ISSN 2754-1118 (Online)
ISSN 2753-6599 (Print)

Functional Safety and Oracular Subsystems

An Observation on ISO/IEC TR 5469: Artificial Intelligence — Functional safety and AI systems

Peter Bernard Ladkin

Causalis Ingenieurgesellschaft mbH, Bielefeld, Germany

Abstract

"AI" subsystems are finding their way into safety-critical systems. Use of this term has come to mean contemporarily software systems based on machine-learning (ML) techniques. It is an open question how such subsystems may be verified and validated in safety-critical applications. A guidance document for functional safety in the presence of AI subsystems has recently been published by ISO and IEC, ISO/IEC TR 5469:2024. This paper considers critically the conception expounded in TR 5469 of how such subsystems are to be construed architecturally and behaviourally, through considering the example of adaptive control. Some concepts which may be useful in characterising AI subsystems are proposed.

1 Introduction

1.1 AI Software and Subsystems

The area of computer science known as Artificial Intelligence has been around since John McCarthy coined the term in 1955[1]. The original idea was to mimic human capabilities computationally. Much of the early success was in so-called symbolic AI, with attempts to emulate logical reasoning by means of automated logic engines, such as the 1959 General Problem Solver of Newell, Shaw, and Simon (Newell et al 1959). In the 1960's, there were attempts to mimic the kinds of processing supposedly going on in human brains, with the 1969 book *Perceptrons* by Marvin Minsky and Seymour Papert an early example (Minsky and Papert 1969). Symbolic AI made considerable progress through the 1980's in areas such as automated reasoning, task planning, common-sense physics, symbolic machine learning, and expert systems. However, interest was increasing in statistical methods such as Dempster-Shafer Theory (Dempster–Shafer theory n.d.) and Judea Pearl's Bayesian Networks (Pearl 1988), as well as the successor to perceptrons, called artificial neural networks (ANN), with the general approach becoming known as connectionism (Rumelhart and McClelland 1986).

[1] The term became more generally known through the 1956 *Dartmouth Summer Research Project on Artificial Intelligence*, organised by McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon.

Techniques of machine-learning, still addressed through the 1980's by symbolic methods, have made significant progress in the last quarter century through using the paradigm of ANN (I shall subsequently drop the "A") applied to increasingly large training data sets. Significant advances have been made in computer vision, pattern recognition, image analysis, and computational linguistics, as well as other fields, using the NN paradigm and large data sets. Nowadays this paradigm has more or less taken over the domain of machine learning (ML). In the *Volksmund*, "AI" has come nowadays to mean ML/NN techniques [2].

Software in critical systems based on symbolic AI techniques is mostly amenable to standard software verification and validation (V&V) techniques, whereas software based on ML is not. A trained piece of ML software is more often considered to be a "black box", and techniques such as proposed in the functional safety standard IEC 61508 (IEC 61508:2010) do not handle "black box" software whose internal structure and operation is not amenable to analysis. Statistical techniques could theoretically be used, but for the evaluation of safety-related software such techniques are controversial at time of writing. Nevertheless, the capabilities of, for example, automated vision systems entail that they are used in applications such as autonomous road vehicles, which by their nature are safety-critical. The challenge is how to encapsulate "black box" NN software in a safety-critical system, as well as how to increase our ability to derive meaningful V&V results from the internal structure of such software and its training history.

1.2 Early Success with DLNNs in Control and Vision Systems

For a number of years, systems claiming to be "artificially intelligent" (AI systems for short) have been used in safety-related applications. For example, such systems have been used in adaptive control, in which traditionally-designed control systems are enhanced by feedback from deep-learning neural networks (DLNN), which received the same sensor inputs as the traditional control algorithm and whose output is used to modify the control commands to the actuators.

Some of the earliest and most successful experiments with adaptive control were conducted in the US by NASA from the 1990's on. In the wake of the accident to United Airlines Flight 232 at Sioux City on July 19, 1989, in which the DC-10 aircraft lost the ability to actuate any of its aerodynamic control, and reverted to control purely through differential thrust from the engines, there followed a successful NASA project to control the flight of a large transport aircraft, an MD-11 (the successor aircraft to the DC-10), using differential engine thrust only (NASA 2002). The idea here was that, instead of the pilots learning how directly to use engine thrust commands to control flight, as the Sioux City accident crew did, the crew of the adaptive-control MD-11 gave the usual aerodynamic-control inputs for pitch, bank and yaw, but these were translated into differential-thrust commands with the "same" effect by an interposed DLNN. The DLNN was trained statically, but also engaged in dynamic "learning" (further adaptation) during flight, hence the term "adaptive control".

Figure 1, overleaf, shows the Propulsion Controlled Aircraft (PCA) MD-11 during trials.

[2] More recently, since the end of 2022 with the release of ChatGPT, it seems to have even more narrowly come to mean Large Language Models (LLMs). The AI subsystems considered in this paper are not LLMs.

Figure 1 ~ PCA MD-11 landing at Dryden under Propulsion Control, 1995-08-29

There followed a series of flight experiments in the early 2000's, with a modified military F-15 interceptor aircraft (known as a NASA NF-15), in which the adaptive control had two goals. First, that "normal" control input could be exercised by the pilot even when some aerodynamic control surfaces were no longer available (for example, due to battle damage). Second, to enhance certain manoeuvres beyond that which the pilot could achieve using standard control. The "adaptive" system interpreted pilot's intent and using a different combination of control surfaces and propulsion (NASA 2014, paragraph *Intelligent Flight Control System*).

Figure 2 shows the NF-15B in flight during trials of the Intelligent Flight Control System.

Figure 2 ~ The NASA NF-15B on an Intelligent Flight Control System Mission

Flight is in some sense an easier control environment than the ground, because there is air everywhere you look (and fly). Whereas using wheels on a surface there are routinely obstacles; not only hard objects barring the way, but different surfaces affecting traction and steering, not to speak of other mobile objects.

These adaptively-controlled aircraft were not "drones". They had human pilots and, although the flights took place over a sparsely-populated (or non-populated) area of the Californian Mojave desert, the range of possible grounding locations in the event of an in-flight upset is of the order of tens of kilometres from the (on-ground) location of the point of upset. The V&V is non-trivial: Nguyen and Jacklin (2010, Section 4.2) state: *"[t]he current approach is to verify a neural net adaptive flight control over an exhaustive state space using the Monte Carlo simulation method*".

In contrast, adaptive control in road vehicles can be pursued in environments from which humans are excluded, and therefore, with appropriate boundary controls, upsets can be contained without engendering harm. The first automated-road-vehicle-in-actual-environment experiments commenced with the DARPA Grand Challenge off-road competition in 2004. No vehicle finished. In 2005 (Figure 3), five vehicles finished the 212 km off-road course (DARPA Grand Challenge (2005) n.d.).

Figure 3 ~ On the Off-road — the DARPA Grand Challenge 2005

Figure 4 shows the award ceremony for the 2005 winner.

Figure 4 ~ The 2005 Winner — Stanley, from the Stanford AI Laboratory

In 2007, the DARPA Urban Challenge put competitors into a simulated urban traffic environment in which the moving obstacles were other competitors' vehicles, see Figure 5 overleaf. The "course" was 96km long and had to be completed in 6 hours. 6 teams finished.

