations as it is possible in the case of a complete state space. Nevertheless, the offline detection of human errors could be realized faster.

Although multiple partial state spaces can reduce the calculation time, they may contain a large number of irrelevant situations and actions respectively. Hence, for systems with many different dynamical elements, the sequences of alternative actions of the human operator could not be calculated large enough. Furthermore, it is difficult to distinguish whether a waiting phase (no user action and many changes in the environment) of the human operator is goal directed or not.

A. *Size-variable situation for reduction of complexity*

In technical systems, the input from the measured environment can be prefiltered (e.g., by data mining approaches) to reduce the number of characteristics to be processed. The number of these characteristics is usually fixed, which does not influence the performance of the most kinds of technical applications. However, if a state space has to be generated from a certain current situation to plan or analyze the behavior of a cognitive system, every characteristic which is irrelevant for the following actions (until a certain length), would enhance the size of the state space unnecessarily. If a set of situated irrelevant characteristics can be ignored, the analysis would be more effective and faster. A possible solutions for this is given in [7] proposing a situation vector with variable structure.

A size-variable situation vector can also be applied in the action model of the HMI Analysis Architecture. The relevance of a certain characteristic depends on the current situation or the situation in combination with the last user actions as well as the dynamic of the environment. Due to technical reasons, the structure of the situation represented by a composite token can not be changed during simulation, but some characteristics can be set to default values to avoid their influence on the state space to be generated.

The connections between the relevance of the characteristics and (sub) situations, user operators, and the dynamic of the environment have also to be represented within the HMI Analysis

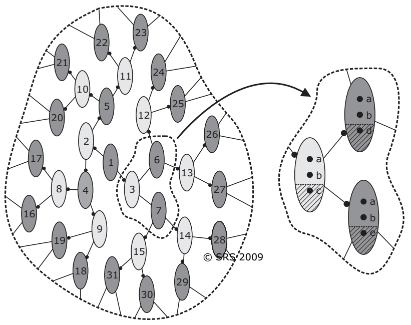

Fig. 5. Partial state spaces and dynamic environments

Architecture. Here, it has to be distinguished between actions of the user and actions of elements from the environment. Thus, several partial state spaces (with different focuses) can be investigated to analyze cooperative or concurrent interaction behaviors between several agents.

In Fig. 5, a partial state space with relevant and irrelevant characteristics and actions respectively is illustrated. According to SOM symbolism, the ellipses represent the situations and the arrows with dotted head represent the operators. Situation s_1 is the current situation and the origin of the state space calculation. The state space contains actions from the human operator and other relevant dynamic elements leading to the light-colored situations and actions from the environment leading to the dark-colored situation changing characteristics which are currently irrelevant. The right part of the figure details the function of the user-independent actions from the environment, which are assumed as not to be relevant in this context. The first two characteristics with the parameters a and b are relevant in the current situation or scenario and the third with the parameter c is irrelevant. If the changing of the third characteristic is ignored, the size of the state space would be reduced.

Furthermore, the size of the calculated state space can be reduced by limiting the possible actions of the human operator at the initial situation. It is assumed that the planned path to the next sub goal does not contain the initial situation more than once. Thus, the possible actions of the human operator are limited to the initialized direction and paths to the assumed irrelevant areas of the environment will be ignored.

B. Technical realization

The proposed functions to reduce the state space are implemented by an additional module named 'State Space Reducer (SSR)' for the HMI Analysis Architecture. The SSR module takes the situation containing all measured characteristics and all actions as input. As output, the SSR module modifies the action model of the HMI Analysis Architecture by the activation or deactivation of characteristics and by blocking paths which are irrelevant.

The modifications of the environment are based on a model representing the connections between the relevance of the characteristics and (sub) situations, user actions, or actions of dynamic elements in the environment. Here, the model is realized by several places, each place for every action from the human operator or the environment. On these places lay composite tokens storing the connection between (sub) situation s_{sub}, last action o_{last}, and token activity $true/false$ (connection tokens). The structure of a connection token is given with (s_{sub}, o_{last}, $true/false$).

