Planning and Control Method Based on Fuzzy Logic for Intelligent Machine

Anatolii Kargin and Tetyana Petrenko

Ukrainian State University of Railway Transport, Feuerbach sq., 7, Kharkiv, 61050, Ukraine

Abstract

Intelligent Machine (IM) control is based on a complex solution of three tasks: perception of data from sensors, planning actions and decisions making (DM) in accordance with the actual plan and process's history. The solution to this problem is the main challenge of the IM. Spatio-temporal datasets from a lot of heterogeneous sensors, non-deterministic, dynamic, and partially observable characteristics of environment impose additional restrictions on the DM technology. Fuzzy logic systems (FLS) are not used in such conditions due to the large dimension of the problem. The two-stage Computing with Words (CWW) approach overcomes the dimensionality problem by using abstraction engine at the first stage, which maps the meaning of data from a lot of sensors to the meaning of a few words. In this article the CWW technology is extended by a Short-Term Memory (STM) and Long-Term Memory (LTM) models. The STM stores a time sequence of data in the form of an ordered sequence of events. The LTM stores an action plan in the form an ordered sequence of plan stages. Both are formalized in the form of a flat vector field each element of which is represented on word fuzzy characteristics. The STM model is supported by the footprint blur algorithm, and LTM model is supported by dynamic planning algorithm. An example of the use of STM and LTM data in the FLS when decisions making by IM, is given.

Keywords 1

Fuzzy logic systems, intelligent machine control, abstraction engine, data from sensors, short-term memory, long-term memory, dynamic planning of action

1. Introduction

Decision-making (DM) and reacting to inputs are the main tasks of the Internet of Things (IoT). DM by Intelligent Machine (IM) and smart machine is a major challenge that IoT developers are focusing [1-4]. Two factors are considered when making control decisions in these applications. Firstly, knowledge about the process's full history and, secondly, an action plan to achieve the goal. Due to the complexity of formalizing process's full history in the form of a temporal sequence of input data and previously adopted controls, Artificial Intelligence (AI) and IoT use the Markov model and Bayesian Markov Decision Processes [5,6]. However, with such a simplification, IM cannot always find a solution that determines its rational behavior. Examples can be given where IM cannot find a control decision without knowing the process's history. The IM approaches a crossroad, and its sensory system detects a yellow traffic light. Without information about what signal was before, it is impossible to make the right decision (to slow down or continue to move at the same speed). Another example of an IM that serves an automated warehouse. The order in which the containers were previously loaded in IM determines the subsequent logistics of container delivery. For example, IM decides at position F to continue moving either to position *H* or to position *G* (Figure 1). The decision depends on the prehistory: where (from position *A* or *B*) and by what plan (via position *D* or *E*) IM arrived at position F. To decide, the IM must store a sequence of data from sensors, namely, a

COLINS-2021: 5th International Conference on Computational Linguistics and Intelligent Systems, April 22–23, 2021, Kharkiv, Ukraine
EMAIL: kargin@kart.edu.ua (A. Kargin); petrenko_tg@kart.edu.ua (T. Petrenko)
ORCID: 0000-0003-2885-9071 (A. Kargin); 0000-0001-6305-7918 (T. Petrenko)

sequence of data describing situations at different stages of the plan. In addition, if the original plan involved moving the IM to position *G*, loading the container there and moving to position *I*, and the obstacles disrupted this plan, then replanning is required. Such obstacles could be blockage of the passage from position *F* to position *G*, lack of containers for loading at position *G*, full loading of IM with containers at position *F*, so that IM cannot be loaded anymore. When replanning, IM should take into account the completed part of the plan, the environment current state and the remaining fragment of the plan for execution. The above example shows that IM should be able to comprehensively solve the following tasks: 1) perception of data from sensors about the environment and the state of the IM; 2) processing of data on IM process's full history; 3) planning, using the accumulated experience, monitoring the implementation of the plan; 4) DM in accordance with the current plan, based on data about the process's history and current state of the environment. The tasks listed are interrelated, they share knowledge and data, and must be solved in an integrated manner in real time.

The purpose of this work is to choose models that can be comprehensively used to solve the listed tasks, and, on their basis, to propose a control method for IM, which will ensure the execution of the action plan in conditions of non-determinism.

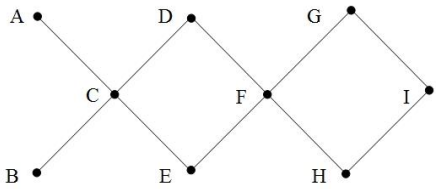

Figure 1: Containers loading and delivery action plan of IM

The central task from the above list is the direct task of making decisions. The other three tasks prepare the data for control DM. When justifying the choice of a DM model, it is necessary consider the requirements due to the IM domain specifics. The IM domain refers to non-deterministic (limited or unlimited non-determinism), dynamic and partially observable. In addition, such factors as IM autonomy, the real time mode, and the need to use heterogeneous data from many different sensors impose additional restrictions on the DM technology [7, 8]. The basic conceptual model for such domain is Rules Based Systems. Most development platforms use the Rules Engine (RE) to implement the functionality of an IoT application [9, 10]. In REs, AI methods and models are used to find solutions, which take into account the specified features of the IoT domain to varying degrees. However, due to the large dimension of the problem, the well-known REs does not use the Fuzzy Logic System (FLS), which have proven themselves well in solving real-time DM problems in non-deterministic conditions [7, 8, 11]. In order to create conditions for the use of FLS in RE, in [12, 13], a two-step Smart RE (SRE) technology is proposed, which allows one to reduce the dimension of the problem. In SRE, at the first data processing step, the Abstraction Engine (AE), by generalization and abstraction, receives a description of the input data from the sensors in the form of a small number of a high-level generalization words of natural language. Fuzzy characteristics of such words, which represent the meaning of a large-dimensional spatio-temporal segment of data from sensors, are then used as input FLS variables at the second step to make a decision. The FLS model, integrated in this way with the AE model and tested on DM problems in IM [12, 13], is the basis of the proposed method.

The IM autonomy and the real-time mode impose the main restrictions on the choice of the planning model. Action planning should be carried out in close connection with the implementation of the plan by making control decisions that ensure the implementation of the plan stages. This is a premise for the integration of Automated Planning Engine (APE) with FLS. Classic planning considered in AI is mainly focused on creating algorithms for a deterministic, fully observable environment [6, 14, 15, 16, 17]. Most of them focus on the technique of finding a good plan under the assumption that the planner has complete knowledge of the current world state and the causal relationships that govern changes in this world [6, 14]. The implementation of these technologies for IM is difficult since these methods do not allow to achieve a sufficiently high performance and do not imply re-planning for a non-deterministic environment. If the performance problem is overcome using various techniques (plans hierarchy, reactive planning, rule-based planning [6, 16, 17, 18]), then the

issues of replanning mechanism “from what has been achieved” remain open [6, 19]. Significant results on real-time scheduling for a non-deterministic environment have been obtained for mobile robots and autonomous mobile systems [20, 21]. The planning algorithms for these systems are aimed at the tasks of navigation and localization of mobile systems, take into account the domain peculiarities and cannot be implemented as an universal planning method, the description of the plan stages of which can be used in DM model. As an alternative to re-planning, mechanisms are considered in which it is proposed give up complicated planning in favor of sketchy planning using explanation-based reasoning approaches or machine learning [22]. Case-based reasoning approaches [6, 23] opens opportunities for quick re-planning (from what has been achieved) using ready-made fragments of the plan (precedents). When using Case-based reasoning, re-planning technology "from scratch" can be avoided if fragments of the completed plan (CPF) and fragments of the plan to be executed (EPF) have the same presentation model. The choice of a planning model is also influenced by the need to integrate the APE with the FLS. From all the above, it follows that in order to make decisions in IM by jointly solving the previously listed tasks, a model is required that will interface the inputs and outputs of the following components: FLS, automatic planning systems, AE and history data processing systems.