Figure 5 ~ DARPA Urban Challenge 2007

"Assistance systems" built on such experience have now been incorporated into road vehicles which operate on public roads. The Tesla Autopilot has been involved in a number of road accidents, some of them fatal. Autopilot operation is intended to be actively monitored by the human driver, but this has not always happened. (The weaknesses of supervisory control, as this is called, have been well-studied for decades.) The US NHTSA[3] has investigated, and is investigating, a number of Tesla accidents. At least two fatal Tesla accidents happened when the car was under Autopilot control, failed to recognise an obstacle (a truck crossing the road at an intersection; respectively a metal guardrail) and the driver did not react. News reports said in March 2021 that the agency was currently reviewing 23 Tesla crashes (Shepardson 2021). We understand that the majority of these investigations were requested by the automaker Tesla itself.

On 2018-03-18, a woman walking her bicycle across a road in Tempe, Arizona, was hit and killed by an experimental automated road vehicle in the fleet of Uber. An evaluation of system behaviour up to and including the collision showed that she and her bicycle, while sensed, were interpreted as an obstacle by the system, but the classification was variable, rapidly changing and resetting in the five seconds before collision (Harris 2019). There was considerable technical discussion at the time of the design of the sensing system. Further, an autonomous braking system on the vehicle installed by the manufacturer, Volvo, had been disabled in favour of Uber's own system, which did not brake until a fraction of a second before collision. The US NTSB found that Uber's 40 experimental vehicles had been involved at this point in 37 "incidents", many of which were collisions, in the year and a half between September 2016 and March 2018.

In summary, AI subsystems are already in use in safety-related systems, in controlled, "test" circumstances, even when performing on public roads, and have not always performed perfectly. Accordingly, standardisation bodies are attempting to formulate general principles for the incorporation of AI subsystems into safety-related systems. ISO/IEC TR 5469 is one such document (ISO/IEC TR 5469:2024).

[3] NHTSA, The National Highway Traffic Safety Administration is an agency of the U.S. federal government, part of the Department of Transportation

2 General Observations on Standards, and What ISO/IEC TR 5469 Tries to Do

Standards documents are hierarchically ordered. They contain clauses, which are like individual sections treating one major subtopic, which themselves consist of hierarchically-numbered subclauses. They also contain a Bibliography, as well as Annexes, which amplify on matters contained in the main text. Clauses, and subclauses, may be normative or informative. A normative subclause contains a requirement which must be fulfilled by anyone claiming to follow the standard; they are nominally mandatory. An informative subclause contains guidance, but nothing nominally mandatory. The TR 5469 document (as I shall call it for short) is a *Technical Report*, that is, it is purely informative and there is nothing normative about it. However, courts and other organisations do take IEC standards documents, whether informative or normative, to provide guidance as to how the electrotechnical profession views the state of its practice, as to how "things should be". For outside organisations, then, the intra-standards distinction between informative and normative is not as significant. However, the writers of standards documents do typically distinguish between how (they think) things *must* be, and how they *could* be conformant with the state of the *praxis*.

The authors of TR 5469 appear to be conceptualising how an AI subsystem can be embedded in a safety-related system, and what assessments are required in order to determine if it is to perform safely. The question arises whether this conception applies generally.

3 Architectures for Feedback Control

3.1 Control Systems and Systems with AI Subsystems

Franklin et al (1994) state that:

> *Control is the process of causing a system variable to conform to some desired value, called a reference value.*

A control-block architecture for a traditional automobile cruise control is shown in Figure 6, which is derived from Franklin et al. (1994, Figure 1.3).

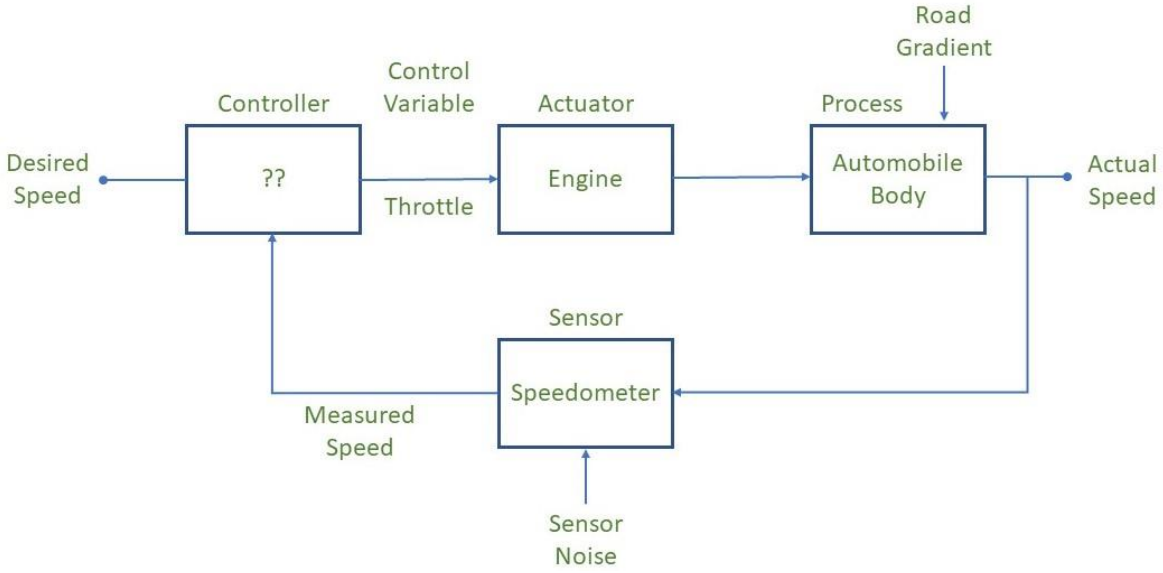

Figure 6 ~ Control Block Diagram for a Cruise Control System

This shows feedback control with an active component, a controller, which is further specified in a Function Block Diagram (FBD) in Figure 7 (derived from Franklin et al. (1994, Figure 1.4)), which specifies the mathematical relationships used by the controller to produce its output.

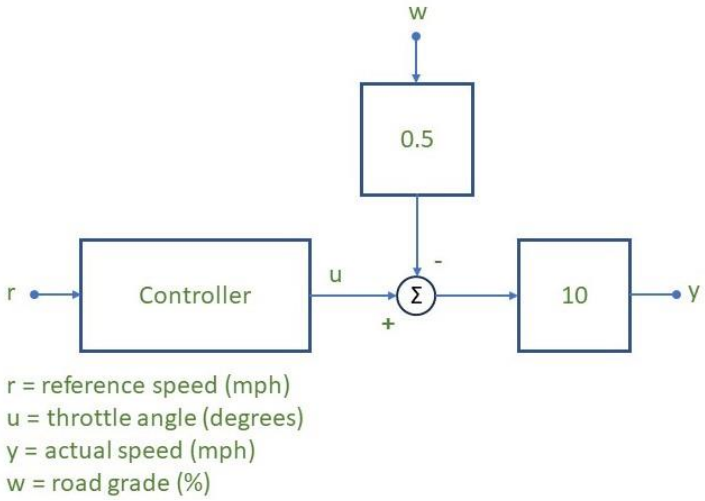

Figure 7 ~ A Function Block Diagram of a Cruise Controller

FBDs for specific examples of open-loop control and closed-loop control are shown in Figures 8 & 9, respectively (derived from Franklin et al. (1994, Figures 1.5 & 1.6)). The number *1/10* in the *Controlle*r block of Figure 8, for example, means that the input value designated by *r* is multiplied by *1/10* to become the value designated by *u*, i.e. $u = r/10$. On the other input branch, the input value designated by *w* has been multiplied by *0.5* and then turned negative: that is, $-0.5w$. These two quantities are then summed (the Σ) and the result is then multiplied by *10* to become *y*. This means in this simple case $y = 10(r/10 - 0.5w) = r - 5w$ (Franklin et al 1994, Chapter 1).