During the simulation, the SSR module checks the occurred actions and their corresponding initial situations for every dynamic element if related connection tokens exist. If this is not the case, the action model is not changed. Otherwise, the corresponding characteristics will be activated and deactivated respectively. The deactivation of a characteristic means that this characteristic is set to a default value (or removed from a list etc.). If a previously deactivated characteristic has to be activated again, the current situation is used to set the characteristic to the corresponding value. Finally, the irrelevant parts of the environment can be ignored by removing the last field of the environment, which was entered by the agent, from the action model. After the modification of the action model, the state space is generated again and is used for the next analysis.

VI. Experimental simulation example

In the following, the functionality of the extended HMI Analysis Architecture is presented with respect to the reduction of complexity. Within an example scenario, the modification of the action model and the resulting size of the corresponding state spaces are described. Furthermore, the detection of the human errors 'thematic vagabonding' and 'rigidity' based on the reduced state spaces is illustrated.

A. Scenario

In Fig. 6, an example scenario is shown. Here, the agent controlled by the human operator has to pick up one emerald and then enter the exit door in the lower right corner. The light-colored monster moves up and down and can 'eat' the agent. The dark-colored monster can 'eat' the agent as well as the emeralds. Furthermore, it can change its direction at the upper right corner.

The state space of the whole scenario without any modifications by the SSR module results to 43.722 states. To reduce the number of states, for the two monsters (dynamic elements) connection tokens are defined visualized by the four different boxes and arrows in Fig. 6. For example, the two vertical boxes border the fields which are contained as sub situations in the connection tokens of the dark-colored monster. If the agent is within one of these boxes and moves to the direction corresponding to the related arrows the dark-colored monster is activated and deactivated respectively. Hence, an operator o_{down} in the box with the continuous line style leads to a non-consideration of the dark-colored monster in the next state space generation.

B. Simulation and results

The following simulation illustrates the reduction of state spaces caused by the SSR module in combination with the au-

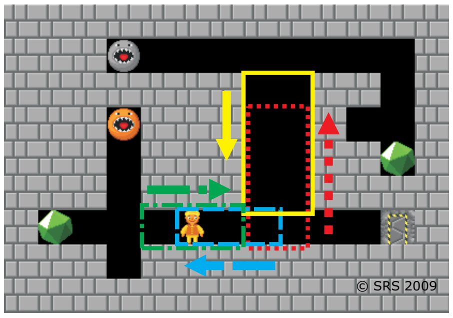

Fig. 6. Example scenario

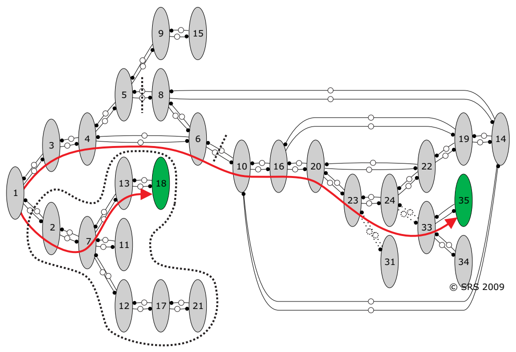

Fig. 7. Graphical representation of the reduced state space

tomated detection of the human errors ‘thematic vagabonding’ and ‘rigidity’ as an example. The agent is in the lower corridor and the monsters start from the left and upper positions of their corridors (cf. Fig. 6). At the beginning, the positions of the monsters are not contained in the situation. Hence, a relative small state space with 139 situations is calculated. The first action of the human operator is ‘right’. Due to this the previous field is blocked additionally and the state space is reduced to 52 situations. The corresponding state space is visualized in Fig. 7 representing all possible actions independently from collected emeralds. The dotted arrows represent a sequence of equal actions. The dotted area is the part of the state space, which is ignored, if the agent performs the operator ($\mathrm{o}_{\mathrm{right}} : \mathrm{s}_1 \rightarrow \mathrm{s}_3$).

Now, it is assumed that the human operator observes the movement of the dark-colored monster and decides to pick up the other emerald (on the left side) and changes the direction ($\mathrm{o}_{\mathrm{left}} : \mathrm{s}_3 \rightarrow \mathrm{s}_1$). Due to this, the light colored monster is activated and the size of the state space increases to 1.988 situations. Then, the human recognizes that the dark-colored monster changes the direction to the left and decides to change the strategy again by performing the operator ($\mathrm{o}_{\mathrm{right}} : \mathrm{s}_2 \rightarrow \mathrm{s}_1$). Due to the fact that the human operator changed the behavior twice, although the aimed sub goal is still reachable and the opposite can not be detected respectively, the human error ‘thematic vagabonding’ is assumed.

