Int. J. Appl. Math. Comput. Sci., 2021, Vol. 31, No. 4, 713–727
DOI: 10.34768/amcs-2021-0049

A HIERARCHY OF FINITE STATE MACHINES AS A SCENARIO PLAYER IN INTERACTIVE TRAINING OF PILOTS IN FLIGHT SIMULATORS

MAŁGORZATA BACH[a,*], ALEKSANDRA WERNER[a], MAGDA MROZIK[b], KRZYSZTOF A. CYRAN[c]

[a] Department of Applied Informatics
Silesian University of Technology
Akademicka 16, 44-100 Gliwice, Poland
e-mail: {Malgorzata.Bach,Aleksandra.Werner}@polsl.pl

[b] Department of Air Transport
Silesian University of Technology
Krasińskiego 8, 40-019 Katowice, Poland
e-mail: Magda.Mrozik@polsl.pl

[c] Department of Graphics, Computer Vision and Digital Systems
Silesian University of Technology
Akademicka 16, 44-100 Gliwice, Poland
e-mail: Krzysztof.Cyran@polsl.pl

The paper presents the concept of a control unit, i.e., a scenario player, for interactive training pilots in flight simulators. This scenario player is modelled as a hierarchy of finite state machines. Such an approach makes it possible to separate the details of an augmented reality display device which is used in training, from the core module of the system, responsible for contextual organization of the content. Therefore, the first contribution of this paper is the mathematical model of the scenario player as a universal formulation of the self-trained control unit for interactive learning systems, which is applicable in a variety of situations not limited solely to flight simulator related procedures. The second contribution is an experimental verification achieved by extensive simulations of the model, which proves that the proposed approach is capable to properly self-organize details of the context information by tracing preferences of the end users. For that latter purpose, the original algorithm is derived from statistical analysis, including Bayesian inference. The whole approach is illustrated by a real application of training the preflight procedure for the captain of the Boeing 737 aircraft in a flight simulator.

Keywords: Bayesian inference, deterministic Moore machine, flight simulator, hierarchy of finite state machines, scenario player.

1. Introduction

Training is an integral part of working in an aviation sector. Licensed pilots not only acquire, but also regularly renew their rating by completing training for a specific airplane. Also a cabin crew constantly preserves and/or improves its knowledge and skills by undergoing courses of specific training, for example, in the field of technical details of a given aircraft design or mandatory procedures in a variety of situations. This is important because the better the proficiency in the performance of the procedures as well as cockpit understanding, the better the safety of the passengers (Cox, 2020).

In order to reduce costs of such training as well as increase safety of the trainee while practicing flight in extreme conditions, flight simulator devices have been used more and more often since the second decade of the aviation history, when the first simple simulators occurred (Zazula *et al.*, 2013). However, training in flight simulators not always is sufficiently effective. Therefore, sometimes it needs to be significantly prolonged, in particular, during the first years of employment, because a pilot (or, more generally, the trained crew) often faces the lack of easily accessible knowledge on various aspects of airplane construction.

*Corresponding author

To increase the training effectiveness, relevant information can be presented in the form of a smart list to suggest novices the most important content and get them more engaged in the process. This means that the content of the list, including hyperlinks to relevant information, should vary depending on the command which is currently performed by a trainee (context-based learning). Besides, the general information most often used by a large number of course attendees should be placed on top for fast reference.

Taking into account these facts, we propose and describe in the paper the model of an event-based control unit for interactive training in flight simulators. In the following, this unit is referred to as the scenario player as it is responsible for generating various scenarios of aviation procedures. It is implemented as a component of a larger system, the main purpose of which is to develop a new product based on augmented reality (AR) that will be used for the interactive work of service staff and maintenance training, as well as the basic training of pilots and aircraft technicians. The complete system comprises a set of software applications and hardware tools. It is designed to work in three modes:

- training mode,
- maintenance and service mode, and
- learning mode.

This study concerns the training mode, mainly used by pilots to familiarize with the simulator's cockpit and systems of a specific type of an aircraft, as well as the appropriate aviation procedures. The user gets via AR-goggles interactive information about parts of the cockpit, the functionality of various airplane subsystems, and steps necessary to perform normal, abnormal or emergency procedures.

Formally, the proposed scenario player is modelled by a hierarchy of finite state machines (FSMs). The states of the first-level machine are associated with subsequent tasks of the flight procedure (they form a context), while the second level machine enables the efficient handling of contextual information in a form of websites.

Analyzing previous achievements and findings of the research conducted in the area of aircraft crew learning, as described, for example, in the work of Rypulak (2017), we noticed that personalized AR-based training would be particularly valuable when trainees could interactively control the content displayed in intelligent glasses, for example, by using gesture recognition. An additional benefit would be giving them access to supplementary knowledge, in particular knowledge organized similarly to the wiki pages. Taking into account both insights, the objective of the research was to propose a formal and universal model of the control unit, which would be able to support the 'learn-as-you-go' paradigm (Singh and Singh, 2018) by self-learning from actions of the trainees. We present how the scenario player module, which is responsible for responding to input signals by selection of an appropriate virtual scene tailored to the user, can be mathematically modelled by a hierarchy of finite state machines. Such a formal approach is valuable because of its structural simplicity and versatility, which allows the proposed solution to be used in many other contexts. The whole method is illustrated by a real application of training the preflight procedure for the captain of a Boeing 737 aircraft.

The rest of the paper is organized as follows. Section 2 reviews the possibilities of representing the aircraft system event logic and presents the state of the art of pilot training methods. The concept of the interactive training system and the proposed structural model of the scenario player as a hierarchy of finite state machines are included in Sections 3 and 4, respectively. In Section 5 the proposed model is analyzed from a probabilistic perspective, in particular by using Bayesian reasoning, while details of the simulation experiments are described in Section 5.1. There is also a short description of analyzed data as well as information about applied statistical metrics to evaluate the system. The results of the performed tests and the discussion of the outcomes are given in this part of the paper, too. Finally, Section 6 presents the conclusions of the conducted research and provides information about application in actual, new-technology flight simulator system.

2. State of the art

Active, high strained tasks carried out by aircraft team, which demand solving complex problems, require also a non-conventional approach to teaching and training. This fact was already pointed out dozens years ago by one of the most recognizable representatives of the American trend of learning psychology—Robert Mills Gagné (Gagné, 1965), who was involved in the training of Air Force Corps pilots during World War II. In 1965 he created the concept of the teaching process that was tailored to meet the needs of the pilot student, and developed nine sequenced learning events of instruction. Two of them were 'Stimulating Recall of Prior Learning' (Event 3) and 'Presenting the Stimulus Content' (Event 4), according to which ensuring access to prior knowledge and providing a variety of methods for active navigation through the material are very important in acquiring new knowledge.

The traditional method of teaching practical skills, i.e., a graded sequence of manoeuvres on a real plane, which was commonly used in the early stage of aviation development, was not only dangerous for pilots, but also did not allow a large number of aviators to be trained quickly. Besides, it involved huge expenses incurred mainly on aviation fuel and aircraft maintenance (Camilleri, 2018; FAA, 2020). Fortunately, the digital world, which became an integral part of human life many years ago has affected the aviation training process. It has caused the spread of flight and diagnosing simulators, offering genuinely cognitive training for the entire aviation personnel (Eschen *et al*., 2018; Parsons, 2020).

Over time, starting from the 1910s when the first mechanical Antoinette simulator was built (Page, 2004), through the first patented 'Pilot Trainer' created by Edwin Link in 1929, the industry of flight training devices has developed a lot (Myers *et al*., 2018). Thanks to the FSTSC (Flight Simulator Technical Sub-Committee), a sub-committee of the IAFSTA (International Airline Flight Simulator Technical Association) association, which introduced a number of standards and applicable rules, nowadays it is an integral part of pilot education. Changes in the way they are trained in the current era of computer-based learning have made Gagné's pivotal idea a true challenge, because students are often prone to other stimuli, which may distract them from the goal of learning. Therefore, there is a need to adapt the original Gagné's approach and give participants interactive content, which will help them focus and engage with the training steps seen on the screen (Myers *et al*., 2018).

As aircraft systems become more and more complex, also their design, evaluation and testing become quite challenging tasks. Due to the necessity to validate the processes of individual component systems, various methods and techniques are used to present the interdependencies between them and to ensure that they meet demanding requirements. Moir and Seabridge (2008, Ch. 4) present several approaches used to design advanced flight control systems, which reveal the depth and breadth of the issues that need to be addressed in modern aircraft. Fault tree analysis, dependency diagrams, Markov analysis and other ways of representing the specific behaviour of system(s) are only the chosen representatives of the methods that allow us to analyze the aircraft from various perspectives.