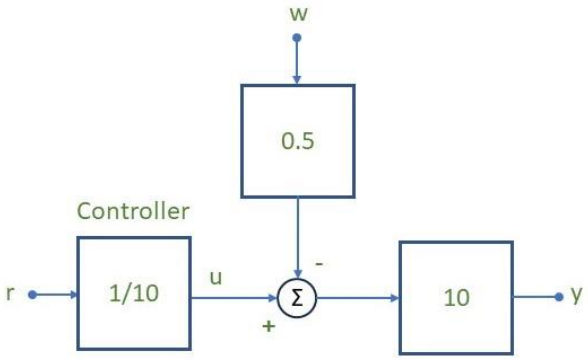

Figure 8 ~ Open-Loop Cruise Control FBD

The value of y in the closed-loop control example in Figure 9 is a little trickier. Input r is multiplied by 100 to form u*: $u = 100.r$*. This is then summed with $-0.5w$ as before, and the result multiplied by 10 as before. However, the result $10(100.r - 0.5w)$ is then "fed back" to be summed with r at the leftmost Σ, and the result $r - 10(100.r - 0.5w))$ then goes through the diagram again. This can be put into simultaneous equations and solved analytically (Franklin et al 1994 Chapter 1), but a good way to think of it is that there is an equilibrium value for fixed r and w, which is "distorted" when one of them changes slightly, and adjustment leads to a new value of y.

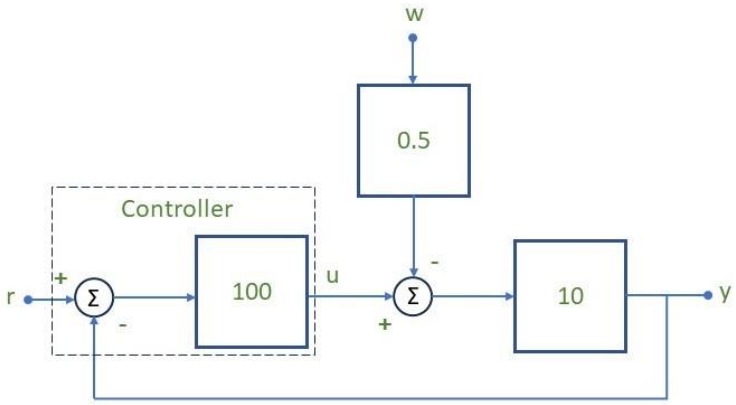

Figure 9 ~ Closed-Loop Cruise Control FBD

A modified Control Block Diagram, incorporating elements of an FBD, for the NASA NF-15B control system architecture is given in Figure 10, after Smith et al. (2010, Figure 2). A more detailed FBD for the combination of adaptive neural network and inversion controller is given in Figure 11, after Nguyen and Jacklin (2010, Figure 1). (These are included here for expository completeness; the rest of the text does not use any specific mathematically-annotated FBDs.).

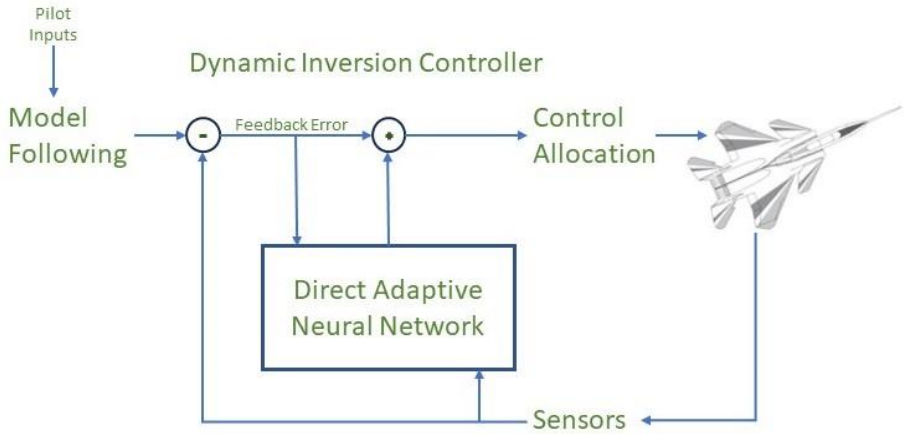

Figure 10 ~ IFCS System CBD+Abstract FBD

I shall show that these CBD-FBD architectures are not accommodated in the current architectural conception of TR 5469. It follows that the considerations of TR 5469 do not apply *per se* to these adaptive-control architectures.

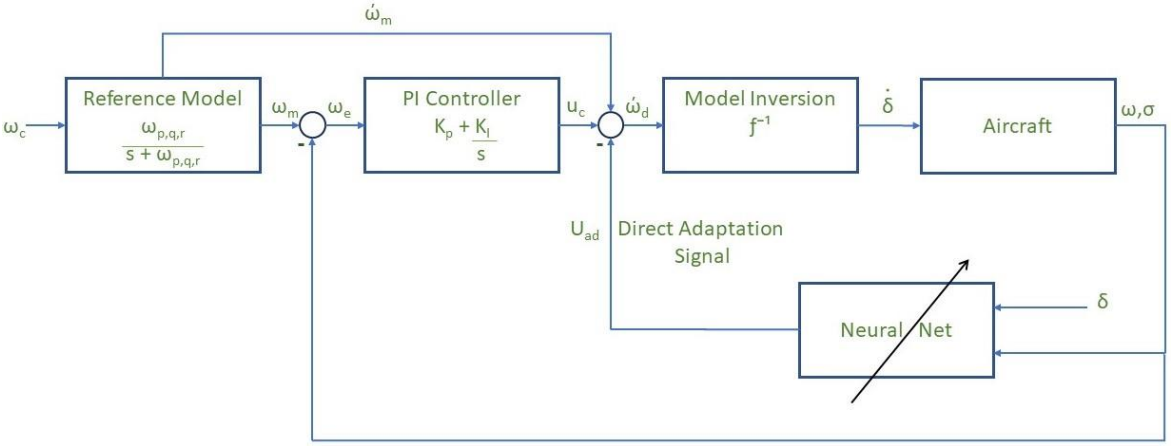

Figure 11 ~ The FBD for the NF-15B Adaptive Control

3.2 A Control Block Diagram for Adaptive Feedback Control

A controller has sensors which determine the state and behaviour of the controlled object (here, I shall call it a vehicle) and the state and behaviour of the environment. The behaviour (that is, change of state) of both entities can be captured in *history variables*, so, for example, from elementary differential calculus we know that rates may be captured by first derivatives, accelerations by second derivatives. The first and second derivatives can be included in the state as history variables, and usually are. They are derived by calculation from discrete location data over the previous short time period. The specification of the controller determines what quantities are required for control, so the history variables which are needed are determined by the specification of the controller input. Thus can "state and behaviour" be replaced by "state" alone. However, I mention both here.

Figure 12 shows the inputs to the sensors from the state+behaviour of the controlled object, the EUC[4] in the terminology of IEC 61508, here denoted as the vehicle, and from the state+behaviour of the environment.

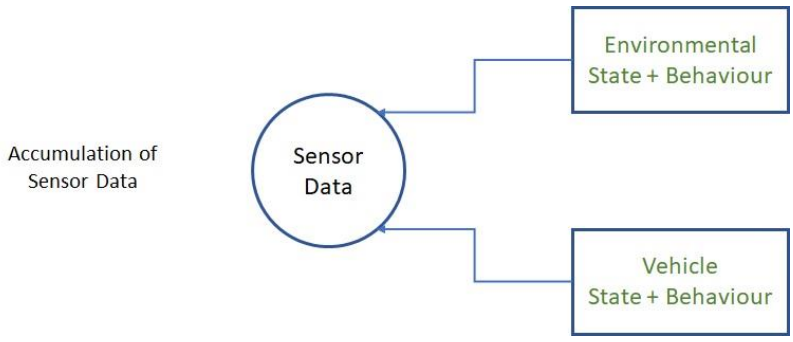

Figure 12 ~ Sensors Receive Input from Vehicle + Environment

The sensor data then needs to be incorporated into a model of the physical environment + vehicle, suitable for making control decisions, as in Figure 13.