If the agent performs the operators ($\mathrm{o}_{\mathrm{up}} : \mathrm{s}_5 \rightarrow \mathrm{s}_8$) or ($\mathrm{o}_{\mathrm{up}} : \mathrm{s}_6 \rightarrow \mathrm{s}_{10}$), the position of the dark-colored monster is added to the situation and the state space increases to 7.787 situations. If the dark-colored monster changes its behavior and moves down and if the agent keeps performing the operator o_{up}, the human error ‘rigidity’ is detected, because the aimed sub goal can no longer be reached.

VII. Summary and Future Work

This contribution presents the extension of the HMI Analysis Architecture by an additional SSR module to analyze Human-Machine-Systems with several dynamic elements. The architecture uses an action model of the interaction between human operator and environment to calculate a partial or complete state space containing all possible situations and actions respectively. The SSR module modifies the initial situation of the action model depending on the last (complete) situation measured from the environment and the last performed user action or change of the environment. By an example simulation, it is shown that the proposed approach is applicable, state spaces and calculation time are reduced and existing algorithms for the detection of human errors are not influenced by the extensions.

In the future, the HMI Analysis Architecture will be applied to other kinds of human errors. Furthermore, the analysis of human interaction behavior in real technical processes will be focused.

References

[1] K. Naab, “Automatisierung bei der Fahrzeugführung im Strassenverkehr,” *at - Automatisierungstechnik*, vol. 48, pp. 211–223, 2000.

[2] M. Maurer and C. Stiller, *Fahrerassistenzsysteme mit maschineller Wahrnehmung*. Heidelberg: Springer Verlag, 2005.

[3] T. B. Sheridan, *Telerobotics, Automation and Human Supervisory Control*. Cambridge: The MIT Press, 1974.

[4] D. Dörner, *The Logic of Failure: Recognizing and Avoiding Error in Complex Situations*. Basic Books, 1997.

[5] D. Söffker, “Understanding MMI from a system-theoretic view - Part I and Part II,” in *Proc. 9th IFAC, IFIP, IFORS, IEA Symp. Analysis, Design, and Evaluation of Human-Machine Systems*, Atlanta, Georgia, USA, 2004.

[6] D. Gamrad, H. Oberheid, and D. Söffker, “Automated Detection of Human Errors based on Multiple Partial State Spaces,” in *MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling*, Vienna, Austria, 2009, pp. 651–659.

[7] D. Söffker, “Systemtheoretic Modeling of the knowledge-guided Human-Machine-Interaction (In German),” Habilitation Thesis, U Wuppertal, 2001, published at Logos Wissenschaftsverlag, Berlin, 2003.

[8] K. Jensen, *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*. Springer-Verlag, 1997, vol. 1-3.

[9] K. Jensen, S. Christensen, and L. M. Kristensen, *CPN Tools State Space Manual*, Univ. of Aarhus, Dep. of Computer Science, Aarhus, 2006.

[10] J. McCarthy, “Situations, actions and causal laws,” Stanford University, Tech. Rep., 3th July 1963.

[11] E. Ahle and D. Söffker, “Interaction of Intelligent and Autonomous Systems Part II: Realization of Cognitive Technical Systems,” *MCMDS - Mathematical and Computer Modelling of Dynamical Systems*, vol. 14, no. 4, pp. 319–339, 2008.

[12] D. Gamrad, “Development of high-level Petri Net Patterns for computer-aided Simulation and Analysis of Situation-Operator-Models (In German),” Diploma Thesis, University of Duisburg-Essen, Duisburg, 2006.

[13] B. Baumgarten, *Petri Netze - Grundlagen und Anwendungen*. Spektrum Akademischer-Verlag, 1996.

[14] C. A. Petri, “Kommunikation mit Automaten,” Dr.-Ing. Thesis, Schriften des Institutes für instrumentelle Mathematik, Bonn, 1962.

[15] (2004, 19th June) Artsoft Entertainment - Rocks’n’Diamonds. [Online]. Available: http://www.artsoft.org/rocksndiamonds/