In general, the control unit is the core component of computer-based devices as it is responsible for performing a set of instructions by generating the appropriate sequence of microinstructions (Adrego da Rocha, 1999). Designers of digital systems use various approaches to describe and/or visualize the behaviour and logic of the control unit, making it easier to understand how this component works and providing its formal synthesis. Transition graphs, state tables, flow charts, dependency diagrams, state diagrams, Petri nets and many combinations of graph schemes can be used for the description of behavioural layer components of digital systems (Ledermann and Schmalstieg, 2005; Zajac *et al*., 2019). Other examples include specific dedicated languages.

Ledermann and Schmalstieg (2005) present the XML-based language APRIL (Augmented Reality Presentation and Interaction Language) for expressing all aspects needed to create compelling interactive AR content and combine it with UML state charts for describing scenarios. Meanwhile, Zhu *et al*. (2015) use SWRL (Semantic Web Rule Language) to define the logical relationships among the contexts defined in relation to presented augmented reality maintenance system.

Finite state machines (FSMs) are another common way to model the high-level control problem. It is sometimes referred to as abstract automaton and is the most general model of a logic circuit (Stańczyk *et al*., 2007). This method has some disadvantages (Olsson, 2016), however, according to many professionals, e.g., Sklyarov *et al*. (1998) or Adrego da Rocha (1999), it has been a technique of great importance in the last decades, because it provides a general mathematical model to describe a given unit and due to its ease of visualization. Besides, the FSM abstraction seemed to be one of the main and simplest methods to model a system where inputs cause the system to transition to a particular state (Zhang *et al*., 2010; Zajac *et al*., 2019; Young, 2015; Ferdania *et al*., 2021). It is also commonly used for modelling control processes in aeronautics and aviation. For example, Turner *et al*. (2008) mention several FSM-based spacecraft autonomy systems, whereas Rabbath (2013) and Hejase *et al*. (2016) apply FSMs to design the controllers of an unmanned aircraft system. In 2013 a vehicle management system using finite state machines was patented (US patent no. 2013026 1853A1), but the application was directed to flight control law architectures for unmanned and manned aircraft. FSMs are also used by Wang *et al*. (2016) and Spagnolo *et al*. (2018) to develop a model of a complete aircraft electric power system, and by Yon (2015) to model the controllers of the aircraft power supply system.

We chose the FSM formalism to present the architecture of the developed event-based scenario player for flight simulators. Our decision was motivated by the fact that this notation is intuitive and clear. It does not require programing skills, and hence allows domain experts, who are not required to be programers, to participate in the authoring process.

Our scenario player is designed as a control unit of a larger product that will be used for the interactive training of pilots in flight simulators with the use of augmented reality (AR) goggles, such as Microsoft HoloLens 2.

The AR technology has proven to be highly helpful in the educational process (Feiner *et al*., 1993; Bower *et al*., 2014; Khan *et al*., 2019) by motivating and inspiring the students to learn. It also allows them to experience interactions with the environment in three-dimensional space which is the natural workplace of aircraft pilots. Providing more and more realistic image reproduction within the actual cockpit environment, AR makes the required knowledge more easily absorbed by the trainee (Safi *et al.*, 2019; Velichko, 2020; Kearns *et al.*, 2020). Interestingly, the term AR is believed to be coined by the Boeing researcher Tom Caudell in the 1990s, who used this phrase in reference to aircraft maintenance (Caudell and Mizell, 1992). However the first applications which enriched the real scenery with additional information date back to the 1960s (Lee, 2012); currently these first attempts are often referred to as AR technology. Over time, the AR or assisted reality technologies have been used not only for maintenance repair or operation performed by technical staff. They can be also utilized to virtually ‘teleport’ the distant environment of interest where some object (e.g., a vehicle or a UAV) operates, and to immerse it in a selected space of the real environment of the operator (Cyran *et al.*, 2018a; 2018b).

Due to the fact that AR provides also gesture-based interaction with instruments of the cockpit environment, it is increasingly boldly used in pilots’ education. In (Schaffernak *et al.*, 2020) one can find an extensive review of the literature concerning the history and potential domains of the use of AR in aviation. Besides, the authors present the results of the recent survey on preferences of a number of pilots and flight instructors regarding the use of AR in training. In this study most of the respondents found AR a beneficial environment for teaching both theoretical and practical parts of pilot training. Moreover, they claimed that this technology should also be advantageous in simulator training, in particular, in cockpit and emergency procedures’ training. Taking into account the above-mentioned facts, it is not surprising that significant progress toward AR applications is currently envisaged (Grzegorczyk *et al.*, 2019; Markets and Markets, 2019), and the system for which the scenario player is presented in this paper is in this trend.

According to the research report by Valenta (2018) the flight simulator market is supposed to have as high as 30% growth rate in the coming five years, part of which is related to applications of AR, which confirms a more and more intense interest in the avionic industry (Brown, 2017). Goel (2018), Eschen *et al.* (2018) and Dhaliwal (2019) adduce evidence for this claim, pointing such challenges which can be solved by AR-based avionic applications as, for example, predicting maintenance time and need, offering novel support avenues to the aviation technicians who want to improve their own productivity and/or want to share their know-how, finding hidden defects in prototypes of aircraft, real-time interaction with a new item before it ever goes into production, etc. (see Neumann and Majoros, 1998; Haritos and Macchiarella, 2005; Schaffernak *et al.*, 2020).

Cerqueira and Kirner (2012) present the authoring tool basAR (Behavioral Authoring System for Augmented Reality) to create a set of AR-based interactive applications. To organize the application working flow and build up a list of necessary objects, they use flowcharts, mind maps, and state diagrams.

Our contribution to the presented above picture of contemporary interactive learning of pilots lies in separation of the details of the AR display device which is used in that training, from the core module of the system, responsible for contextual organization of the content. For that purpose, we formulated a mathematical model of the scenario player as a universal self-trained control unit for interactive learning systems, which is applicable in a variety of situations not limited solely to flight simulator related procedures. The second contribution is an experimental verification achieved by extensive simulations of the model, which proves that the approach proposed is capable to properly self-organize details of the context information by tracing preferences of the end users.

3. Concept of the interactive system

Figure 1 shows an architecture of the designed system which is able to create mixed reality by interoperating with a professional flight simulator in order to facilitate the process of training as well as maintenance of the flight simulator. Main hardware components and their interactions relate to specific objects within the entire system.

The crucial element of the system is the control unit which communicates with the trainee, who belongs to the flight crew or is, e.g., a service engineer. It is responsible for playing various scenarios, such as flight procedures or maintenance and service sequences, which are recorded in persistent storage. Required information between the main computer system and a human is transferred using the universal interface equipped with appropriate sensors, i.e., AR glasses. It allows recognition of the specific real-world objects, in order to update the augmented reality scene. It also gives the user the opportunity to interactively control the flow of the procedure being played. This means that the same scenario may be carried out differently depending on the learner’s prior knowledge, i.e., depending on the number of ‘prompts’ the student will go through.

In such a system, training participants can have access to additional knowledge organized similarly to the wiki site, and they can individually control the content displayed in smart glasses. Texts tailored as per user’s preferences, which are displayed on the augmented reality scene after clicking a hyperlink, can be enriched with images, videos, and other multimedia elements (see

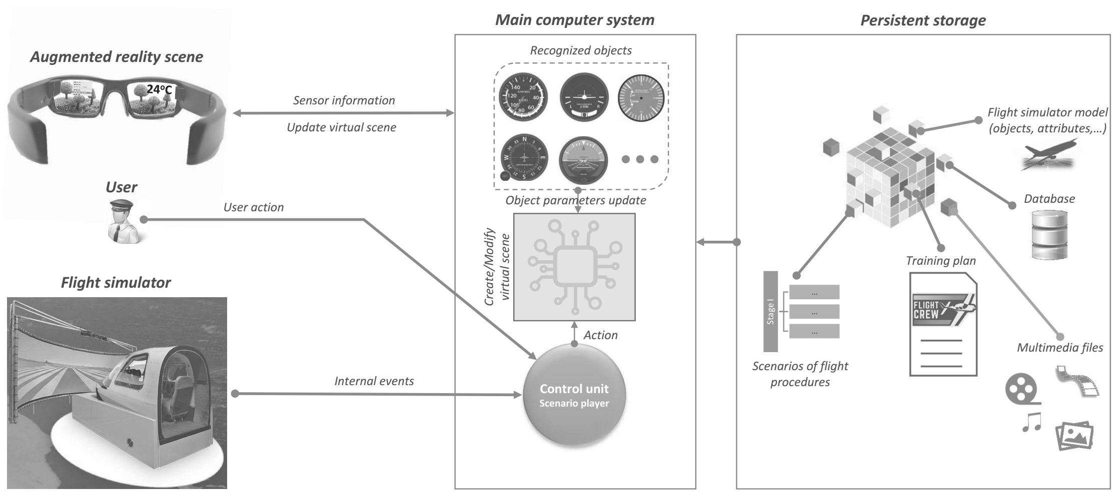

Fig. 1. Idea of an interactive system for supporting training in the flight simulator.

persistent storage content in Fig. 1), which may contribute to broadening the learner's knowledge.