The main hypothesis of this article is that the memory model can perform the function of conjugation of individual models that underlie DM in IM. Memory is seen not just as a data store, but as a component that performs data processing. Figure 2 shows the structure of the IM control system as a possible variant of the hypothesis implementation.

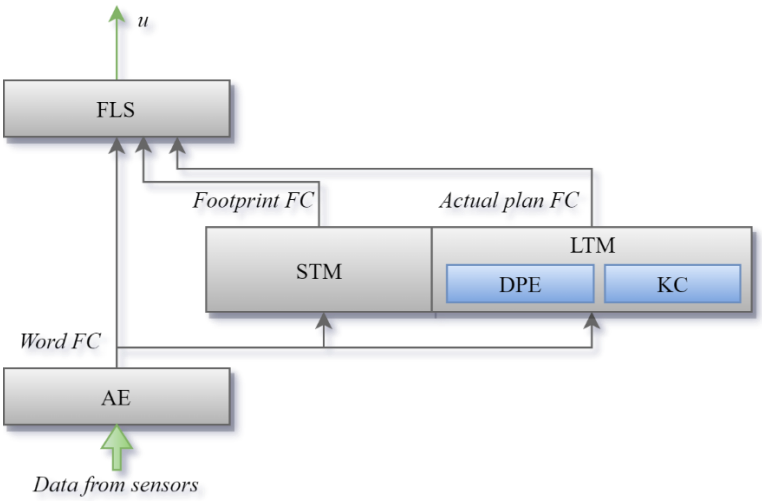

Figure 2: Intelligent machine control center

The first thing to decide when choosing a memory model is in what form to store the process's full history. For an IoT application, storing a spatio-temporal segment of the primary "raw" data from sensors directly is not realistic. In the IoT, special attention is paid to the issues of obtaining meaningful information from sensor data for DM [24, 25]. However, methods of preliminary processing of data from sensors, including data fusion, do not lead to a significant reduction the data, while maintaining their meaning. The issues of the meaning of data during memorization and retrieval from memory are considered in cognitive psychology [26-28]. Two models of short-term (STM) and long-term (LTM) memory have been studied. It is shown that the storage of a temporal sequence of significant data in STM is organized in the form of an ordered sequence of characters that reflect the meaning of the perceived data. There are still different views on the interaction between STM and LTM. This work is based on the conceptual model [29, 30], according to which LTM implements a data storage in the form of an ordered sequence, allows to create new structured representations from these data items, and transfers data from LTM to STM in the correct order. However, there is no information about the computer implementation of such a complex memory model for applications that process data from sensors in real time. Detached functions of the cognitive memory model have been implemented, for example, an artificial neural network model called long-short-term memory

LSTM [31]. The memory models used in this work with the abbreviated names STM and LTM have nothing to do with the above model LSTM of a neural network.

The aim of this work is to propose a memory model in which data storage and processing is carried out uniformly for both STM and LTM components. In STM, the process's history is stored and processed in the form of a sequence of events, and in LTM, the actual plan of operation is in the form of a sequence of stages. The task is to integrate the STM&LTM memory model with AE and FLS. Show the possibility of using FLS for control in IM, taking into account an action plan in a non-deterministic situation, which is formed in real time based on data from a large number of sensors.

Section 2 provides models of the individual components that support FLS decision making. Section 2.1 begins with review how AE maps the meaning of data from sensors to the meaning of a words and continues with a detailed outline of the STM model in section 2.2. The STM data storage model is introduced in the form of an events sequence footprint represented by a dynamic flat vector field. An algorithm for processing an event fingerprint is given. Section 2.3 introduces a model for storing an action plan in LTM in the form of a knowledge cube, the slice of which represents a separate plan, which is also formalized in the form of a flat vector field. Section 2.4 presents a dynamic planning algorithm in the form of operations on fuzzy characteristics of a vector field. Section III will present method of constructing FLS, namely linguistic variables and fuzzy rules, which use fuzzy characteristics of the situation, IM's history, and action plan. Also, will present experiments and their discussion.

2. Intelligent machine control center: the models

The block diagram of the IM control center (IMCC) is shown in Figure 2. As already noted, the IM functionality is supported by FLS, or rather by a knowledge base containing fuzzy rules for making control reactions. AE, STM and LTM prepare the FLS input. AE perceives data from sensors and maps them into the meaning of high-level abstraction words. The meaning of a word is formally represented by the certainty factor (CF) and is a fuzzy characteristic (FC) of the word. CF values are input numeric variables FLS, STM and LTM. Words CFs are a description of the environment current state. The AE algorithm for calculating the word CF is given in detail in [12, 13]. The STM stores and processes data about the IM's history, namely, the description of the events sequence that occurred earlier in the IM environment. FC of such temporal fragments of history are also input numerical variables of FLS. The LTM stores knowledge about all possible IM plans and processes the actual plan at the current time. The APE calculates the fuzzy characteristics of the individual stages of the plan and carries out dynamic planning (at the end of the current stage of the plan, it updates the EPF). The CFs of the individual plan stages are FLS input numerical variables, too. In addition to APE, LTM includes a Knowledge Cube (KC) about possible plans, as seen in Figure 2. Precedent plans in the KC are replenished, either by learning (accumulating their own IM experience), or by inputting expert knowledge.

2.1. AE: Word meaning representation

Below, a word meaning model in the form of a fuzzy CF is discussed in a concise form. The AE perception system receives information about the IM state and the environment using n sensors. The set of possible values of the ith sensor is divided into l_i information granules, so that the set of possible data from the sensors is $\mathbf{L}$, $Card(\mathbf{L}) = l_1 + l_2 + \ldots + l_n$.. At the tth step of receiving data from the sensors, AE maps the meaning of "fresh" data (a subset of $\mathbf{L}$) to the meaning of a subset of words $\mathbf{W} = \{w_1, w_2, ..., w_k\}$, where k is significantly less than $Card(\mathbf{L})$. Thus, the dimension of the problem is significantly reduced by mapping the meaning of data from lot of sensors into the meaning of a few words. AE performs such a mapping based on the domain knowledge base using the Computing with Words (CWW) technology [32, 33]. In [12, 13], a model of the word meaning was introduced by word FC, which is represented by a special fuzzy L-R number.

$$\mathbf{X}: \{x \mid m_X(x), \forall x \in [-1, +1]\} \quad (1)$$

with Gaussian L-R membership function

$$m_{\mathbf{x}}^{L} = \exp(-(x-\alpha)^2 / 2t_L^2, \forall x \in [-1, \alpha] \quad (2)$$
$$m_{\mathbf{x}}^{R} = \exp(-(x-\alpha)^2 / 2t_R^2, \forall x \in [\alpha, +1]$$

the parameters of which are the certainty $(-1 \le \alpha \le +1)$ and the time intervals $t=t_L+t_R$ $(0 \le t < \infty)$, where t_L and t_R are the time intervals since the last data acquisition from the sensor and the data change, respectively.