[4] Equipment Under Control

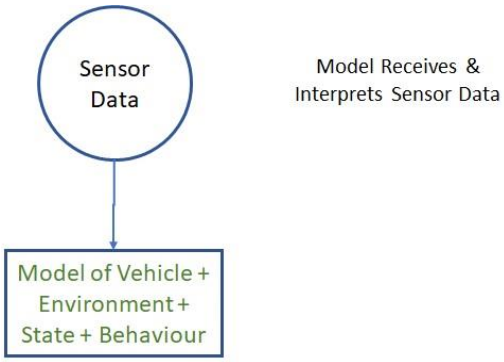

Figure 13 ~Sensor Data is Passed to a Model Component

The adaptive controller receives information from the model on the state+behaviour of the environment and the vehicle, and determines what actuator commands to output to the actuation subsystems, as well as possibly revising its own configuration, if it is a dynamically-trained NN (or not, if it is statically trained). This is shown in Figure 14.

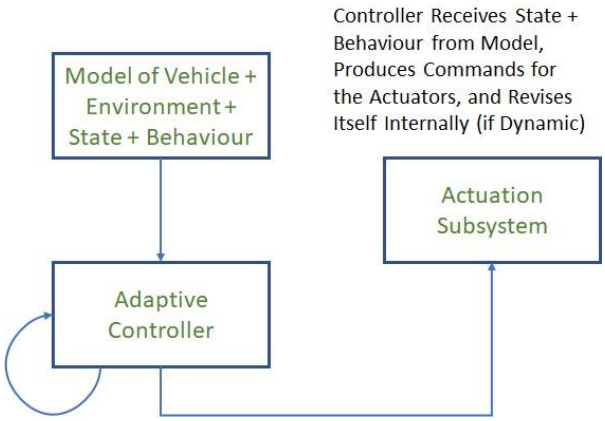

Figure 14 ~ Controller Receives Input from Model, Determines Output for Actuation

In Figure 15, the actuation operation is shown.

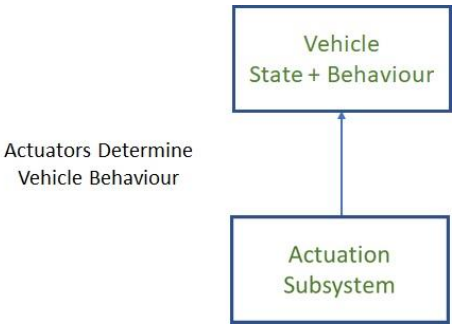

Figure 15 ~ Actuation

Finally, the entire feedback control loop looks as follows in Figure 16.

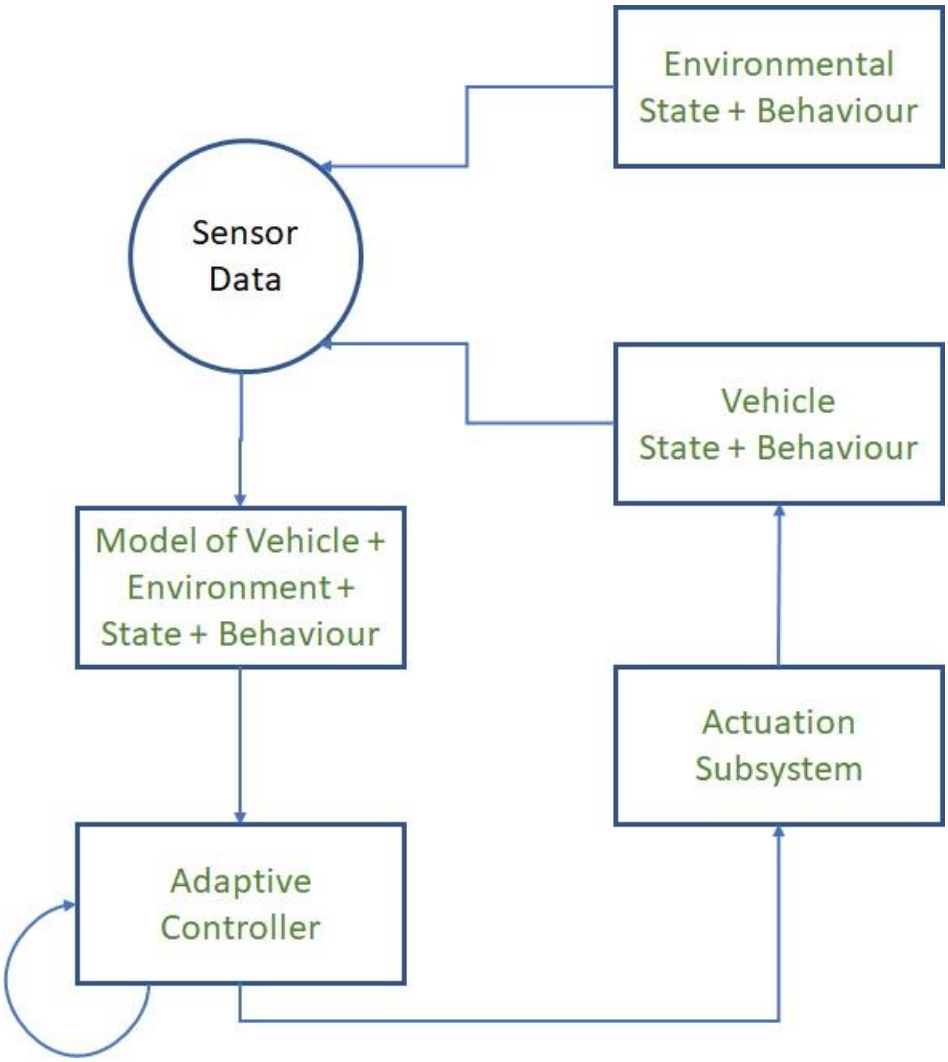

Figure 16 ~ The Feedback Control Block Diagram

4 System and Software Architecture in Standards Documents: A Comparison

4.1 System Architecture According to TR 5469

TR 5469 explains its system architectural concept in words (ISO/IEC TR 5469:2024): *" ……. the capability of AI is often achieved by the combination of an algorithm and a model. The model typically represents information that achieves the application's purpose, (e.g. knowledge about how various inputs are to be distinguished and recognized), while algorithms infer information from a model and inputs, (e.g. to make a prediction).*"

The model is a kind of "knowledge engine", in other words, and the algorithm a query facility for the knowledge engine. Another way of expressing this is to say that the model is an oracle for the domain about which it represents knowledge — an oracular subsystem. Figure 17 gives a control block diagram for this "*achievement*" of the "*capability of [AI]*".