Importantly, tasks performed as part of a given procedure may have a complex structure, which means they may include subtask(s). An example nested structure of one of the flight procedures is presented in Fig. 2.

The scenario player should be able to control the process for both plain and structured operations as well as the user nested choices associated with wiki searches. Therefore, its mathematical model corresponds to the hierarchy of finite state machines, primary and secondary ones. Transition between the machines will be triggered by the selection of the hyperlink made by the individual learner.

4. Formal model of the control unit

The mathematics of FSMs is a convenient formal tool for event-based control logic. Therefore, we use this formalism to model complex operation of the scenario player. There exist two types of FSMs: Moore and Mealy's machines (Stańczyk *et al*., 2007; Barkalov *et al*., 2020). In the described representation the control unit is defined as a Moore FSM that is the sextuple (Plummer *et al*., 2019; Giantamidis *et al*., 2019)

$$M = (Q, \Sigma, O, \delta, \lambda, q_0), \tag{1}$$

where

- Q is a finite non-empty set of internal states,
- Σ represents a non-empty set of input symbols,
- O represents a non-empty set of output symbols,
- δ is a transition function; $\delta : \Sigma \times Q \mapsto Q$,
- λ is an output function defined for Moore's type of FSM as λ: $Q \mapsto O$,
- q_0 represents an initial state.

Any FSM is completely specified (or just complete) if its functions δ and λ are defined for any pair of (q, s) from the Cartesian product of sets $Q \times \Sigma$. If there exists at least one pair (q, s) from the set $Q \times \Sigma$ for which at least one of the functions δ or λ is undefined, then the FSM is not completely specified. Therefore, a Moore FSM $M = (Q, \Sigma, O, \delta, \lambda, q_0)$ is completely specified if and only if $\forall (q, s) \in \Sigma \times Q, \exists\, \delta(q, s) \wedge \exists\, \lambda(q)$, otherwise the Moore FSM is not completely specified (Stańczyk *et al*., 2007).

Our system assumes that the user (while being in some state, say q_i) not only can respond by setting the expected switch (or more generally by performing some expected action), but also can interrupt the main training course by selecting a description of one of the topics relevant to that state q_i.

The FSM implementing our scenario player is therefore not completely specified, as there exist many such states for which operation is undefined for a given input symbol (because such a symbol in that state will never occur; for example, for state q_i only links to topics relevant to that state are presented to the user, and, therefore, only such input alphabet symbols which are associated with these particular links can occur in q_i, while input symbols associated with irrelevant topics will never occur at this state).

Moreover, in order to implement such an interactive scenario player with above-mentioned functionality, the underlying complex (and non-deterministic) FSM can be structurally decomposed to a hierarchy of two levels of much simpler FSMs: Master M^1 and Slave M^2.

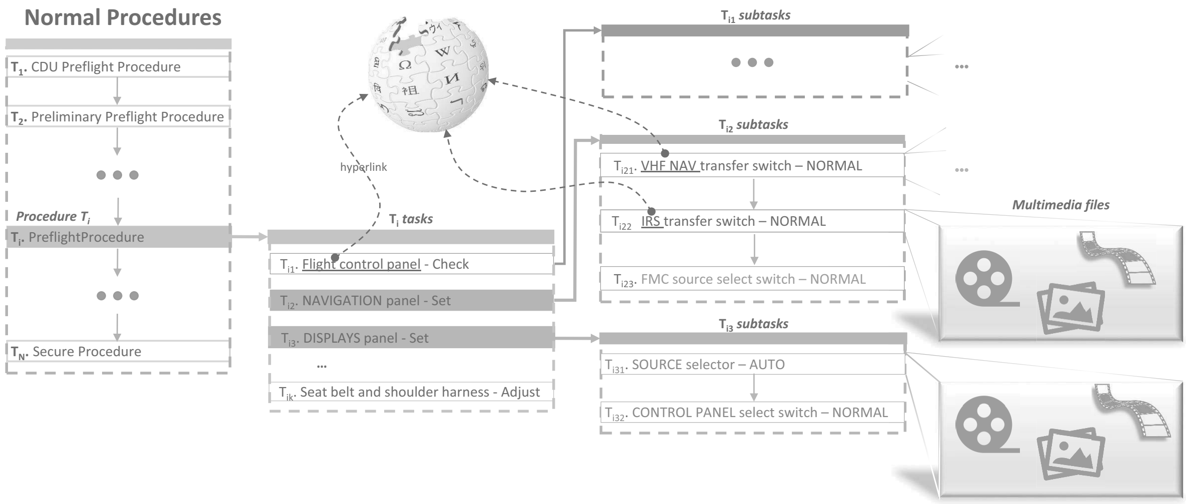

Fig. 2. Tasks and subtasks of the preflight procedure—clipping.

Furthermore, a non-deterministic second level FSM can be determinized not only by using the general determinization approach which results in the initial number of states of the deterministic FSM equal to 2^n in order to implement the non-deterministic FSM with n states (the subject for a further minimization process). Importantly, by utilizing the regularities in the way how the system responds to human actions (irrespective of which state M^1 is in and which particular topic description has been selected) it is possible to transform one predefined (and common for all states of M^1) second-layer non-deterministic FSM to a set of k (k is the number of states in Master machine M^1) deterministic FSMs (dynamically defined by the user interactions with the system during normal operation). For typical numbers n and k around 50, the number of states in the resulting two-layered structure is around $k \times n = 2500$ compared with $k + 2^n \approx 2^n = 2^{50}$ states resulting from the application of the general determination method, which would make this latter method computationally infeasible for subsequent minimization of this enormous initial number of states.

Formally, this concept of structuralization and determinization of the complex FSM to Master M^1 and k Slave M^2 FSMs (Fig. 3) is given below.

The states of machine M^1 are associated with subsequent tasks of the flight procedure. The top level machine is then formally defined as follows:

$$M^1 = (Q^1, \Sigma^1, O^1, \delta^1, \lambda^1, q_0^1), \tag{2}$$

where

$$\begin{aligned} Q^1 &= \{S_0^1, S_1^1, \dots, S_n^1\}, \\ \Sigma^1 &= \{0, 1\} \cup E, \\ \delta^1 &= Q^1 \times \Sigma^1 \mapsto Q^1, \\ O^1 &= \{t_0, t_1, \dots, t_n\}, \\ \lambda^1 &= Q^1 \mapsto O^1, q_0^1 = \{S_0^1\}. \end{aligned}$$

S_j^1 represents the j-th state of machine M^1, where $j \in \{0, \dots, n\}$. In our notation 0 in Σ^1 denotes no confirmation of the task execution (condition $approval = 0$ in Fig. 3), 1 means its approval ($approval = 1$ in Fig. 3), whereas E symbolizes the in-system actions/events.[1] Output t_i represents the text corresponding to i-th task.

Because the training participants need to have access to additional knowledge organized similarly to the wiki site, the second level machine should enable the handling of a knowledge base website (wiki). It is represented as the second level machine

$$M^2 = (Q^2, \Sigma^2, O^2, \delta^2, \lambda^2, q_0^2), \tag{3}$$

where

$$\begin{aligned} Q^2 &= \{S_0^2, S_1^2, \dots, S_{n+l}^2\}, \\ \Sigma^2 &= \{h_1, h_2, \dots, h_l, \\ &\quad \text{BACKWARD, FORWARD, CROSS}\}, \\ \delta^2 &= Q^2 \times \Sigma^2 \mapsto Q^2, \\ O^2 &= \{\emptyset, t_{h_1}, t_{h_2}, \dots, t_{h_l}\}, \\ \lambda^2 &= Q^2 \mapsto O^2, \\ q_0^2 &= \{S_0^2\}. \end{aligned}$$

S_j^2 represents the j-th state of machine M^2, where $j \in \{0, \dots, n + l\}$, l is the number of available hyperlinks h, and n is the number of states of machine M^1. Symbol

[1]This is marked as 'internal events' in Fig. 1.