Based on the FC (1), the CF as a fuzzy numerical characteristic $-1 \le cf \le +1$ of the word meaning is calculated.

$$cf = \alpha \cdot h_t \quad (3)$$

where

$$h_t = 1 - \frac{\sum_{\forall x \in [-1,\alpha]} m_{\mathbf{x}}^{L}(x) + \sum_{\forall x \in [\alpha,+1]} m_{\mathbf{x}}^{R}(x)}{Card([-1,+1])} \quad (4)$$

Thus, at an arbitrary time t, the meaning of the spatio-temporal segment of data received from the IM sensors is mapped into meaning of the words. of a set of AE vocabulary, that is

$$\{cf(\mathrm{w}_i), \ \forall \mathrm{w}_i \in \mathbf{W}\} \quad (5)$$

where **W** is AE vocabulary, w_i is ith word designation, $cf(\mathrm{w}_i)$ is the meaning of the ith word.

This word meaning model forms the basis of the STM and LTM models considered in this work.

2.2. STM: Events history footprint

STM stores the IM's history as a time-ordered sequence of $m + 1$ latest events. An event is considered to be a change in the data of the set **L**. Since the meaning of data in AE is represented by the meaning of words $\mathrm{w} \in \mathbf{W}$, the event in STM is stored as a set of words $\{\mathrm{w}^0_i, \mathrm{w}^{-1}_j, \mathrm{w}^{-2}_p, ..., \mathrm{w}^{-m}_s\}$, whose CF values have changed. The digits $q = 0, -1, -2, ..., -m$ indicate the ordinal number of the event in the sequence with respect to the event w^0_i that occurred at the current moment in time. The event w^{-1}_j happened earlier before the event w^0_i, and so on. Events are not tied to a specific timeline.

A dynamic discrete flat vector field is taken as a numerical model of STM (Figure 3). The x-axis in the right half-plane indicates the elements of the set of words $\mathbf{W} = \{\mathrm{w}_1, \mathrm{w}_2, ..., \mathrm{w}_k\}$, and the elements of the set of events $\boldsymbol{W} = \{w_1, w_2, ..., w_k\}$ are indicated in the left half-plane. Events are indicated in italics to distinguish them from the corresponding words. The negative semiaxis y in the left half-plane shows the ordinal numbers of events q with respect to the event that has occurred at the current time. For an event that has occurred at the current moment of DM, $q = 0$. On the positive y-axis in the right half-plane, the ordinal numbers of the stages of the upcoming EPF are indicated. The z-axis shows the word CF cf. We consider a special discrete vector field in which a vector $\boldsymbol{cf}(\mathrm{w}_i, q)$ is associated with each point of the two-dimensional space (w_i, q). A vector can only have two directions along the z-axis: positive and negative, as shown in Figure 3.

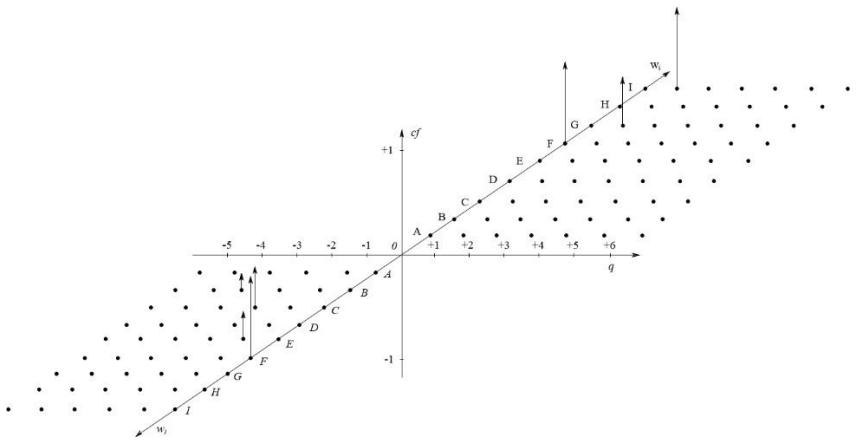

Figure 3: An example of a state of a flat discrete vector field

A dynamic vector field is considered, in which at the event occurrence moment the characteristics values of all previously occurred events are changed. Firstly, each event is shifted "deep into memory" (the parameter q of the event w^q_i is replaced by $q - 1$, w^{q-1}_i) and, secondly, the vector modulus is reduced by the coefficient of "forgetting" $exp(-v)$, where $0 \le v \le 1$ is the data aging rate. This operation simulates the results of studies of STM, obtained in cognitive psychology [23, 24], about the gradual "blur" of event footprint in the memory. Thus, the content of an STM at an arbitrary time represents an Events sequence Footprint (EF).

Let us introduce a formal representation of the vector field before proceeding to the consideration of the STM characteristics updating algorithm when the event w_i appears. Taking into account the above features, we represent the vector field by a matrix and denote

$$\mathbf{cf} = \begin{pmatrix} cf(w_1,0) & cf(w_1,-1) & \dots & cf(w_1,-m) \\ cf(w_2,0) & cf(w_2,-1) & \dots & cf(w_2,-m) \\ \dots & \dots & \dots & \\ cf(w_k,0) & cf(w_k,-1) & \dots & cf(w_k,-m) \end{pmatrix} \quad (6)$$

Since in the introduced vector field model only a limited set of vectors is used (directed only along the z-axis in the positive or negative direction), then in (5), instead of vectors, scalars are indicated in the form of signed numbers $cf(w_i, q)$. The vector field is processed using the Footprint Blur algorithm (FBA). Following is FBA of the STM characteristics updating.

1. Initialization of the algorithm at the moments of time when an event occurs related to the CF change of any word $w \in \mathbf{W}$ from the AE dictionary. A simple computational procedure for determining an event is proposed.

$$cf(w_i) = \begin{cases} cf(w_i), \ if \ \ cf(w_i) > e \ or \ cf(w_i) < -e, \\ 0, \quad otherwice, \end{cases} \quad (7)$$

where $(0 < e < 1)$ is the sensitivity threshold for data changes.

2. Removing the $(m+1)$th column of matrix (6) and multiplying the resulting matrix $k \times m$ by the forgetting coefficient

$$\mathbf{cf}' = \exp(-v) \cdot \mathbf{cf}(:, m+1) \quad (8)$$

3. Horizontal concatenation of a column-vector $\mathbf{cf}''$ in which, apart from w_p, all other elements are 0 and the resulting matrix $\mathbf{cf}'$ in (8)

$$\mathbf{cf}'' = \begin{pmatrix} 0 \\ \dots \\ cf(w_p,0) \\ \dots \\ 0 \end{pmatrix} \quad (9)$$

$$\mathbf{cf} = [\mathbf{cf}'', \mathbf{cf}'] \quad (10)$$

4. Calculating the confidence that EF corresponds to a given sequence of events, for example, the EPF. Figure 3 shows the state of EF at the time when the event F appeared. Formally, EF and action plan, например, PL are represented by matrices (6), in which $m_{PL} \ne m_{EF}$, in the general case. Therefore, before performing the matching operation between EF and PL, it is aligned to the size of the matrix m_{EF}, either by cutting off the last $m_{PL} - m_{EF}$ events when $m_{PL} > m_{EF}$, or by adding columns in EF (columns with numbers $m_{PL}+1$, $m_{PL}+2, \dots, m_{EF}$) containing zeros. Preliminarily, the projections of the matrices (6) EF and PL are found: the first (vertical) and second (horizontal) projections of the vector fields (6). For this, a modified operation of matrix convolution was used [34].