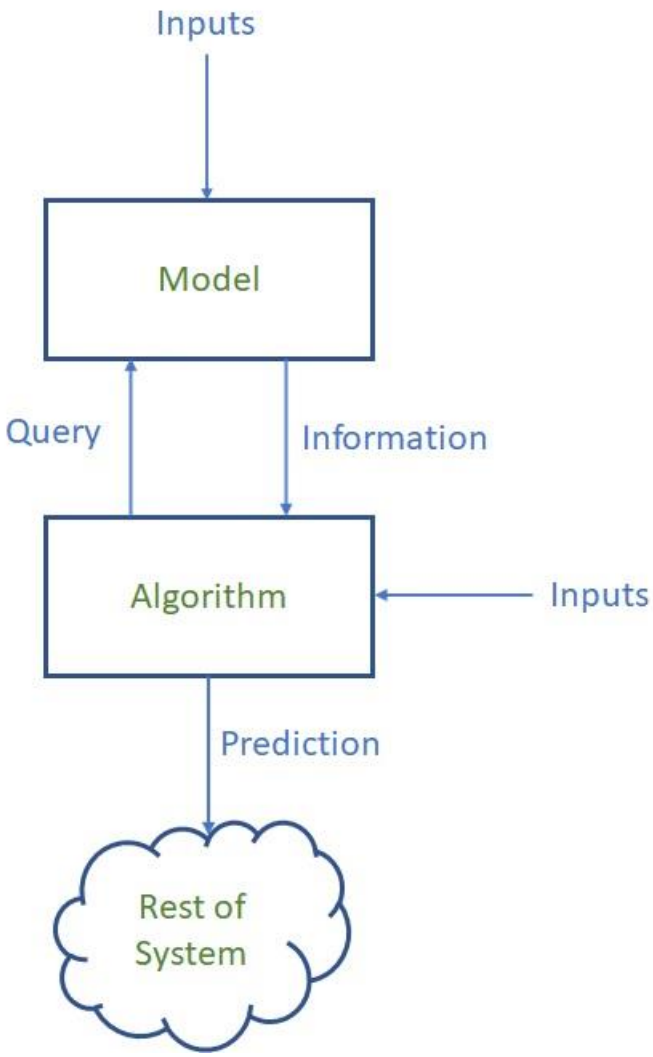

Figure 17 ~ Control Block Diagram of the "Capability of AI" as Model+Algorithm

The model receives inputs from somewhere, and uses these inputs to construct its representation of its knowledge. The algorithm uses knowledge from the model (the transaction is represented by a query from the algorithm and information passed from model to algorithm). It also may have (according to the text above) further "*inputs*". Its output, according to the text, is a "*prediction*". This is precisely what is represented in the control block diagram in Figure 17. I shall continue to use the example of a *prediction* output in what follows.

Let us compare the CBD in Figure 17 with that in Figure 16. I write the item labels in italics. There is a *Model* in both diagrams. In the feedback-control CBD in Figure 16, the *Model* represents knowledge about the actual state (+behaviour) of vehicle + environment. This is what would be in the *Model* also in Figure 17. The *Inputs* in Figure 17 would be the *Environmental State+Behaviour* and the *Vehicle State+Behaviour* from Figure 16. These parts of the CBDs cohere with each other. There is a question with feedback control as to what in Figure 16 might play the role of the block *Algorithm in* Figure 17. There are three main differences between the CDBs, as follows:

- In Figure 16, there is no *Query* from the *Adaptive Controller* to the *Model*. Any communication from *Adaptive Controller* to *Model* proceeds through *Actuation*, *Vehicle State+Behaviour*, and *Sensor Data*; in other words through the feedback loop. The

information that the *Adaptive Controller* gets from the *Model* in feedback control is determined in advance.
- The output from an *Adaptive Controller* is not *Prediction*, as in Figure 17, but is a set of actuation commands to the *Actuation Subsystem*.
- The *Adaptive Controller* does not receive any *Inputs* other than those which come from the *Model*, nor do any other elements on the path to *Vehicle State+Behaviour* (namely the *Actuation Subsystem*).

In respect of the first and third points, Figure 16 with *Adaptive Controller* regarded as the *Algorithm* could be regarded as a special case of the architecture in Figure 17, namely an instance without *Query* from *Algorithm* to *Model* and without *Inputs* to *Algorithm*. However, that the output from *Algorithm* is to be a *Prediction* in Figure 17, but is *actuation commands* in Figure 16, is an irreconcilable difference. ("*Prediction*" is, according to the text from TR 5469 above, only one example of an algorithm output. I have not explored other examples, since for my purpose here it suffices to show one irreconcilable difference between the architectures.)

Furthermore, the *Model* in Figure 17 is supposed to contain the entire AI component, and the *Algorithm* is a "traditional" software object. According to TR 5469 Section 7.2:

> *"Usually, algorithms that interact with the models contain only a limited amount of knowledge or implications about the application's goals. This is quite similar to the role of foundational software libraries or programming environments (compilers, etc.) in non-AI software. That is, the algorithm itself does not play a functional safety role, but its correctness is critically important for functional safety to be achieved. In this way, the integrity of algorithms in AI technology can often be handled with existing principles of functional safety as specified in the IEC 61508 series ....., similar to that of non-AI software components."*

There are four points of difference here between the *Adaptive Controller* in Figure 16 and this description of *Algorithm*.

- Most importantly, the Adaptive Controller does not "*contain only a limited amount of knowledge or implications about the application's goals ..... quite similar to the role of foundational software libraries or programming environments*". It embodies all "*knowledge and implications about the application's goals*" in that it is the controller.
- Second, it contains AI itself, namely a DLNN exercising the adaptive control. It is not at all "*similar*" to a "*software librar[y]*" or a "*programming environment[ ]*."
- Far from "*not play[ing] a functional safety role*" the Adaptive Controller is one of the main actors in terms of the safety of the system. If it gives the wrong actuation commands, the vehicle's behaviour will change and may cause harm. Any safety functions to be introduced to avoid or mitigate the risk must be introduced between *Adaptive Controller* and *Actuation Subsystem*, and presumably receive inputs from the *Model*, maybe under *Query* requests.
- It is implausible that its integrity "*can often be handled with existing principles of functional safety as specified in the IEC 61508 series ....., similar to that of non-AI software components.*" This is exactly why TR 5469 is being written — because the safety integrity of AI-technology based components is not seen to be encompassed by IEC 61508 as it is.

4.2 Other Applications of the "Model-Algorithm" Architecture

We have just seen that the architecture of Figure 17 is inappropriate for describing adaptive control using DLNNs. However, it is appropriate for describing various other "knowledge" architectures. Theorem provers, although used in AI, are largely associated with attempts at system and program verification, having been largely supported by the US Department of Defense. A history up to about twenty years ago may be found in (MacKenzie 2001). Decision procedures are used for various decidable mathematical theories in most theorem provers nowadays. Various mathematical theories are decidable, for example Presburger arithmetic and Skolem arithmetic, and the theories of finite fields and of real-closed fields, but some are not, for example group theory (although Abelian group theory is decidable). The question of how to incorporate decision procedures into general theorem-proving mechanisms was addressed in two definitive papers by Nelson and Oppen (1979) (1980). Shostak also devised a method of incorporating mathematical domain-specific theories into a general theorem prover in (Shostak 1984). The "Shostak prover" was the inference engine in SRI International's EHDM[5] specification and verification system.

A decision procedure, or a domain-specific mathematical theory, can fulfil the role of *Model* in Figure 17, and the general inference mechanism the role of *Algorithm*. Thus the CBD in Figure 17 describes also the architecture of such cooperating theorem provers, which is nowadays a popular architecture for theorem provers.

4.3 The Definitions of "Model" and "Algorithm"

Despite their central use in the above description of an "AI architecture" in TR 5469, the terms "algorithm" and "model" are not defined in the Terms and Definitions section of the TR.

There is a definition of "model" in the ISO/IEC standard defining AI concepts and terms (IEC 22989:2022, subclause 3.1.23).