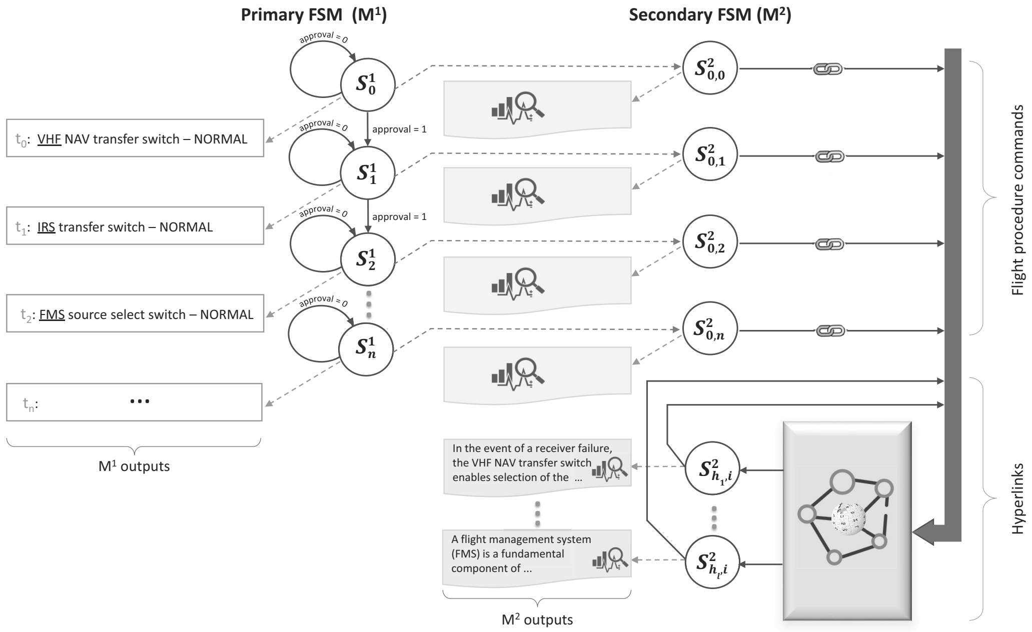

Fig. 3. Hierarchy of state machines—the proposed architecture of the control unit.

t_{h_j} represents the content of wiki page related to the j-th hyperlink, while $\emptyset$ means empty output.[2]

Hyperlinks can be nested, i.e., they may redirect a user to other pages which can also include their own hyperlinks. Therefore, the inputs of subsequent states for the second level machine are not only hyperlinks (e.g., h_1, h_2 or others), but also symbols that allow moving forward (FORWARD symbol), backward (BACKWARD symbol) as well as the symbol of termination—CROSS. The CROSS input makes the machine annihilated. To be precise, it entails destroying of the given instance of M^2 and returning to the state of machine M^1 that initiated it.

The BACKWARD and FORWARD inputs imply that the machine's transitions are not uniquely determined by the current state and the input symbol. They also depend on the prior state, which was active previously, at time $t-1$. Formally, this can be presented as

$$S^2_{h_j}(t) \mid S^2_{h_i}(t-1) \wedge \text{BACKWARD} \Rightarrow S^2_{h_i}(t+1). \quad (4)$$

Since these inputs may make the machine can move to various sequences of states depending on the history of the user actions, the secondary FSM is non-deterministic, i.e., it represents a non-deterministic meta machine.

The above description presents a conceptual model of the second level state machine. In fact, each such machine is a chain of its specific instances, and these specific instances are always deterministic. The indeterminacy of the meta machine is dynamically resolved by interactive user actions, which generates particular instances of a deterministic machine. Example transitions which are the result of user interactions are presented in Fig. 4, where each epoch uniquely corresponds to a specific instance of M^2.

The instances of the second level machine can be initiated by any state of machine M^1. To clarify the idea, an example referring to the preflight procedure is introduced.

Example 1. We use the following notation: $S^x_{y,z}$, where the superscript x refers to the machine level. It takes the value 1 for the primary state machine, and value 2 for the secondary one. The first subscript symbol y indicates the state number. The second subscript symbol z occurs only for the second level machine and corresponds to the machine's instance number.

Suppose that the command *'VHF NAV transfer switch – NORMAL'* is displayed in state S^1_0 of machine M^1 (t_0 in Fig. 3). Such a command contains underlined *'VHF'* word, which means that there is a link to the relevant wiki page with detailed information about VHF radio. If a user clicks the hyperlink *'VHF'*, a second level machine (M^2) starting at state $S^2_{0,0}$ is created. The state output S^1_0 of machine M^1 is an input symbol for state $S^2_{0,0}$ of machine M^2. The next state of this second level (secondary) machine is responsible for displaying the wiki page explaining that *'In the event of a receiver failure, the VHF NAV transfer switch enables selection of the alternative receiver . . . '* ($S^2_{h_1,i}$ in Fig. 3). This explanation appears in a new window while the command displayed in S^1_0 state of M^1 machine is still visible.

[2]This issue is elaborated on in more detail in the next paragraphs.

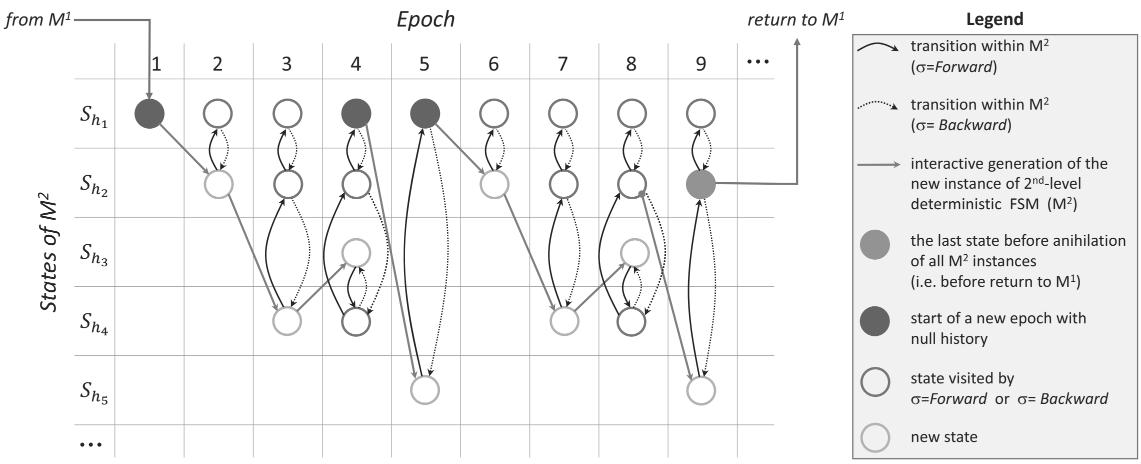

Fig. 4. Example finite state meta machine M^2 as an evolution of its deterministic instances.

As mentioned above, hyperlinks can be nested. This means that a certain website may contain links to other wiki pages. Therefore, the number of states as well as their sequence in the second level machine depend on which hyperlinks the user activates. In other words, when machine M^2 is initialized, its structure is undefined. M^2 is defined interactively by the user. To explain this issue, an example activity of a user is analyzed below. ♦

Example 2. To navigate between pages visited before, the user can use BACKWARD or FORWARD input.

Continuing the analysis of the command from Example 1 ('*VHF NAV transfer switch – NORMAL*'), consider the following sequence of events (Fig. 5(a)).

1. A user is in the first state which refers to the *'switch'* term.
2. The *'gauge'* hyperlink included in the page content is clicked.
3. The *'panel'* hyperlink is clicked.

If the BACKWARD symbol is pressed on the *'panel'* wiki page, the user is returned to the state associated with the *'gauge'* concept. When being in the *'gauge'* state BACKWARD is clicked once again, the user is moved to the *'switch'* state.

The situation will be different if the user goes from the *'switch'* page directly to the *'panel'* one. After clicking BACKWARD on the *'panel'* page, the user is redirected straight to the *'switch'* state. This is shown in Fig. 5(b).

A similar situation occurs when FORWARD symbols are used. The effect of clicking the FORWARD link on a given wiki page depends on which hyperlink was previously, at time $t-1$, active. ♦

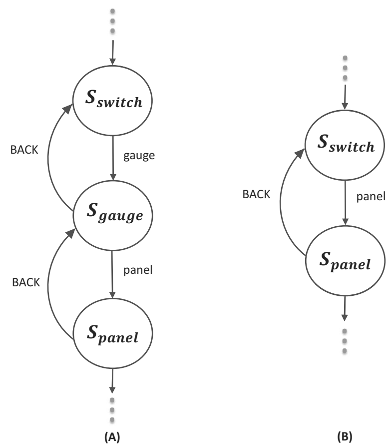

Fig. 5. Example sequence of events.