$$\begin{aligned} &\mathbf{cf}^1 = (cf^1(q) = \underset{i=1,k}{MAX}(cf_{EF}(w_p,q) \cdot cf_{PL}(w_p,q)), \ q = -1,\dots,-m), \\ &\mathbf{cf}^2 = (cf^2(w_i) = sign(cf_{EF}(w_i,q)) \cdot \underset{q=-1,-2,\dots,-m}{MAX}(|\, cf_{EF}(w_i,q)\,|), \ i = 1,\dots,k). \end{aligned} \quad (11)$$

Certainty that EF corresponds to PL (Certainty Match the EF to PL) cf_{EF_PL} is calculated as the weighted sum of the first projections (11).

$$cf_{EF\_PL} = \frac{1}{m*} \sum_{q=-1,-2,...,-m} cf^1(q) \cdot \quad (12)$$

5. The end of algorithm.

The use of the cf_{EF_PL}, $cf^1(q)$ and $cf^2(w_i)$ values in the STM, LTM and FLS models will be discussed below.

2.3. LTM: knowledge representation of the plan

Knowledge about the plan is represented on the set of words $w_i \in \mathbf{W}$, each of which reveals the meaning of a certain situation by $cf(w_i)$. An Action Plan (AP) is a plan to achieve a target situation, for example $w_{**} \in \mathbf{W}$ from a starting situation, for example $w_* \in \mathbf{W}$. The AP is represented by an ordered sequence of words

$$Pl=(w_*, w_l, w_g, ..., w_p, w_{**}), \quad (13)$$

in which for each pair of adjacent words there is knowledge of cause-and-effect relationships in the form of triplets

$$(w_*, u_*) \to w_l, \ (w_l, u_l) \to w_g, \ (w_g, u_g) \to w_d, ..., (w_p, u_p) \to w_{**}, , \quad (14)$$

where $u_j, u_l, u_g, ..., u_p$ there are control actions that ensure the execution of the plan.

Expressions (13), (14) represent the *Q = Card (Pl)* stage plan. A separate stage of the plan, for example, the third in (14) includes the word w_g, which, on the one hand, is a description of the situation characterizing the successful completion of the previous second stage of the plan, and, on the other hand, is a condition necessary for the successful implementation of the action u_g of this third stage plan. Let's call a word, for example, w_g a descriptor of a plan stage.

This paper proposes a unified data representation model for both STM and LTM. Therefore, just like the sequence of events in EF, we will represent the sequence of plan stages (13) as a set of words ordered by plan stages (index *q*).

$$PL(w_*, w_{**})=(w_*^0, w_l^{+1}, w_g^{+2}, ..., w_p^{+q}, ..., w_{**}^{+Q}), \quad (15)$$

where the superscript, for example, *q* in the word designation w_p^q is an integer indicating that the word belongs to the *q*th stage of the plan.

We will call a fragment of plan (15) any subsequence of adjacent stages for which the monotonicity of the numbering of *q* stages is preserved. This means that for a fragment $PL(w_j, w_r) \in PL(w_*, w_{**})$ for which $w_j^{q_1}, w_r^{+q_2}$, $q_1<q_2$ the condition $(w_j^{q_1}, w_l^{q_1+1}, w_g^{q_1+2}, ..., w_p^{+(q_2-1)}, w_r^{+q_2})$ must be met. In what follows, we will assume that in the fragment of the plan, as well as in the plan (15) for the initial stage, *q = 0*.

The LTM stores knowledge of N different plans.

$$\mathbf{PL}=\{\{w_*^{j0}, w_l^{+j1}, w_g^{+j2}, ..., w_q^{+jq}, ..., w_{**}^{+jQ}\}, \ j=1,2,...,N\}. \quad (16)$$

The set of all possible plans (16) is represented by a three-dimensional $k \times Q \times N$ KC, where Q is the maximum allowable length of the plan (the number of stages). Below is the *i*th slice KC, representing knowledge about the *j*th plan (16) in the form of a matrix.

$$\mathbf{PL}_j = \begin{pmatrix} w/r & 0 & +1 & ... & +q & ... & +Q \\ w_1 & cf_{PL_j}(w_1,0) & cf_{PL_j}(w_1,1) & ... & cf_{PL_j}(w_1,q) & ... & cf_{PL_j}(w_1,Q) \\ w_2 & cf_{PL_j}(w_2,0) & cf_{PL_j}(w_2,1) & ... & cf_{PL_j}(w_2,q) & ... & cf_{PL_j}(w_2,Q) \\ ... & ... & ... & ... & ... & ... & ... \\ w_k & cf_{PL_j}(w_k,0) & cf_{PL_j}(w_k,1) & ... & cf_{PL_j}(w_k,q) & ... & cf_{PL_j}(w_k,Q) \end{pmatrix}. \quad (17)$$

In (17) $cf_{PL_j}(\mathrm{w}_i, q)$ there is CF in that the word w_i is a descriptor of the qth stage of the plan PL_j. When there is no data about the situation represented by the descriptor w_i, or are outdated, then the meaning of the word is $cf_{PL_j}(\mathrm{w}_i, q) = 0$. If the situation represented by the w_i descriptor indicates that the stage of the plan has not been completed or has been performed unsuccessfully, then the meaning of the word is there $cf_{PL_j}(\mathrm{w}_i, q) = -1$. Finally, when the w_i describes the local target situation of the plan stage, then $cf_{PL_j}(\mathrm{w}_i, q) = +1$.

2.4. LTM: dynamic planning engine

The IMCC model under consideration is Goal-driven planning. The model is based on the following hypothesis. At time *t* AE, based on data from sensors, forms a meaning of the situation with the word w*. In LTM, the currently active target is represented by the word w**. These two activated words are actualized in the KC slice, which contains a $PL(\mathrm{w}_*, \mathrm{w}_{**})$ fragment of the plan (17). This is AP at time *t*. Thus, at time *t* in IMCC, the following are relevant: EF in STM, which stores data on previously executed plan stages (CPF as history of events), and AP in LTM, which stores stages of a plan that has not yet been completed (EPF).

The DPE performs AP data processing. Just like EF, AP is represented by a discrete flat vector field. Along one axis are words, the meaning of which describes the conditions for the execution of the stages of the plan, and along the other axis are the numbers of the stages of the plan. First, we will consider the essence of the planning procedure using the example of the dynamics of a vector field (how the vector field changes after replanning when an event occurs due to the word w*). Let, for the considered example (Figure 1), after the execution of the stage of the plan represented by the word E (the appearance of the event *E*), the vector field corresponding to AP has the form (18).

$$\mathbf{cf}_{AP} = \begin{pmatrix} cf(A,0)=0 & cf(A,+1)=0 & cf(A,+2)=0 & cf(A,+3)=0 \\ & & \ldots & \\ cf(E,0)=0.9 & cf(E,+1)=0 & cf(E,+2)=0 & cf(E,+3)=0 \\ cf(F,0)=0 & cf(F,+1)=0.8 & cf(F,+2)=0 & cf(F,+3)=0 \\ cf(G,0)=0 & cf(G,+1)=0 & cf(G,+2)=0.8 & cf(G,+3)=0 \\ cf(H,0)=0 & cf(H,+1)=0 & cf(H,+2)=0 & cf(H,+3)=0 \\ cf(\mathrm{I},0)=0 & cf(\mathrm{I},+1)=0 & cf(\mathrm{I},+2)=0 & cf(\mathrm{I},+3)=1.0 \end{pmatrix}. \quad (18)$$