> **model**
>
> physical, mathematical or otherwise logical representation of a system, entity, phenomenon, process or data
>
> [SOURCE: ISO/IEC 22989:2022, 3.1.23]

There is no definition of "algorithm" *per se* in either the draft TR or IEC 22989; rather, there is a definition of "machine learning algorithm" (IEC 22989:2022, subclause 3.3.6).

> **machine learning algorithm**
>
> algorithm to determine *parameters* (3.3.8) of a *machine learning model* (3.3.7) from data according to given criteria
>
> EXAMPLE: Consider solving a univariate linear function $y = \theta 0 + \theta 1x$ where y is an output or result, x is an input, $\theta 0$ is an intercept (the value of y where x=0) and $\theta 1$ is a weight. In *machine learning* (3.3.5) the process of determining the intercept and weights for a linear function is known as linear regression.
>
> [SOURCE: ISO/IEC 22989:2022, 3.3.6]

[5] Enhanced Hierarchical Development Methodology

There are two definitions of "algorithm" (unqualified) in the International Electrotechnical Vocabulary (IEV n.d.).

> **algorithm**
>
> completely determined finite sequence of instructions by which the values of the output variables can be calculated from the values of the input variables
>
> Note 1 to entry: The behaviour of a system with discrete-value input and output variables (for example a switching system) may be described completely by an algorithm. For a system with continuous-value and continuous-time input and output variables the algorithm is given by or derived from the mathematical relationship between the input and output variables.
>
> Note 2 to entry: This entry was numbered 351-21-37 in IEC 60050-351:2006.
>
> [SOURCE: IEV 351-42-27][6]
>
> **algorithm**
>
> finite set of well-defined rules for the solution of a problem in a finite number of steps
>
> [SOURCE: IEV 714-21-02][7]

Besides these definitions, there is yet another definition of "algorithm" in IEC standards, which is a variant of IEV 714-21-02:

> **algorithm**
>
> finite set of well-defined rules for the solution of a problem in a finite number of operations
>
> [SOURCE: IEC 61499-1:2012 *Function blocks — Part 1: Architecture*; also IEC 61804-2:2018 *Function blocks (FB) for process control and electronic device description language (EDDL) — Part 2: Specification of FB concep*t (SC65E).

The two IEV definitions are homonyms and nominally require resolution (the IEV is supposed to be homonym-free)[8]. It is clear from these definitions that a "machine learning algorithm" is different from an "algorithm" as defined (in either version) in the IEV, or as defined in Function Block standards. To which definition TR 5469 is intended to adhere is not specified in the document.

5 Models, Precision and Vagueness

5.1 A Set of Concepts for Properties of "Models"

We have seen that TR 5469 says that a **model** "*represents information that achieves the application's purpose*". It follows that a model is a knowledge-representation artifact, an

[6] Section 351 of the IEV is Control technology.

[7] Section 721 of the IEV is Digital technology — Fundamental concepts

[8] Annex SK of ISO/IEC Directives (2021). gives Rules for terminology work. Section SK 2.3, p104, contains the requirement *The rule "one concept – one definition" shall be applied*. Terms with multiple definitions are to be distinguished by *adding a specific use* to the term (p104), denoted in angle-brackets "<...>".

oracle for a knowledge domain. The terms "*knowledge representation*" and "*knowledge representation and reasoning*" (KR&R) are common in AI. The Wikipedia page for KR&R, however, only adduces symbolic-AI approaches to KR&R. It does not include NNs or ML approaches (Knowledge representation and reasoning n.d.). This is appropriately traditional in the AI subfield.

When computing, there is typically a collection of subject-matter domains, within which certain kinds of computations are performed. I have noted that one design of theorem prover, for example the Shostak prover or the Nelson-Oppen approach, uses a general logic engine and a collection of decision procedures for formal mathematical domains (such as "Presburger arithmetic" for arithmetic). Obviously, a decision procedure is a means of "representing knowledge" about the domain over which it decides assertions; arithmetic, say, in the case of Presburger arithmetic, or Skolem arithmetic, or quantifier elimination for finite fields. Such a procedure can be regarded as an oracle for its domain.

It seems worthwhile to define "domain"[9]. A **domain** is a collection of types of mathematical, or logical objects, or such which are representations of real-world objects, with explicit properties, relations and operations, which are nominally closed with respect to those operations. Associated with the domain is a **domain language**, which has terms for the types, the properties, relations and operations and can express assertions concerning them (for example, using a many-sorted logical language; the sorts correspond to the types of object). Let me call *answering questions and deciding (the truth of) assertions* in a domain **decision making** or **making decisions** in the domain. A model is then a *computational means of decision making in the domain language*.

A domain is **precise** in so far as questions and assertions in the domain language have determinate real-world answers, respectively truth-values, and **vague** in so far as it is not precise. A model is **precise** or **vague** in so far as its domain is precise or vague. A model is **accurate** in so far as its answers to questions and the truth-values of its assertions cohere with the real-world answers. A model is t**raceable** in so far as the computations which give the answers to questions and determine the truth-values of assertions produce as output (upon request) a chain of reasoning which determines that the answers, respectively the truth-values, are correct and/or justified.

A model may be vague and nevertheless accurate, for example a DLNN decision procedure that determines whether Magnetic Resonance Tomography scans are "normal" or anomalous. There are two artifacts in play in such an example. The first translates pixels into the model domain. The domain consists of the parts of the bodily anatomy, as well as anomalous or parasitical objects and properties. The accuracy of the model likewise consists in two phenomena. First, the veridicality of translation from pixels to model domain ("image recognition"). Second, the designation of areas of concern (deviations from "normal"). Model accuracy is determined by experience, and coherence with expert judgements and outcomes.

A model may be vague and nevertheless traceable, in that, for every decision, it can output intermediate or auxiliary decisions which provide a humanly-recognisable rationale for its decision. One can imagine the model for Magnetic Resonance Tomography scans outputting on request the anatomical configurations it has identified, along with the areas of concern, and the characteristics of those areas of concern. A human doctor can assess the anatomical configuration for accuracy in a more or less precise manner. The

[9] The definitions given in this and the following paragraph of *domain*, *domain language*, *decision making*, *making decisions*, *model*, *precise* (for domain and model), *vague* (for domain and model), *accurate*, and *traceable* are mine.

identification of areas of concern is more-vague. One can imagine that the usefulness of such an identification lies in simply directing the attention of a human to those areas, rather than any more complex calculation such as automatically proposing a conclusion from within the model.

Nelson-Oppen-style decision procedures form precise models, as do Bayesian networks (and other statistical methods). Nelson-Oppen-style decision procedures are accurate. However, Bayesian networks are not necessarily accurate.

DLNNs and other ML techniques are used to build vague models. They may be accurate, or not. They may be traceable, or not.

Bayesian networks are partially traceable. The weights on edges constitute assertions as to how strongly the phenomena named in the node labels are correlated. A trace consists of the conjunction of these correlations for all adjacent node pairs.

5.2 Concerning Algorithms

A model M may be a subsystem or subcomponent of a computation-based system S. Typically, not all the representational features of M will be used in S, but only a subset. It seems to me from the text above that the authors of TR 5469 want an *algorithm* in the AI sense to be query engine for M in S. That is consistent with the current definitions of *algorithm* in the IEC glossary.

5.3 Other Desirable Properties

If you train a DLNN on data, you intuitively want the DLNN to exhibit the same decision making no matter in what order you give it the data in training. Call the DLNN **commutatively invariant** if so. Ross Anderson's group has discovered public-facing DLNNs that are not commutatively invariant (Schumailov et al. 2021). D'Amour et al. (2020) have suggested that some DLNNs are underspecified. I suspect that non-commutative-invariance is a subcategory of underspecification.