The set of the output signals for state machine M^2 (cf. Eqn.(3)) contains the symbol $\emptyset$. It refers to $S^2_{0,i}$, the initial state of the machine, and indicates that in contrast to other states, no content directly related to the currently executed task is generated. This does not mean that the generated output is empty, because the system shows the trainee not only the static hyperlinks but also the dynamic ones. The static links are hyperlinks to wiki pages, which are predefined by aviation experts. Meanwhile, dynamic ones are related to wiki pages which have been most often visited by other learners at a given stage of training so their list can change over time. More information on the methods used for selecting this additional content is given in the next section.

As is presented in Fig. 2, some tasks within the considered procedure are complex and consist of subtasks. Therefore, compound states are needed for them. In our model, such compound states may exist in the first level machine. Each given compound state can be modelled as a separate state submachine with the above described two-level structure.

5. Probabilistic logic of the defined Moore state machine

A particular challenge in the implementation of the proposed FSM model is related to generating the list of dynamic hyperlinks, which should be suggested for the user being on the specified page. To achieve that goal, we decided to perform simulations in order to collect data on hyperlinks selected by a certain number of training participants and apply the appropriate statistical analysis.

Our reasoning was as follows. There are k static hyperlinks within a given page (site). When a user selects one of them, there is a movement to the referred wiki page. A user can read its content and then select the successive hyperlink(s). For each user, the process of selecting the next steps is repeated until the required knowledge is finally reached. Based on observations of the regular use of the hyperlinks we found that users who are to take the next step, can be interested in having a list of suggested hyperlinks, hereinafter referred to as dynamic hyperlinks. These hyperlinks refer to other websites that are most visited by others when they analyze the page where the user is currently located.

As shown in Fig. 3, selecting a subsequent hyperlink entails the activation of the new state of machine M^2. Therefore, in the sequel in which the proposed solution is explained, the concepts of state, page as well as hyperlinks are used interchangeably in certain contexts.

With the use of Bayesian statistics (Berger, 1985), the posterior probability of visiting some i-th state (S_i) given the current j-th state (S_j) can be defined as

$$P(S_i \mid S_j) = \frac{P(S_i \cap S_j)}{P(S_j)}, \tag{5}$$

where $P(S_j)$ is the probability of S_j, $P(S_i \mid S_j)$ means the probability of S_i given S_j, $P(S_i \cap S_j)$ stand for the joint probability of both S_i and S_j being true.

In theory, the sequence of steps associated with the selection of subsequent hyperlinks can be unbounded, but in practice it usually consists of at most several steps.

It was assumed that it was irrelevant in which step of the analyzed sequence there was a transition to a given state. In other words, the state visited in the first step was treated as equally important as the other one which was visited further. Consequently, when calculating probabilities, the same weight was assigned to pages visited directly as to those to which the transition took place in the subsequent steps (distant pages). A more detailed analysis of the problem showed that it might have been useful to use a weight factor that considered the

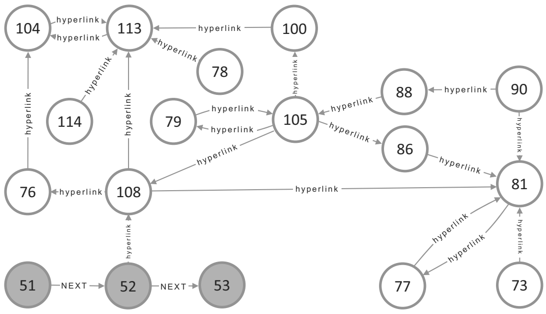

Fig. 6. Transitions between wiki pages accessed from one of the static hyperlinks (108) of PFP command 52.

'distances' between pages. In the reported research the weighting factor was inversely proportional to the step number (i), in which the page was visited, i.e., it was $1/i$.

Anticipating that there may be a situation in which certain states will not be visited at all, the suitable correction is proposed to solve the problem of zero probability:

$$\tau \times (1 - \delta_{ij}), \tag{6}$$

where $\tau = 1$, δ means the Kronecker delta, i is the current state index, and j is the index of the state to which the transition occurs.

5.1. Simulations.

To verify the validity of the proposed logic and prove the functionality of the designed system, the appropriate simulations were carried out. The preflight procedure PFP for a captain of Boeing 737 was selected to demonstrate the way of generating dynamic hyperlinks. PFP consists of 65 commands, which means that the primary state machine M^1 has 65 internal states.

We assumed that 20,000 trainees went through the preflight procedure,[3] performing its 65 commands step by step. Depending on the user's skills and needs, when executing particular commands they could develop knowledge using the available static hyperlinks. For each preflight procedure command as well as for each page related to static hyperlinks from 0 to 5 hyperlinks to other websites were defined. The total number[4] of pages that could be accessed via hyperlinks was set to 50. A selected part of static connections in the form of a directed graph with wiki pages as nodes and hyperlinks as edges is shown in Fig. 6.

We simulated that at any stage of the training each user could select one or more hyperlinks to wiki pages containing information that might have been useful for

[3]The number of users was selected experimentally. It was found that the established number was enough to consider their choices of hyperlinks as representative.

[4]In the figures and diagrams/charts presented in furthers part of the paper, PFP commands are appropriately numbered from 1 to 65 while numbers 66–115 correspond to the wiki pages.

them, or not choose any. To determine the boundary value of the parameter 'Maximal length of the analyzed sequence of visited pages', the bibliography on the subject of the depth in the arrangement of web links (hypertext depth) was analyzed. According to the findings of Zaphiris (2000), users prefer expandable menus for depths of 2 and 3, but for depth 4 their preference is to use a sequential menu. Besides, the study by Parkinson *et al.* (1988) demonstrates that navigational error rates increase significantly as the hypertext depth increases. Also Bernard (2002) compared navigation efficiency through sites of varying depths and breadths, and the results complied with the findings mentioned above as their authors conclude that links leading to 2–3 subsequent items are the most efficient and the least prone to errors.

The full list of parameters for simulations is presented in Table 1.

In the research the users' traversals between hyperlinks were analyzed. Therefore, the $N_{n+l} \times N_l$ matrix F was defined, where elements $F[i, j]$ represented the number of visits to the j-th wiki page given the prior visit to the i-th PFP command or a wiki page. The number of rows N_{n+l} was 115, and it corresponded to the sum of the number of PFP commands (n) and the number of available wiki pages (l). The number of columns N_l was 50, and it corresponded to the number of wiki pages accessed via hyperlinks. The matrix F was the basis for calculating joint and marginal probabilities, and for determining the posterior probability $P(S_j \mid S_i)$ (cf. Eqn. (5)).

As was mentioned in the previous paragraph, the text displayed on pages could include from 0 to 5 static hyperlinks. Because the number of static hyperlinks determined the range of possibilities for further exploration of wiki pages, it seemed to be reasonable to analyze separately pages with a specific number of static hyperlinks, i.e., one, two, three, etc.

In order to better understand the values contained in the matrix F and to compare them, their graphical representation was prepared. The frequency charts for selected pages with 3 defined static hyperlinks are shown in Fig. 7. They relate to preflight command 3 and wiki pages 106 and 108, respectively.

The bars showing the frequency of visiting the pages indicated by static hyperlinks are marked in dark gray. In most cases, the frequency of such pages' visits was high, as expected. However, there were situations, where the number of jumps to pages that were not directly referenced by the given page's static hyperlink was greater than for those indicated by a static hyperlink. Figure 7(c) shows the trainees' behaviour when they studied wiki page 108. It can be noticed that considering 3 first steps in the sequence of visited wiki pages, page 104 was chosen most often, even though it was not directly accessible from page 108. The explanation is that two of the three pages pointed to by static hyperlinks on page 108 had only one static hyperlink defined, just to page 104. It is also visible in Fig. 6.

The charts presented in Fig. 7 show that even for pages for which the same number of static hyperlinks was defined, the distribution of page visits in the three consecutive steps differed significantly. We found that it depended, among others, on how many static links were contained in the nested hypertexts.

As the number of visits to individual pages often varied significantly, the question arose on how many times a page had to be visited to consider it as containing useful knowledge in the context of the analyzed issue, and to suggest it by a dynamic hyperlink.

To answer the question and determine a suitable cut-off point (threshold), appropriate rankings were created. In the consecutive rows of matrix F, the numbers of transitions from the given (i-th) page to others (j-th) were recorded. We sorted the values in an individual row in descending order, and selected the top ten values for each of them. In other words, for each of the 115 analyzed hypertexts, we found the ten most visited pages taking up to 3 transitions into consideration.