The first column *(q = 0)* of matrix (18) indicates the CF $cf(E,0)=0.9$ of the completion of the plan stage *E*, the value of which is calculated in AE. Based on its value and an externally set goal $cf(I,+3)=+1$, DPE calculates CFs for the intermediate stages of the plan $cf(F,+1)=0.8$ and $cf(G,+2)=0.8$.These AP characteristics are then used in the FLS's decision on the action required to carry out the next *F* stage of the plan. After the completion of this *F* stage, the AP given in (19) and graphically shown in Figure 3 will be created.

$$\mathbf{cf}_{AP} = \begin{pmatrix} cf(A,0)=0 & cf(A,+1)=0 & cf(A,+2)=0 \\ & & \ldots \\ cf(E,0)=0 & cf(E,+1)=0 & cf(E,+2)=0 \\ cf(F,0)=0.9 & cf(F,+1)=0 & cf(F,+2)=0 \\ cf(G,0)=0 & cf(G,+1)=0.85 & cf(G,+2)=0 \\ cf(H,0)=0 & cf(H,+1)=0 & cf(H,+2)=0 \\ cf(\mathrm{I},0)=0 & cf(\mathrm{I},+1)=0 & cf(\mathrm{I},+2)=1.0 \end{pmatrix}. \quad (19)$$

The planning algorithm implemented by DPE "shifts" the plan stages in the vector field space in the direction of decreasing *q* (decreasing the ordinal numbers of the plan stages) and recalculating the fuzzy characteristics of the vector field considering the following factors: the state of the plan execution (the last completed stage), the presence in the KC of a fragment of the plan linking the

current stage with the target. The priority is to continue the execution of the current AP: if it is possible to use it to achieve the goal, then new options for EPFs are not considered.

Below is the DPE algorithm, which considers the listed factors.

1. Calculation of CF, characterizing the attainability of the goal (w_{i**}) from the current state of the plan execution (w_{i*}) in accordance with AP.

$$cf* = \underset{i=i*,\ldots,i**}{MIN}(cf^2_{AP}(w_i)\cdot cf(w_i)). \tag{20}$$

where $cf^2_{AP}(w_i)$ is the second projection (11) of the vector field AP, $cf(w_r)$ is CF of the word describing the conditions for the implementation of the *i*th stage of the plan.

2. Correction of the AP when the condition $cf* \geq 0$ is satisfied.

2.1. Updating a fragment $PL(w_{i*}, w_{i**})$ of the current plan PL_h.

2.2. Modeling the propagation of decaying activity from the word w_{i*} to the target w_{i**}.

$$\begin{aligned}
&\mathbf{cf}'_{AP}(0) = cf(w_{i*})\cdot \mathbf{cf}_{AP}(0),\\
&\mathbf{cf}'_{AP}(j) = \exp(-\nu)\cdot cf'^1_{AP}(j-1)\cdot \mathbf{cf}_{AP}(j),\ j=1,\ldots,m-1,.\\
&\mathbf{cf}'_{AP}(m) = cf(w_{**})\cdot \mathbf{cf}_{AP}(m),
\end{aligned} \tag{21}$$

where $\mathbf{cf}_{AP}(j)$ is the *j*th column-vector of the $\mathbf{cf}_{AP}$ matrix (18), *m* is the number of columns in the $\mathbf{cf}_{AP}$ matrix, $cf'^1_{AP}(j)$ is the first projection (11) of the *j*th column of the $\mathbf{cf}_{AP}$ matrix.

2.3. Modeling the propagation of decaying activity from the target w_{i**} to the word w_{i*}.

$$\begin{aligned}
&\mathbf{cf}''_{AP}(m) = cf(w_{i**})\cdot \mathbf{cf}_{AP}(m),\\
&\mathbf{cf}''_{AP}(j) = \exp(-\nu)\cdot cf''^1_{AP}(j+1)\cdot \mathbf{cf}_{AP}(j),\ j=m-1,\ldots,1.\\
&\mathbf{cf}''_{AP}(0) = cf(w_{i*})\cdot \mathbf{cf}_{AP}(0).
\end{aligned} \tag{22}$$

2.4. Calculation of CFs of the vector field elements by modeling the coherent interaction of two counter streams of propagation of decaying activities (21) and (22).

$$\mathbf{cf}_{AP} = \frac{\mathbf{cf}'_{AP} + \mathbf{cf}''_{AP}}{2}. \tag{23}$$

3. Replanning when the condition $cf* < 0$ is met.

3.1. Finding a subset of KC's slices that contain fragment $PL(w_{i*}, w_{i**})$.

$$\mathbf{Plan} = \{n,\ if\ \ MIN(cf'(n),\ cf''(n)) > e\}_{n=1,N}, \tag{24}$$

where

$$cf'(n) = cf^2_{PL_n}(w_{i*})\cdot cf(w_{i*});\quad cf''(n) = cf^2_{PL_n}(w_{i**})\cdot cf(w_{i**}).$$

3.2. Finding a subset of plans **Plan*** from the set **Plan** in (24) for which it is possible to achieve the goal.

$$\mathbf{Plan}* = \{n,\ if\ \ \underset{l=i*,\ldots,i**}{MIN}(cf^2_{PL_n}(w_l)\cdot cf(w_l)) > e\}_{n\in \mathbf{Plan}}, \tag{25}$$

3.3. Finding the plan *n** from the set **Plan*** in (25) for which EF coincides to the greatest extent with CPF of AP.

$$cf_{EF\_CSP_{n*}} = \underset{n\in \mathbf{Plan}*}{MAX}(cf_{EF\_CSP_n}). \tag{26}$$

3.4. Execution of items 2.1-2.4 of this algorithm for the $PL(w_{i*}, w_{i**})$ fragment of the plan PL_{n*} belonging to the *n** slice of the KC.

4. End of the DPE algorithm.

3. Method of using STM and LTM data in FLS

The method of using AE, STM and LTM data in FLS for IM control is considered on the example of the transportation and loading / unloading of containers problem. A graphical representation of possible plans is shown in **Error! Reference source not found.**.**Error! Reference source not found.** The STM and LTM models are integrated with both the AE and FLS models. This is done by using the values of CF words calculated in AE as input numerical variables STM and LTM, and the output variables STM (CF footprint) and LTM (CF actual plan) are used as input numerical variables FLS

(Figure 2). In addition, the AE vocabulary includes a set of words that are used to describe events in the STM and plan stages in LTM. Let, for the example under consideration, the symbolic designation of words be their sign model and the stages descriptors of the plan. For example, in Figure 1, this is the following dictionary **W** = {A, B, C, D, E, F, G, H, I}. AE calculates the meaning of each of these words (*cf* value) based on the received data from the sensors. For example, the meaning of the word B can be formulated as follows: "IM is in position B in a state ready for dispatch, the container is loaded". AE calculates the numerical value of *cf* as the degree of correspondence of the received data from the sensors to the meaning of this word formulated above. If the data is complete (received from all sensors) and fully correspond to the meaning of the word, then the *cf* value will be close to *+1*. If the data from the sensors is complete, but they do not match the meaning of the word, for example, the container is not loaded, then *cf* will be close to *−1*. And, if there is no fresh data, then *cf* will be close to *0*.