5.4 Partial Conclusion on Model Properties

It seems that precision/vagueness, accuracy and traceability capture the qualities of a model which are relevant for assessment for functional safety properties. The various properties of specific DLNNs arising out of the work of both Shumailov et al. and D'Amour et al. seem to be highly relevant to determining accuracy and traceability.

5.5 Observations on Safety

To determine safety properties of a system, it is necessary to perform system-level and subsystem-level hazard analyses. These are required as part of “Hazard and Risk Analysis” (H&RA) in subclause 7.4 of IEC 61508-1. A causal analysis (the left part of the “Bow Tie”) is required to find out how subsystems can contribute to causal factors of a hazard. If we are talking an adaptive controller, then it trivially seems as if we require it to be accurate on all inputs in its operational envelope. If we are talking a model of another sort, along with an algorithm which queries it, then the answers to queries (may) form part of a causal chain leading to a hazard. So it is necessary to know what queries are raised, and whether the model can be argued to be accurate on those queries. Given the conceptualisation in TR 5469, this seems to me to be all that is required.

Exactly how this is operationalised for various representations of risk is a further question. Since functional safety in standards is phrased in terms of risk, this is the big question. I do not address it further here.

6 Summary

Adaptive control was one of the first applications of machine learning and DLNNs in safety-critical systems. However, the general architecture of ISO/IEC TR 5469 does not fit the architecture of adaptive control systems. The functional safety of adaptive controllers has not been adequately addressed in electrotechnical standards and it follows that it will not be adequately addressed through publication of TR 5469. The notions of precision/vagueness, accuracy, and traceability apply (or not) to such oracular systems as are built using AI and it seems as if desirable properties for functional safety can be phrased in terms of them.

Correspondence Address

Corresponding e-mail address: ladkin@causalis.com.

Acknowledgments

I thank Stuart Russell for apprising me of D'Amour et al's work, and Ross Anderson for apprising me of Shumailov et al's work.

The images of Figures 1 and 2 are from the NASA Dryden Flight Research Center[10] photo collection. Figures 3, 4 and 5 are from the Defense Advanced Research Projects Agency of the US Government (DARPA), and available from Wikimedia.

References

D'Amour A., Heller K., Moldovan D., Adlam B., Alipanahi B., Beutel A., Chen C., Deaton J., Eisenstein J., Hoffman M. D., Hormozdiari F., Houlsby N., Hou S., Ferfel G., Karthikesalingam A., Lucic M., Ma Y., McLean C., Mincu D., Mitani A., Montanari A., Nado Z., Natarajan V., Nielson C., Osborne T. F., Raman R., Ramasamy K., Sayrea R., Schrouff J., Seneviratne M., Sequeira S., Suresh H., Veitch V., Vladymyrov M., Wang X., Webster K., Yadlowsky S., Yun T., Zhai X., and Sculley D. (2020). *Underspecification Presents Challenges for Credibility in Modern Machine Learning*. Preprint, 2020-11-06, available at https://arxiv.org/abs/2011.03395. Accessed 29th August 2023.

DARPA Grand Challenge (2005). (no date). In Wikipedia: https://en.wikipedia.org/wiki/DARPA_Grand_Challenge_(2005) . Accessed 29th August 2023.

[10] Now the NASA Armstrong Flight Research Center.

Dempster–Shafer theory. (no date). In Wikipedia: https://en.wikipedia.org/wiki/Dempster–Shafer_theory. Accessed 29th August 2023.

Franklin G. F., Powell D. J., and Emami-Naeini E. (1994). *Feedback Control of Dynamical Systems*, Third Edition. Addison-Wesley, Boston MA.

Harris M. (2019). *NTSB Investigation Into Deadly Uber Self-Driving Car Crash Reveals Lax Attitude Toward Safety*. IEEE Spectrum, 2019-11-07. Available at https://spectrum.ieee.org/cars-that-think/transportation/self-driving/ntsb-investigation-into-deadly-uber-selfdriving-car-crash-reveals-lax-attitude-toward-safety. Accessed 29th August 2023.

ISO/IEC Directives. (2021). *Procedures for the technical work — Procedures specific to IEC*. Directives Part 1 + IEC Supplement, Edition 17, 2021-05. International Organisation for Standardization/International Electrotechnical Commission, Geneva.

IEC 61508. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems*. IEC 61508, in 7 parts[11], 2nd Edition, 2010. International Electrotechnical Commission, Geneva.

IEV. (no date). *International Electrotechnical Vocabulary*. International Electrotechnical Commission, IEC 60050. Available from https://www.electropedia.org. Accessed 29th July 2023.

ISO/IEC 5469. (2024). *Artificial Intelligence – Functional safety and AI systems*. ISO/IEC TR 5469:2024, 1st Edition, January 2024. International Organization for Standardization/International Electrotechnical Commission, Geneva.

ISO/IEC 22989. (2022). *Information technology — Artificial intelligence — Artificial intelligence concepts and terminology*. ISO/IEC 22989:2022, 1st Edition, July 2024. International Organization for Standardization/International Electrotechnical Commission, Geneva.

Knowledge representation and reasoning. (no date). In Wikipedia: https://en.wikipedia.org/wiki/Knowledge_representation_and_reasoning. Accessed 29th August 2023.

MacKenzie D. (2001). *Mechanizing Proof*. M.I.T. Press, Cambridge MA.

Minsky M., and Papert S. (1969). *Perceptrons*. M.I.T. Press, Cambridge MA.

NASA. (2002). *MD-11 Propulsion Controlled Aircraft (PCA)*. National Aeronautics and Space Administration (NASA). Available at https://www.nasa.gov/centers/dryden/multimedia/imagegallery/MD-11PCA/MD-11PCA_proj_desc.html. Accessed 29th August 2023.

NASA. (2014). *NASA Armstrong Fact Sheet: NF-15B Research Aircraft*. National Aeronautics and Space Administration (NASA). Available at https://www.nasa.gov/centers/armstrong/news/FactSheets/FS-048-DFRC.html. Accessed 29th August 2023.

Nelson G., and Oppen D. C. (1979). *Simplification by cooperating decision procedures*. ACM Transactions on Programming Languages and Systems, 1(2):245–257, 1979.

[11] Parts are typically denoted with hyphen and part number, e.g., IEC 61508-3:2010 is Part 3, dealing with software development requirements.

Nelson G., and Oppen D. C. (1980). *Fast decision procedures based on congruence closure*. J. Ass. Comp. Mach., 27(2):356–364, 1980.

Newell A, Shaw J. C., and Simon H. A. (1959). *Report on a general problem-solving program*. Proceedings of the International Conference on Information Processing, Paris 15–20 June 1959. Available at http://bitsavers.informatik.uni-stuttgart.de/pdf/rand/ipl/P-1584_Report_On_A_General_Problem-Solving_Program_Feb59.pdf. Accessed 29th August 2023

Nguyen N., and Jacklin S. A. (2010). *Stability, Convergence, and Verification and Validation Challenges of Neural Net Adaptive Flight Control*. In: Schumann J., and Liu Y. (editors) *Applications of Neural Networks in High Assurance Systems*. Studies in Computational Intelligence, vol 268. Springer, Berlin, Heidelberg.

Pearl J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann, Burlington MA.

Rumelhart D. E., and McClelland J. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Foundations*. M.I.T. Press, Cambridge MA.