Further analyzes were performed separately for pages with a particular number of static hyperlinks. This means that separate calculations were made for pages with k static hyperlinks, where $k \in \{1, 2, 3, 4, 5\}$. We calculated average values for subsequent positions in the rankings for each group. This means that five rankings labelled R_k, containing average values for the pages, were created. The rankings for pages with one, three, and five static links are presented in Figs. 8(a)–(c), respectively.

The values determined in such a way formed a basis for proposing threshold values that had to be taken into account when choosing dynamic hyperlinks which would be displayed for users regardless of static ones. The conducted analyzes showed that the k highest-rank venues were mainly influenced by k static hyperlinks. For example, for pages with three static links, the three greatest values of the above-mentioned rankings of averages referred to visits of hypertexts accessible directly from a given page. The above observations were taken into account when establishing dynamic hyperlinks. For pages with a certain number of static hyperlinks (k), the value of T_k was determined separately. It was defined as follows:

$$T_k = \frac{\text{value specified for the } (k+1)-\text{th position in } R_k \text{ ranking}}{\text{value specified for the } (k)-\text{th position in } R_k \text{ ranking}}. \quad (7)$$

The bars corresponding to the compared values for rankings presented in Fig. 8 are marked with oval outlines. Then, for each page, its static hyperlinks were re-analyzed to find the one for which the number of activations was the lowest. This value is denoted by Min_i, where i corresponds to the page number.

Table 1. Boundary values of parameters adopted for simulations.

Parameter	Value
Number of users	20000
Number of PFP commands	65
Number of wiki pages accessed via hyperlinks	50
Maximal number of static hyperlinks per page	5
Number of dynamic hyperlinks in the window	3
Maximal length of the analyzed sequence of visited pages	3
Number of analyzed positions in the ranking of page visits	10

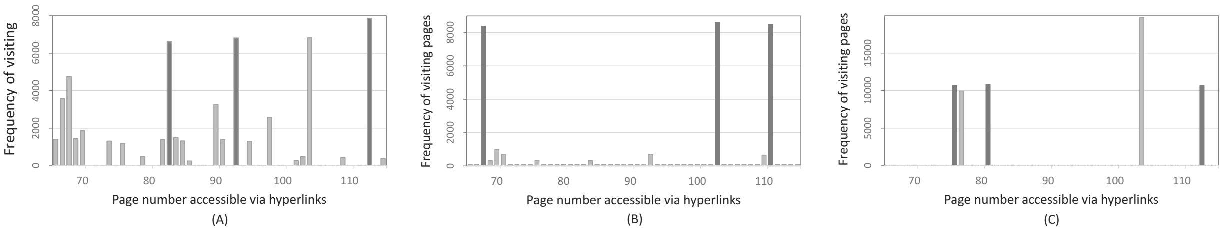

Fig. 7. Frequency of page visiting: PFP command 9 (a), page 106 (b), page 108 (c).

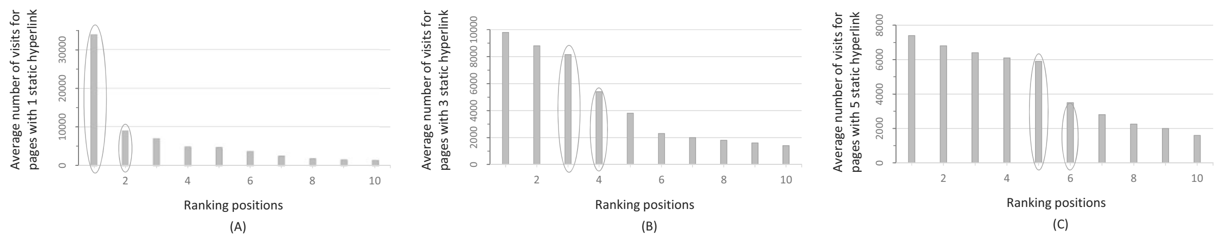

Fig. 8. Average visit frequency rankings: R_1 (a), R_3 (b) and R_5 (c).

When determining dynamic hyperlinks, only those were considered that were not included in the set of static hyperlinks for the given page. Additionally, a candidate for being a dynamic hyperlink for page S_i had to meet the condition that the number of its activations was greater than or equal to $\mathrm{Min}_i \times T_k$, where T_k corresponded to the threshold value determined for the appropriate number of static links (k) of page S_i. Figure 9 shows the number of dynamic hyperlinks for individual commands of the preflight procedure determined in accordance with the described algorithm. The commands that did not have any static hyperlinks defined are marked in white.

It was assumed that the user would be offered three dynamic hyperlinks at each stage of the training. When there were more than three hyperlinks satisfying the conditions, three with the highest number of activations were chosen. In the case where the same number of visits was obtained for several hyperlinks which would be dynamic ones, the weighting factor[5] was used to determine which of them should actually be included in the set of dynamic hyperlinks. In other words, not only the number of activations of a given hyperlink was analyzed, but also on which stage of the three-step sequence it took place.

In a situation where fewer than three hyperlinks met the specified criterion (it was greater than or equal to $\mathrm{Max}_i \times T_k$), the lacking dynamic hyperlinks were suggested based on the global page ranking (top of the top). Analogously, when there were no static hyperlinks accessible from a specific page or PFP command, it was not possible to visit other pages (white bars in Fig. 9). In such a case, a kind of 'dead end' appeared, similarly to an anomaly in a web graph, and dynamic hyperlinks were also suggested on the basis of the global frequency page ranking.

The set of dynamic hyperlinks for individual pages obtained on the basis of the presented simulations was the starting point for the created system. However, as the preferences of pilot students may change over time, this set is going to be periodically updated to ensure that new trainees will receive the most up-to-date information.

[5]The weighting factor $1/i$ was inversely proportional to the step number (i), in which the page was visited.

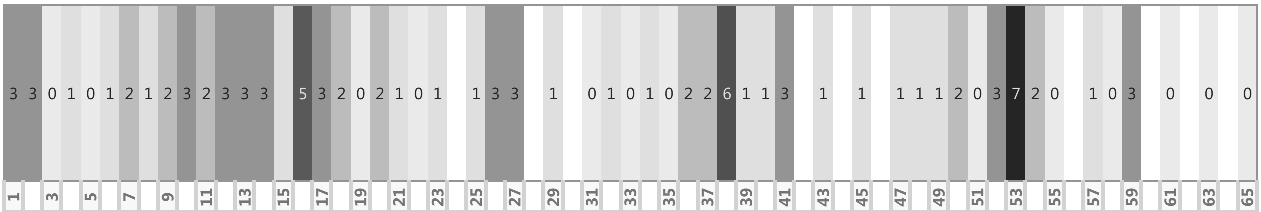

Fig. 9. Number of candidates for being dynamic hyperlinks found for 65 PFP commands.

Fig. 10. Second level FSM output as a schematic of the AC distribution in an aircraft.

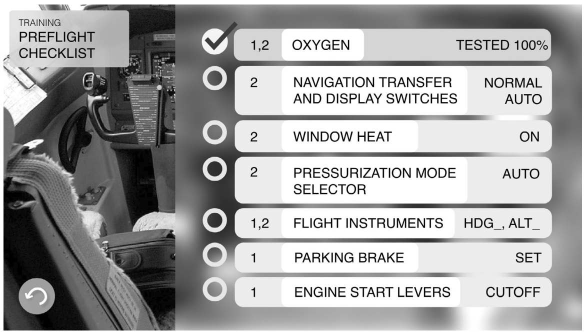

Fig. 11. Preflight check list displayed by AR goggles to the pilot trained in a new-technology flight simulator.

6. Conclusions

Nowadays, flight simulator devices are widely used in pilot skills training. When practicing flight procedures, a crew often faces a lack of easily accessible knowledge on various aspects of airplane construction, which makes the learning/practice process ineffective and prolonged, in particular during the first years of an employment. Not only names, but also the location of cockpit instruments as well as specific terminology used in the Flight Crew Operations Manual can slow down making progress in acquiring the competencies by the members of cockpit teams. The mere existence of informational pages including suitable explanations which can clarify the displayed command is no longer the only way for learning. Relevant pages should be recommended in a form of a smart list, to suggest novices the most important content and get them more engaged in a process. This means that the content of the list, including hyperlinks to relevant information, should vary depending on the command which is currently performed by a trainee (context based learning). Besides, the general information most often used by a large number of course attendees should be placed on top for fast reference.

Taking into account all these facts, we proposed the architecture of an event-based scenario player for flight simulators. Its conceptual model corresponds to the hierarchy of finite state machines. The states of the first-level machine are associated with subsequent tasks of the flight procedure (they form a context), while the second level machine enables the efficient handling of contextual information (in the form of websites, such as wiki pages or schematics of interesting systems of the airplane; see Fig. 10).