Thus, the problem of matching FLS with AE, STM and LTM is overcome by unifying the representation of their input and output variables. The CF values generated in AE are input numeric variables for all three mechanisms FLS, STM and LTM. If now traditional for FLS method define linguistic variables (LV) on the CF universe $-1 \le cf \le +1$, then the logic of solving the problem will be based on the data of AE, STM and LTM. Below are five fuzzy rules that illustrate the different uses of this data. The rules recommend solutions in different situations when the execution of stage F of the plan has been completed (**Error! Reference source not found.**). An example is considered for two plans $PL1 = \{B^0, C^{+1}, E^{+2}, F^{+3}, G^{+4}, I^{+5}\}$, $PL2 = \{A^0, C^{+1}, D^{+2}, F^{+3}, H^{+4}, I^{+5}\}$, which are presented in the form (16).

R_1: **IF** *event(F)* **and**
CF_1 is HIGH **and**
CF_PL_1 is HIGH
THEN *CF_ Continue* is HIGH

R_2: **IF** *event(F)* **and**
CF_G_1 is LOW **and**
CF_H_1 is HIGH **and**
CF_PL_1 is HIGH
THEN *CF_Move* is H

R_3: **IF** *event(F)* **and**
CF_G_1 is LOW **and**
CF_H_1 is LOW
THEN *CF_Move* is NO

R_4: **IF** *event(F)* **and**
CF_F_0 is HIGH **and**
CF_G_1 is NO **and**
CF_H_1 is NO **and**
CF_I_2 is HIGH **and**
CF_PL_1 is HIGH
THEN *CF_Move* is G

R_5: **IF** *event(F)* **and**
CF_F_0 is HIGH **and**
CF_H_1 is HIGH **and**
CF_PL_2 is HIGH
THEN *CF_Move* is H (27)

Rules (27) in the *IF* field have the term *event(F)*. This means that the rule is processed by the FLS when the event occurred: the CF of the word *F* in the current step of AE data processing is $cf(F) > e$, and in the previous step this condition was not met. If this event does not occur at the current step, then this rule is disabled from the FLS processing (in AI classical production systems this rule is struck out). This technique significantly reduces the number of rules to be processed and, as a result, significantly reduces the amount and time of calculations, which is critical for the IM. In rules (27) *CF_xx_N* these are the names of input LVs, for which the definition of three terms LOW, NO and

HIGH with trapezoidal and triangular accessory functions is given on the CF universe, as shown in Table 1.

Table 1

Membership function parameters of linguistic variables *CF_F, CF_PR₁, CF_PR₂*

Linguistic Variables	LOW	NO	HIGH
CF_xx_N	−1.0, −1.0, −0.75, −0.25	−0.75, −0.25, 0.25, 0.75	0.0, +0.4, +1.0, +1.0

The output LV *CF_Move* is set by three singletons H, G, NO, the meaning of which is as follows: "go to stage H" "go to stage G" and "wait", respectively.

LVs *CF_PL1*, *CF_PL2* fuzzy estimate the correspondence of EF to the executed fragments of plans *PL1* and *PL2*, respectively. The term HIGH has the meaning "EF corresponds to the plan." The terms NO and LOW have the meaning, respectively, "there is not enough data in EF about the events of interest" and "EF does not correspond to a plan". The input numerical variables of these LVs are cf_{EF_PL}, which are calculated in STM by formula (12). In the rules, in contrast to the previously considered version (27), it is possible to use not the cf_{EF_PL} characteristic for the whole plan fragment, but the fuzzy characteristics of individual STM data. For example, if in R_1 the sequence in which the containers were loaded is not important, then it is possible to specify independently the certainty that IM was once loaded at positions B and E. To do this, it is sufficient to use independently of one another two LVs *CF_Load_in_B* and *CF_Load_in_E*.

In a similar way, rules (27) use LTM data. These are LV *CF_1*, *CF_F_0*, *CF_G_1*, *CF_H_1*, *CF_I_2*, the last character in the name of which indicates the sequence number of the stage of the current plan. These LVs are also defined by three terms as shown in Table 1. These LVs are fuzzy characteristics of AP. For example, at the time of completion of the execution of stage F of the plan PL1, the current plan, according to (15) and Figure 2, will have the form $PL(F,I)=\{F^{+0}, G^{+1}, I^{+2}\}$. From the list of its fuzzy characteristics (19), rules (27) used $cf_{AP}(F,0)$, $cf_{AP}(G,1)$, $cf_{AP}(I,2)$ and the first projection $cf^1_{AP}(1)$ according to (11).

Rules (27) represent knowledge about control in various situations when stage F is completed. Rule R_1 for a normal situation, when stage *F* of plan *PL1* is completed and there are no obstacles, recommends continuing the same actions (*CF_ Continue* is HIGH) when performing the next stage. It should be emphasized that both conditions (*CF_1* is HIGH and *CF_PL_1* is HIGH) of rules (27) are formed by the DPE algorithm when calculating CF, which characterizes the attainability of the goal from the current state of plan execution (20). Another use case for data from STM and LTM is given in rule R_2, which generalizes a situation that, for some reason, does not meet the necessary conditions for the next stage of the plan (*CF_G_1* is LOW). This rule is based on the results of the DPE algorithm, which replaned and generated a new actual plan based on *PL2*. The new AP has the form $PL(F,I)=\{F^{+0}, H^{+1}, I^{+2}\}$. Therefore, rule R_2 checks the possibility of achieving the goal according to the modified plan, when the next stage is not *G*, but *H* (*CF_H_1* is HIGH) and an action (*CF_Move* is H) is recommended to perform this stage. Rule R_3 does not recommend continuing the execution of the stages of either plan *PL1* or plan *PL2*, but recommends to suspend actions (*CF_Move* is NO) when the conditions for any of the stages (*CF_G_1* is LOW and *CF_H_1* is LOW) are not met. Rule R_4 illustrates another option where a risky solution is recommended. There is no complete commitment to achieving a goal. In the absence of "fresh" data, AE calculated the meaning of the situation in the form (*CF_F_0* is HIGH and *CF_G_1* is NO and *CF_H_1* is NO). But at the same time, the final goal remains relevant (*CF_I_2* is HIGH) and it is possible to complete the next stage *I* after the nearest target stages *G* or *H*. The rule recommends to perform the stage (*CF_Move* is G) about which there is no reliable information, when plan *PL1* is relevant *CF_PL_1* is HIGH.

In conclusion, we will discuss several numerical examples of using STM and LTM for FLS-based DM. The IM movement according to plan *PL1* and its service at stages *B*, *E* was simulated. Numerical simulation results are given for three cases, when IM did the stages *C*, *E* and *F* of the plan. The AE, STM and LTM data used to make decisions at stages are shown in Table II in the format $<cf(w_i,0)$; $(cf(w_i,-1), cf(w_i,-2), cf(w_i,-3), cf(w_i,-4))$; $cf^2(w_i)>$ for each event w_i for three stages (stage C, stage E, stage F). The initial STM and LTM data $cf(w_i, 0)$ are coming from AE.