Shumailov I., Schumaylov Z., Kazhdan D., Zhao, Y., Papernot N., Erdogdu M. A., and Anderson R. (2021). *Manipulating SGD with Data Ordering Attacks*. Preprint 2021-04-19, available at https://arxiv.org/abs/2104.09667. Accessed 29th August 2023.

Shepardson D. (2021). *U.S. safety agency reviewing 23 Tesla crashes, three from recent weeks*. Reuters, 2021-03-18. Available at https://www.reuters.com/article/us-tesla-crash-idUSKBN2BA2ML. Accessed 29th August 2023.

Shostak R. E. (1984). *Deciding combinations of theories*. Journal of the ACM, 31(1):1–12, 1984.

Smith T., Barhorst J., and Urnes Sr. J. M. (2010). *Design and Flight Test of an Intelligent Flight Control System*. In: Schumann J., and Liu Y. (editors) *Applications of Neural Networks in High Assurance Systems*. Studies in Computational Intelligence, vol 268. Springer, Berlin, Heidelberg.

This collation page left blank intentionally.

Appendix A. Some IEC Definitions in Functional Safety and AI Systems

artificial intelligence
AI
<discipline> research and development of mechanisms and applications of *AI systems* (3.1l.4)

Note 1 to entry: Research and development can take place across any number of fields such as computer science, data science, humanities, mathematics and natural sciences.

[SOURCE: ISO/IEC CD 2 22989:2022, 3.1.3]

system
set of interrelated elements considered in a defined context as a whole and separated from their environment

> Note 1 to entry: A system is generally defined with the view of achieving a given objective, for example by performing a definite function.
>
> Note 2 to entry: Elements of a system may be natural or man-made material objects, as well as modes of thinking and the results thereof (for example forms of organisation, mathematical methods, programming languages).
>
> Note 3 to entry: The system is considered to be separated from the environment and the other external systems by an imaginary surface, through which pass the links between them and the considered system.
>
> Note 4 to entry: The term "system" should be qualified when it is not clear from the context to what it refers, for example control system, calorimetric system, system of units, transmission system.
>
> Note 5 to entry: This entry was numbered 351-21-20 in IEC 60050-351:2006.
>
> [SOURCE: IEC 60050, the International Electrotechnical Vocabulary, 351-42-08]

system of systems

set of operationally and managerially independent systems that are operated together for a period of time to achieve one or more stated purposes

[SOURCE: IEC 60050, the International Electrotechnical Vocabulary, 871-05-03]

systematic failure

failure, related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors

[SOURCE: IEC 61508-4, ed. 2.0 (2010), 3.6.6]

machine learning
ML
process of optimizing *model parameters* (3.3.8) through computational techniques, such that the *model's* (3.1.23) behaviour reflects the data or experience

[SOURCE: ISO/IEC 22989:2022, 3.3.5]

machine learning algorithm
algorithm to determine *parameters* (3.3.8) of a *machine learning model* (3.3.7) from data according to given criteria

> EXAMPLE Consider solving a univariate linear function $y = \theta 0 + \theta 1x$ where y is an output or result, x is an input, $\theta 0$ is an intercept (the value of y where x=0) and $\theta 1$ is a weight. In *machine learning* (3.3.5) the process of determining the intercept and weights for a linear function is known as linear regression.

[SOURCE: ISO/IEC 22989:2022, 3.3.6]

deep learning
deep neural network learning
<artificial intelligence> approach to creating rich hierarchical representations through the *training* (3.3.15) of *neural networks* (3.4.8) with many hidden layers

> Note 1 to entry: Deep learning is a subset of *ML* (3.3.5)

[SOURCE: ISO/IEC 22989:2022, 3.4.4]

artificial intelligence system
AI system
engineered system that generates outputs such as content, forecasts, recommendations or decisions for a given set of human-defined objectives

> Note 1 to entry: The engineered system can use various techniques and approaches related to *artificial intelligence* (3.1.3) to develop a *model* (3.1.23) to represent data, *knowledge* (3.1.21), processes, etc. which can be used to conduct *tasks* (3.1.35).
>
> Note 2 to entry: AI systems are designed to operate with varying levels of *automation* (3.1.7).

[SOURCE: ISO/IEC 22989:2022, 3.1.4]

model
physical, mathematical or otherwise logical representation of a system, entity, phenomenon, process or data

[SOURCE: ISO/IEC 22989:2022, 3.1.23]

Appendix B. Extract from Draft of TR 5469 from June 2022

8 Properties and related risk factors of AI systems

8.1 Introduction

8.1.1 General

Clause 7 describes how the definition of desirable properties is the first step of the three-stage realization principle. The properties are related to topics and eventually to detailed methods and techniques addressing those topics. Acceptance criteria are then identified from the set of the detailed methods and techniques.

This Clause provides guidance on the properties that characterize systems using AI technology and their related risk factors. Such properties and risk factors include degree of automation and control (Clause 8.2), degree of decision transparency and explainability (Clause 8.3), environmental complexity and vagueness of their defining specifications (Clause 8.4), resilience to adversarial inputs (Clause 8.5), system hardware considerations (Clause 8.6) and technological maturity (Clause 8.7).

Details of the properties and risk factors of systems using AI technology, and their related aspects and challenges, are discussed in this Clause.

8.1.2 Algorithms and models

On a technological level, the capability of AI is often achieved by the combination of an algorithm and a model. The model typically represents information that achieves the application's purpose, (e.g. knowledge about how various inputs are to be distinguished and recognized), while algorithms infer information from a model and inputs, (e.g. to make a prediction). This means the functional safety of applications using AI technology depends on both.

Example types of algorithms include linear functions, logical calculi, dynamic Bayesian networks and artificial neural networks. The models can either be handcrafted by an engineer, or can be synthesized from data by machine learning algorithms that themselves use a systematic analysis process. The algorithms are usually implemented as an executable representation, such as machine code (in the case of software), or special hardware, such as field programmable gate arrays (FPGAs).

Usually, algorithms that interact with the models contain only a limited amount of knowledge or implications about the application's goals. This is quite similar to the role of foundational software libraries or programming environments (compilers, etc.) in non-AI software. That is, the algorithm itself does not play a functional safety role, but its correctness is critically important for functional safety to be achieved. In this way, the integrity of algorithms in AI technology can often be handled with existing principles of functional safety as specified in the IEC 61508 series [16]-[19], similar to that of non-AI software components. The same holds for the logic involved in the translation of the algorithm and the model.

By contrast, models often contain knowledge related to the objective of the systems involving functional safety. There are several different ways of constructing models and different approaches can be used for assessing the completion of risk reduction measures to ensure functional safety.

For example, when models are created manually by engineers, the models can likely reflect the engineers' knowledge about the application, which can be assessed during the management processes used within functional safety lifecycles. In these cases, the lifecycle of existing functional safety International Standards can be followed (AI technology Class I as described in Clause 6.2). It is often feasible to create models manually for simple algorithms such as simple linear functions or logical calculi.

In some cases, models derived from data by machine learning algorithms can be analysed and verified after their creation. Alternatively, models derived by machine learning algorithms can be analysed, the underlying parameters extracted and used to extend general engineering knowledge, that, in turn, can be used to develop further models. With the application of validated engineering knowledge, the lifecycle of existing functional safety International Standards can again be applied (e.g. treating these models as AI technology Class I as described in Clause 6.2).

In other cases, models derived from data by machine learning algorithms can be too complex to be understood, analysed and verified. This is often the case for complex types of models, such as neural networks, because representations of models in these types do not necessarily reflect human understanding or reasoning. The use of different approaches for assessing the risk reduction for functional safety is appropriate in these cases, which and can constitute a major challenge for the use of AI technology in implementation of functional safety systems.