The functionality of the designed unit, as well as its convergence to a state when it can serve as a contextual learning aid for aviation adepts, has been confirmed by extensive simulations. Initially, we planned verification of the system in a Virtual Flight Laboratory by pilot training in flight simulators. However, these plans collapsed with the outbreak of the SARS-CoV-2 pandemic. With that regard, we were unable to conduct research using the data collected during the actual pilot training. Therefore we based our conclusions only on the results of the simulation, but this approach had also its advantages. We were able to follow each stage of the process described in the paper in detail. Thanks to the parametrization of our procedures, we could also track the impact of the individual variable on the result obtained. The starting point for simulations became the preflight procedure for a captain of Boeing 737 airplane (see Fig. 11) accompanied by the glossary of terms identified by the aviation expert.

In this way the system has been equipped with the procedural knowledge that is particularly helpful during training. We have also shown how the list of recommended, so called ‘dynamic’, hyperlinks was created for each page, and how the content of the list was changing when nesting the hyperlinks while performing particular steps of the flight checklist.

To select appropriately relevant dynamic hyperlinks for each page and arrange the list according to their decreasing popularity, we used the original method of determining the threshold for the frequency of visits to the page. The set of dynamic hyperlinks related to the given context was established on the basis of choices made by prior trainees. We separately examined pages that included a given number of static hyperlinks. It was found that at any point of aviation training it was possible to suggest a number of other hyperlinks, so called dynamic ones, presumably helpful for trainees. They referred to pages identified as most visited in a given context or to the pages indexed on top of all searches, when the required number of pages did not exceed the cut-off point. Thus we demonstrated that the system is able (by self-learning from actions of the trainees) to achieve a state of the contextual 'learn-as-you-go' aid.

The proposed scenario player system constitutes a basis for a real AR system supporting a new-technology flight simulator, which we believe will be utilized by actual trainees to speed up their learning. Collecting data during long periods (comprising a significant number of training sessions) and periodically updating the lists of dynamic hyperlinks available in the given context, is an important built-in mechanism of the system, which will allow to self-adapt to new learning preferences. Last but not least, the system is designed as a universal model for controlling AR goggles (not only MS HoloLens 2, and not solely for aviation training) making its applications much wider than new-technology flight simulators.

Acknowledgment

The authors would like to acknowledge that this paper has been written based on the results achieved within the WrightBroS project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement no. 822483. Also, this research work has been co-financed from Polish financial resources for science for the years 2019–2022 granted for implementation of the co-financed international project.

Disclaimer: The paper reflects only the authors' view and the Research Executive Agency (REA) is not responsible for any use that may be made of the information it contains.

References

Adrego da Rocha, A.M. (1999). Synthesis and simulation of reprogrammable control units from hierarchical specification, *PhD thesis*, University of Aveiro.

Barkalov, A., Titarenko, L. and Mielcarek, K. (2020). Improving characteristics of LUT-based Mealy FSMs, *International Journal of Applied Mathematics Computer Science* **30**(4): 745–759, DOI: 10.34768/amcs-2020-0055.

Berger, J.O. (1985). *Statistical Decision Theory and Bayesian Analysis*, Springer, New York, DOI: 10.1007/978-1-4757-4286-2.

Bernard, M. (2002). *Examining a Metric for Predicting the Accessibility of Information within Hypertext Structures*, PhD thesis, Wichita State University, Wichita.

Bower, M., Howe, C., McCredie, N., Robinson, A. and Grover, D. (2014). Augmented reality in education—Cases, places, and potentials, *Educational Media International* **51**(1): 1–15.

Brown, L. (2017). The next generation classroom: Transforming aviation training with augmented reality, *National Training Aircraft Symposium NTAS 2.0, Daytona Beach, USA*, `https://commons.erau.edu/ntas/2017/presentations/40/`.

Camilleri, M. (2018). *Travel Marketing, Tourism Economics and the Arline Product*, Springer Nature, Cham, Chapter 12, pp. 191–204, DOI: 10.1007/978-3-319-49849-2-12.

Caudell, T. and Mizell, D.W. (1992). Augmented reality: An application of heads-up display technology to manual manufacturing processes, *Proceedings of the 25th Hawaii International Conference on System Sciences, Kauai, USA*, Vol. 2, pp. 659–669

Cerqueira, C. and Kirner, C. (2012). Developing educational applications with a nonprogramming augmented reality authoring tool, *World Conference on Educational Multimedia, Hypermedia and Telecommunications (EDMEDIA), Denver, USA,* pp. 2816–2825.

Cox, J. (2020). *Training, Checking and Recency*, Safety Resources of the EASA Community, Section: Air Operations, European Union Aviation Safety Agency, Cologne, `https://www.easa.europa.eu/community/content/training-checking-and-recency`.

Cyran, K.A., Moczulski, W., Myszor, D., Paszkuta, M., Ruranski, A., Kalisch, M., Cyran, J., Adamczyk, M. and Timofiejczuk, A. (2018a). Immersive human-machine interface for controlling the operation of the Telerescuer robot, *International Journal of Advances in Computer Science and Its Applications* **8**(1): 322–326.

Cyran, K. A., Paszkuta, M., Myszor, D., Rohn, T., Drosik, T., Adamczyk, M. and Moczulski, W. (2018b). UAV-based anti-smog monitoring of the quality of exhausts from private chimneys in urban areas, *Towards a Circular Economy: 7th International Symposium and Environmental Exhibition, Vienna, Austria*.

Dhaliwal, A. (2019). *Augmented Reality for the Aviation Industry: Getting Ready for Take Off*, Atheer, Santa Clara, `https://content.atheerair.com/hubfs/eBooks/Aviation-eBook.pdf?t=1541468052547`.

Eschen, H., Kötter, T., Rodeck, R., Harnisch, M. and Schüppstuhl, T. (2018). Augmented and virtual reality for inspection and maintenance processes in the aviation industry, *Procedia Manufacturing* **19**: 156–163, DOI: 10.1016/j.promfg.2018.01.022.

FAA (2020). *Benefit-Cost Analysis,* Federal Aviation Administration, Washington, https://www.faa.gov/regulations_policies/policy_guidance/benefit_cost/.

Feiner, S., Macintyre, B. and Seligmann, D. (1993). Knowledge-based augmented reality, *Communications of the ACM* **36**(7): 53–62, DOI:10.1145/159544.159587.

Ferdania, D.F., Irawati, Garminia, H., Akhmaloka and Rachmansyah, K.A. (2021). Minimal state automata for detecting a β globin gene mutation, *International Journal of Applied Mathematics and Computer Science* **31**(2): 337–351, DOI: 10.34768/amcs-2021-0023.

Gagné, R.M. (1965). *The Conditions of Learning*, Holt, Rinehart and Winston, New York.

Giantamidis, G., Tripakis, S. and Basagianis, S. (2019). Learning Moore machines from input-output traces, *International Journal on Software Tools for Technology Transfer* **23**: 1–29, DOI: 10.1007/s10009-019-00544-0.

Goel, A. (2018). Augmented reality in aviation: Changing the face of the sector through training and simulated experience, *eLearning Industry*, https://elearningindustry.com/augmented-reality-in-aviation-changing-face-sector-training-simulated-experience.

Grzegorczyk, T., Śliwinski, R. and Kaczmarek, J. (2019). Attractiveness of augmented reality to consumers, *Technology Analysis and Strategic Management* **31**(11): 1257–1269(13), DOI: 10.1080/09537325.2019.1603368.

Haritos, T. and Macchiarella, N.D. (2005). A mobile application of augmented reality for aerospace maintenance training, *24th Digital Avionics Systems Conference, Washington, USA*, Vol. 1, pp. 5.B.3–5.1.

Hejase, M., Oguz, A., Kurt, A., Ozguner, U. and Redmill, K. (2016). A hierarchical hybrid state system based controller design approach for an autonomous UAS mission, *16th AIAA Aviation Technology, Integration, and Operations Conference, Washington, USA*, DOI: 10.2514/6.2016-3294.

Kearns, S.K., Mavin, T.J. and Hodge, S. (2020). *Engaging the Next Generation of Aviation Professionals*, Routledge, London.

Khan, T., Johnston, K. and Ophoff, J. (2019). The impact of an augmented reality application on learning motivation of students, *Advances in Human–Computer Interaction* **2019**: 1–14, Article ID: 7208494.

Ledermann, F. and Schmalstieg, D. (2005). April: A high-level framework for creating augmented reality presentations, *IEEE Virtual Reality, Bonn, Germany*, pp. 187–194, DOI: 10.1109/VR.2005.1492773.

Lee, K. (2012). Augmented reality in education and training, *TechTrends* **56**(2): 13–22, DOI: 10.1007/s11528-012-0559-3.