In the first example, the values $cf(w_i, 0) = 0.9$ are used for all events w_i and all positions. The influence of the initial data fuzziness on DM is considered in the second and third examples. The following values of STM parameters were used in the simulation: $m=4$, $v=0.1$, $e=0.75$. Calculations of cf_{EF_PLi} were performed according to (12) for $m^* = 3$. Table II, the Stage *C* column indicates the data that was generated when IM has completed stage *B* and is ready to perform stage *C*. The first digit in each line in italics is the value of $cf(w_i, 0)$, which were calculated by AE based on data from the sensors. At this point, EF contains data (four numbers in parentheses in each row of Table II) that previously there was only one event associated with loading IM at stage *B* $cf(B, -1) = 0.8$.

Table 2

Modeling Results of the IM movement according to plan PL1

w/PL	***Stage C***	***Stage E***	***Stage F***
A	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*
B	*0*;(0.8,0,0,0);*0.8*	*0*; (0,0.7,0,0); *0.7*	*0*; (0,0,0.6,0); *0.6*
C	*−0.9*; (0,0,0,0);*0*	*0*; (−0.8,0,0,0); *−0.8*	*0*; (0, −0.7,0,0); *−0.7*
D	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*
E	*0*; (0,0,0,0); *0*	*0.9*; (0,0,0,0); *0*	*0*; (0.8,0,0,0); *0.8*
F	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0.9*; (0,0,0,0); *0*
G	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*
H	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*
I	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*	*0*; (0,0,0,0); *0*
PL1	cf^1 = (0, 0, 0, 0)	cf^1 = (0, 0, 0, 0)	cf^1 =(0.8, 0.7, 0.6, 0)
	cf_{PL1} = 0	cf_{PL1} = 0	cf_{PL1} = 0.7
PL2	cf^1 = (0, 0, 0, 0)	cf^1 = (0, 0, 0, 0)	cf^1 = (0, 0.7, 0, 0)
	cf_{PL2} = 0	cf_{PL2} = 0	cf_{PL2} = 0.23

Based on these data, the following were calculated:

1. EF parameters (data of the Stage E column of Table II) according to the FBA
2. The second projections of the vector field for all events $cf^2(w_i)$ (the last digit in the Stage C column, highlighted in italics in each row of Table II) according to (11)
3. The first projections $cf^1(q)$ of the vector field for all q for both plans *PL1*, *PL2* (indicated in the lower part of Table II) according to (11)
4. The CF that the EF corresponds to plans (indicated in the lower part of Table II) according to (12)

The obtained data were used as the values of the FLS input numerical variables, namely, $cf(F,0) = 0$, $cf_{PL1} = 0$, $cf_{PL2} = 0$. In this experiment, the value of the input numerical variable of the *CF_G_1* was $cf(G,+1) = 0$. The results of FLS fuzzy inference, on the base of five rules (27), gave $cf_{MOVE\_} = 0$, which corresponds to *CF_MOVE* = NO (stay in place). A similar result was obtained for stage *E* $cf_{MOVE\_} = 0$ (in Table II $cf_{PL1} = 0$, $cf_{PL2} = 0$ and $cf(G,+1) = 0$). For stage *F* the following values are calculated $cf_{PL1} = 0.7$, $cf_{PL2} = 0.23$ and $cf(G,+1) = 0.85$ on the basis of which the FLS received a decision $cf_{MOVE\_}$ = 0.96, which corresponds to *CF_MOVE* = *G* (start stage *G*). Analysis of data for stage *F* shows that the solution recommended by IM is rational for the simulated situation.

In the second example, the study of the information incompleteness influence on DM at the same stage *F* as before was carried out. At the same time, a situation was simulated when data from sensors do not allow AE accurately to determine the stage descriptor. Two cases were considered. In the first case, the IM perception system with certainty *cf(E)* confirmed that this is descriptor of stage *E* and with certainty *cf(D)* that this is descriptor of stage *D*. In fact, it was a description of stage *E*. In the second case, a similar situation was simulated for stages *A* and *B*. The calculation results, like those considered earlier for the first example, are shown in Table 3. The upper part of the table shows the data for the first case, and the lower shaded part of the table shows the data and results for the second case.

Table 3
Certainty cf_{PL} that EF matches the plan for different certainties of stage descriptor identification

cf	1	2	3	4	5	6	7	8	9
cf(E)	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
cf(D)	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
cf_{PL1}	0.7	0.67	0.63	0.6	0.57	0.53	0.5	0.47	0.43
cf_{PL2}	0.23	0.23	0.3	0.33	0.37	0.4	0.43	0.47	0.5
cf_{MOVE}	0.96	0.96	0.95	0.95	0.95	0.95	0.95	0.95	0.95
cf(A)	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
cf(B)	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
cf_{PL1}	0.7	0.67	0.63	0.6	0.57	0.53	0.5	0.5	0.5
cf_{PL2}	0.23	0.23	0.23	0.27	0.3	0.33	0.37	0.4	0.43

Based on the results shown in the table, the following conclusions can be drawn. First, the use of EF compensates for the influence of incomplete and fuzziness data on decision making confidence. The prehistory held by EF increases the certainty that EF matches the plan. As you can see from the Table 3, even with a low confidence in the identification of a actually completed stage *E* of the plan ($cf(E) = 0.3$) compared with a high confidence in the false identification of a non-completed stage $cf(D) = 0.7$, the correct plan is matched with the EF $cf_{PL1} = 0.5$, $cf_{PL2} = 0.43$, $cf_{PL1} > cf_{PL2}$. This allows FLS to make the correct decision $cf_{MOVE} = 0.95$, which still corresponds to $CF\_MOVE = G$.

The second conclusion is that the EF elements aging mechanism introduced into the STM model supports such a property of short-term memory as the dependence of confidence in decision making on the remoteness of an event in memory. From Table 3 shows that two events that occurred at different times have different effects on cf_{PL}. It is enough to compare $_{cf}PL2$ in one column of the table, for example, for $cf(E) = cf(B) = 0.7$ for event *E*, for which $q = -1$, $cf_{PL2} = 0.3$, and for event B, for which $q = -3$, $cf_{PL2} = 0.23$.

In the future, it is planned to investigate the dependence of the certainty factor of the decision made in FLS on the STM and LTM parameters, in particular, memory depth *q*, aging rate *v*, data incompleteness *cf(w)*, length of the plan *Q* and the dimension of the knowledge cube *N*.

4. Conclusion

The proposed method expands the possibilities of using FLS in the IM. The IM application requiring a complex solution of three tasks, the perception of data from sensors, planning of actions and DM in accordance with the actual plan and the process's history can be implemented as FLS. This was made possible by moving away from a task based on data from many sensors to a task based on a few words. The AE matched the spatio-temporal sequence of data from sensors into the meaning of the words. This capability is supported by a three-stage processing data technology based on the integration of three models: generalization of data from sensors using AE, EF processing in STM, dynamic planning of the actions in LTM and FLS. Computer experiments using three-stage processing data technology in the IM DM showed:

1. With a significant reduction in the problem dimension, the decisions made by FLS based on input data from AE and STM, which represent fuzzy characteristics of high-level generalization data, are rational for the simulated situation.
2. The data aging rate parameter introduced into the STM model allows, at the level of FLS rules, firstly, to compensate for the influence of incompleteness and fuzziness of data from sensors on confidence in DM, and secondly, to take into account the dependence of confidence in DM on the distance of an event in memory.
3. The introduced DPE, based on the LTM data model, performs dynamic replanning of the action plan stages, taking into account the state of the environment, which makes it possible to implement situational control of the IM at the level of the FLS rules.

5. References

[1] S. Raghavan, 2020 AI Predictions from IBM Research, 2020. URL: https://www.ibm.com/blogs/research/2019/12/2020-ai-predictions/.