Markets and Markets (2019). Augmented and virtual reality (AR VR) market in aviation global forecast to 2025 by technology (AR, VR), function (training, operations), component (hardware, software), application (on-board, off-board), product, vertical, and region, Market Research Report AS 4553, *Markets and Markets*, https://www.marketsandmarkets.com/pdfdownloadNew.asp?id=230427667.

Moir, I. and Seabridge, A. (2008). *Aircraft Systems: Mechanical, Electrical, and Avionics Subsystems Integration,* 3rd Edn, Wiley, Chichester, DOI: 10.1002/9780470770931.

Myers, P.L., Starr, A.W. and Mullins, K. (2018). Flight simulator fidelity, training transfer, and the role of instructors in optimizing learning, *International Journal of Aviation, Aeronautics, and Aero-Space* **5**(1), Article 6.

Neumann, U. and Majoros, A. (1998). Cognitive, performance, and systems issues for augmented reality applications in manufacturing and maintenance, *IEEE 1998 Virtual Reality Annual International Symposium, Atlanta, USA*, pp. 4–11, DOI: 10.1109/VRAIS.1998.658416.

Olsson, M. (2016). *Behavior Trees for Decision-Making in Autonomous Driving*, Master thesis, KTH Royal Institute of Technology, Stockholm.

Page, R.L. (2004). Brief history of flight simulation, *Semantic Scholar*, Corpus ID: 211478463, DOI: 10.1.1.132.5428.

Parkinson, S.R., Hill, M.D., Sisson, N. and Viera, C. (1988). Effects of breadth, depth and number of responses on computer menu search performance, *International Journal of Man-Machine Studies* **28**(6): 683–692.

Parsons, D. (2020). Full flight simulators incorporate VR for next generation of pilots, *Aviation Today*, https://www.aviationtoday.com/2019/08/01/training-brain-mind/.

Plummer, D., Karamouzis, F., Alvarez, G., Hill, J., Sallam, R., Daigler, J., Hunter, R., Litan, A., Resnick, M., Prentice, B., Natis, Y. and Gaughan, D. (2019). *Gartner's Top Strategic Predictions for 2020 and Beyond: Technology Changes the Human Condition*, Gartner, Inc., Stamford, https://emtemp.gcom.cloud/ngw/globalassets/en/doc/documents/450595-top-strategic-predictions-for-2020-and-beyond.pdf.

Rabbath, C. (2013). A finite-state machine for collaborative airlift with a formation of unmanned air vehicles, *Journal of Intelligent & Robotic Systems* **70**: 233–253.

Rypulak, A. (2017). Using a virtual reality environment to teach practical skills of aviation personnel, *EduAkcja*, https://eduakcja.eu/files/pdf/143.pdf, (in Polish).

Safi, M., Chung, J. and Pradhan, P. (2019). Review of augmented reality in aerospace industry, *Aircraft Engineering and Aerospace Technology* **91**(9): 1187–1194, DOI: 10.1108/AEAT-09-2018-0241.

Schaffernak, H., Moesl, B., Vorraber, W. and Koglbauer, I.V. (2020). Potential augmented reality application areas for pilot education: An exploratory study, *Education Sciences* **10**(4): 86, DOI: 10.3390/educsci10040086.

Singh, V. and Singh, A. (2018). Learn-as-you-go: Feedback-driven result ranking and query refinement for interactive data exploration, *Procedia Computer Science* **125**: 550–559.

Sklyarov, V., da Rocha, A.A. and de Ferrari, A.B. (1998). *Synthesis of Reconfigurable Control Devices Based on Object-Oriented Specifications*, Springer, Boston, DOI: 10.1007/978-1-4757-4419-4_7.

Spagnolo, C., Sumsurooah, S., Hill, C. and Bozhko, S. (2018). Finite state machine control for aircraft electrical distribution system, *Journal of Engineering* **2018**(13): 506–511, DOI: 10.1049/joe.2018.0039.

Stańczyk, U., Cyran, K. and Pochopień, B. (2007). *Theory of Logic Circuits. Vol. 2: Circuit Design and Analysis*, Silesian University of Technology, Gliwice, (in Polish).

Turner, R., Hooda, S., Gersh, J. and Cancro, G. (2008). ExecSpec: Visually designing and operating a finite state machine-based spacecraft autonomy system *Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation for Space, Pasadena, USA*, DOI: 10.1.1.520.1445.

Valenta, V. (2018). Effects of airline industry growth on pilot training, *Magazine of Aviation Development* **6**(4): 52–56.

Velichko, M. (2020). How virtual and augmented reality are used in aviation training and other use cases, *Jasoren*, https://jasoren.com/how-virtual-and-augmented-reality-are-used-in-aviation-training-and-other-use-cases.

Wang, D., Yang, S., Wang, L. and Liu, W. (2016). Hardware-in-the-loop simulation for aircraft electric power system, *2016 International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles/International Transportation Electrification Conference (ESARS-ITEC), Toulouse, France*, pp. 1–5.

Yon, Z. (2015). Modeling and simulation of controllers of aircraft power supply system based on finite state machine, *Semantic Scholar: Engineering,* Corpus ID: 114484490.

Young, J.D. (2015). *Development of a Finite State Machine for a Small Unmanned Aircraft System Using Experimental Design*, Student graduate works, AFIT-ENS-MS-15-M-146, Wright-Patterson Air Force Base, Dayton, https://scholar.afit.edu/etd/136.

Zajac, W., Andrzejewski, G., Krzywicki, K. and Królikowski, T. (2019). Finite state machine based modelling of discrete control algorithm in lad diagram language with use of new generation engineering software, *Procedia Computer Science* **159**(2019): 2560–2569.

Zaphiris, P. (2000). Depth vs. breath in the arrangement of web links, *Proceedings of the Human Factors and Ergonomics Society Annual Meeting, San Diego, USA,* pp. 453–456.

Zazula, A., Myszor, D., Antemijczuk, O. and Cyran, K. (2013). Flight simulators—From electromechanical analogue computers to modern laboratory of flying, *Advances in Science and Technology Research Journal* **7**(17): 51–55, DOI: 10.5604/20804075.1036998.

Zhang, J., Sheng, Y., Hao, W., Wang, P.P., Tian, P., Miao, K. and Pickering, C.K. (2010). A context-aware framework supporting complex ubiquitous scenarios with augmented reality enabled, *5th International Conference on Pervasive Computing and Applications, Maribor, Slovenia*, pp. 69–74, DOI: 10.1109/ICPCA.2010.5704077.

Zhu, J., Ong, S. and Nee, A. (2015). A context-aware augmented reality assisted maintenance system, *International Journal of Computer Integrated Manufacturing* **28**(2): 213–225, DOI: 10.1080/0951192X.2013.874589.

Małgorzata Bach is an assistant professor in the Department of Applied Informatics at the Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, Gliwice, Poland. She has experience in the scope of relational and NoSQL databases, data warehouses as well as computer systems engineering. Her research interests are primarily focused on machine learning, data processing and analysis, and Bayesian inference. She has participated in several EU project as a researcher and teacher.

Aleksandra Werner is an assistant professor and an experienced lecturer in the Department of Applied Informatics at the Faculty of Automatic Control, Electronics and Computer Science of the Silesian University of Technology, Gliwice, Poland. She specializes in data processing and analysis as well as machine learning. At the center of her scientific interests are issues related to e-learning, the CMS, LMS, LCMS, computer architecture, cloud-computing, database federation systems, and machine learning.

Magda Mrozik is a PhD student at the Faculty of Transport and Aviation Engineering of the Department of Aviation Technology, Silesian University of Technology in Katowice, Poland. She is interested in satellite navigation, especially the GNSS, human capabilities and limitations, and operational procedures in aviation. Currently, she is conducting research on the use of the SBAS positioning method in an approach to airplane landing.

Krzysztof A. Cyran is the director of the Virtual Flight Laboratory at the Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology (SUT), Gliwice, Poland. He holds a PhD degree in computer science and a DSc degree (habilitation) in technical sciences (2012) from the SUT. His scientific interests include image recognition with the use of computer generated holograms applied as ring-wedge detectors as well as knowledge acquisition and processing in bioinformatics and evolutionary genetics. He is the author and a co-author of more than 100 publications, with a total number of citations exceeding 500. Currently, he is an associate professor in the Department of Graphics, Computer Vision and Digital Systems at the SUT. He coordinates the EU H2020 *WrightBroS* project involving industrial partners from Austria and Slovakia.

Received: 14 December 2020
Revised: 19 April 2021
Re-revised: 16 July 2021
Accepted: 10 August 2021