[2] J. Ploennigs, J. Cohn, A. Stanford-Clark, A The future of IoT, IEEE Internet Things M, vol 1(1), (2018) 28-33.

[3] Internet of Things: Understanding the Adventure. E-Book. SAS Institute Inc., 2020. URL: https://www.sas.com/sas/offers/20/iot-understanding-adventure.html.

[4] J. Perez, F. Deligianni, D. Ravi, G. Yang, Artificial Intelligence and Robotics, UK-RAS White Papers, 2020. URL: https://www.ukras.org/wp-content/uploads/2018/09/UK_RAS_wp_AI_web.pdf

[5] S. P. Meyn, R.L. Tweedie, Markov Models, in: Markov Chains and Stochastic Stability. Communications and Control Engineering Series. Springer, London, 1993, doi: 10.1007/978-1-4471-3267-7_2.

[6] S.J. Russell, P. Norvig, Artificial Intelligence. A Modern Approach (3rd ed), Upper Saddle River, NJ, USA: Pearson Education, 2010.

[7] V. Pizurica, The Waylay engine, Part 1: One rules engine to rule them all, 2017. URL: https://blog.waylay.io/waylay-engine-one-rules-engine-to-rule-them-all/.

[8] A. Kargin, T. Petrenko, Internet of Things Smart Rules Engine, in: Proc. 2018 Int. Sci.-Pract. Conf. Probl. Infocommun. Sci. and Technol. (PIC S&T 2018), Kharkiv, Ukraine, 2018, pp. 639-644. doi: 10.1109/INFOCOMMST.2018.8632027.

[9] AWS IoT Developer Guide. Rules for AWS IoT, 2019. URL: https://docs.aws.amazon.com/iot/latest/developerguide/iot-dg.pdf#iot-rules.

[10] How To Choose a Rules Engine: Seven Things to Look at When Automating for IoT, Waylay Whitepaper, 2019. URL: https://www.waylay.io/download-how-to-choose-a-rules-engine.html.

[11] A. Piegat, Fuzzy modelling and control, Heidelberg: Physica-Verlag Heidelberg, New York, 2001.

[12] A. Kargin, T. Petrenko, Spatio-Temporal Data Interpretation Based on Perceptional Model, in: Advances in Spatio-Temporal Segmentation of Visual Data. Studies in Computational Intelligence, V. Mashtalir, I. Ruban, V. Levashenko, Eds., vol. 876, Springer, Cham, 2020, pp. 101-159. doi: 10.1007/978-3-030-35480-0_3.

[13] A. Kargin, T. Petrenko, Multi-level Computing With Words Model to Autonomous Systems Control, in: Proc. 9th Int. Conf. Inf. Control Sys.&Tech (ICST-2020), A. Pakštas, T. Hovorushchenko, V. Vychuzhanin, H. Yin, N. Rudnichenko. Eds. Odessa, Ukraine, 2020, CEUR Workshop Proceedings, vol. 2711, pp. 16-30. URL: http://ceur-ws.org/Vol-2711/.

[14] J. Hendler, A. Tate, M. Drummond, AI Planning: Systems and Technique, AI Magazine 11(2), (1990) 61-77. doi: 10.1609/aimag.v11i2.833.

[15] D. S. Nau, Current Trends in Automated Planning, AI Magazine 28(4), (2007) 43-58.

[16] R. Alford, V. Shivashankar, M. Roberts, J. Frank, D. W. Aha, Hierarchical Planning: Relating Task and Goal Decomposition with Task Sharing, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), 2016, pp. 3022-3028. URL: https://www.ijcai.org/Proceedings/16/Papers/429.pdf.

[17] J. Ferrer-Mestres, G. Frances, H. Geffner, Combined Task and Motion Planning as Classical AI Planning, 2017. URL: https://arxiv.org/pdf/1706.06927.pdf.

[18] I. Georgievski, M. Aiello, HTN planning: Overview, comparison, and beyond, Artificial Intelligence 222 (2015) 124–156. doi: 10.1016/j.artint.2015.02.002.

[19] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, J. P. How, Real-time Motion Planning with Applications to Autonomous Urban Driving. IEEE Transactions on Control Systems Technology, vol. 17, no.5 (2009), 1105-1118. doi: 10.1109/TCST.2008.2012116.

[20] M. Elbanhawi, M. Simic, R. Jazar, Autonomous Robots Path Planning: An Adaptive Roadmap Approach, Applied Mechanics and Materials, vols. 373-375 (2013) 246-254. doi:10.4028/www.scientific.net/AMM.373-375.246.

[21] F. Peralta, M. Arzamendia, D. Gregor, D. G. Reina, S. Toral, A Comparison of Local Path Planning Techniques of Autonomous Surface Vehicles for Monitoring Applications: The Ypacarai Lake Case-study, Sensors 20(5), 1488 (2020). doi:10.3390/s20051488.
[22] M. Leonetti, L. Iocchi, P. Stone, A synthesis of automated planning and reinforcement learning for efficient, robust decision-making, Artificial Intelligence 241(1) (2016) 103–130. doi: 10.1016/j.artint.2016.07.004.
[23] K. Hammond, Case-based planning as a memory task, Academic Press, New York, 1989.
[24] W. A. Abdulhafiz, A. Khamis, Handling Data Uncertainty and Inconsistency Using Multisensor Data Fusion, Advances in Artificial Intelligence, vol. 2013, Art. no. 241260, (2013). doi: 10.1155/2013/241260.
[25] X. Zhang, M. Zhao, R. Dong, Time-Series Prediction of Environmental Noise for Urban IoT Based on Long Short-Term Memory Recurrent Neural Network, Applied Sciences 10(3):1144, (2020). doi: 10.3390/app10031144.
[26] R. M. Solso, O.H. MacLin, M. K. MacLin, Cognitive Psychology (7th ed), Allyn & Bacon, 2004.
[27] J.R. Anderson, Cognitive Psychology and Its Implications (7th ed), Worth Publishers, 2009.
[28] R.M. Shiffrin, Short-term store: The basis for a memory system, in: Cognitive theory F. Restle, R. M. Shiffrin, N. J. Castellan, H.R. Lindman, D. B. Pisoni, Eds., vol.1, Hillsdale, NJ: Erlbaum., (2018) 193–218. doi: 10.4324/9780203781548-13.
[29] D. Norris, Short-term memory and long-term memory are still different, Psychological Bulletin, 143(9) (2017) 992-1009. doi: 10.1037/bul0000108.
[30] G. Plancher, P. Barrouillet, On some of the main criticisms of the modal model: Reappraisal from a TBRS perspective, Memory & Cognition, 48 (2020) 455–468. doi:10.3758/s13421-019-00982-w.
[31] S. Hochreiter, J. Schmidhuber, Long short-term memory, in: Neural Computation, MIT, vol.9(8) (1997) 1735-1780. doi: 10.1162/neco.1997.9.8.1735.
[32] L. Zadeh, Computing with words principal concepts and ideas, Studies in Fuzziness and Soft Computing (277). Springer-Verlag, Berlin Heidelberg, 2012.
[33] J. Mendel, D. Wu, Perceptual Computing: Adding People in Making Subjective Judgments, John Wiley & Suns, Inc., 2010.
[34] A. Kaufmann, Introduction to the theory of fuzzy subsets (1th ed), Academic Pr, 1975.