Frame Technology Demonstrator

FRAMES

Eduard Burian, MSc.
LOX Technologies, Bratislava, Slovakia, eburian@zoznam.sk

Draft of a project proposal

Proposal Abstract

With *frames* we understand encapsulation of deliberate information structure, e.g. information object consisting of data and methods for their manipulation, with universal, explicit, semantic-based interface to the encapsulated information. In a narrow sense, frame is abstraction over all possible interfaces offered by encapsulated objects, thus, through the universal interface, encapsulated information structure, data and its methods are intelligible and accessible in a universally defined manner and without any supplementary information. Advent of frames can have huge impact on several aspects of computing in the very general sense: a. because the universality of frame interface, dependable networks of framed data can be much easily established; b. library of true universal algorithms, unique even in their binary form, can be much easily provided and evaluated for trustworthiness; c. frame-based communication over diverse systems allows automated semantics distillation and thus overrides the necessity of accurate protocol formats and guaranties the persistence of information; d. both human and machine intelligibility of information encapsulated in frame can lead to insightful development tools that allow developer's access to any information at any time at one side, and automated, intelligent processing of framed data even of previously unknown structure, at the other. The last aspect, with possibility of virtually deliberate interactions between frames, is a novelty in the computing: we hope, with the novel interactivity within deliberate data, to have, at last, a mechanisms that allows machine cognitive functions evolving beyond the scope pre-programmed by human developer.

Objectives

- Define and implement Generic Frames as encapsulation for virtually deliberate OOP object that embodies universal, explicit, semantic-based and machine-intelligible interface to object's data and corresponding data manipulation methods, in C++, portable to diverse hardware platforms.
- Find and implement closed, recursive set of framed data types – the frame space – that covers any possible outcome of manipulating, interacting or grouping frames, and link the frame types in a logically consistent taxonomy (horizontal inheritance within the frame space).
- Implement mechanisms for channel-based interactions of frames that will allow construction of dependable networks of framed data with embedded resolution for conflicts and deadlocks. Provide an example application with dependable network of framed data.
- Deploy scalable Frame Explorer for Windows capable of constructing and monitoring frames due to user needs, and of manipulation with framed data through intuitive graphic user interface.
- Create extensive algorithm library for framed data with focus in cognitive systems and robotic-specific tasks (digital signal processing, image and video processing, pattern recognition, inverse kinematics, spatial orientation and attitude, optimization, AI algorithms, genetic programming, etc.)
- As demonstration of frame technology capabilities, implement universal Fourier-based correlator as a key algorithm behind multidimensional pattern recognition for deliberate data. Provide example applications of the correlator for diverse target data.

Background

Concept of frames

The key idea behind the concept of frames is machine intelligibility of structured data and methods for their manipulation, in order to achieve an effective automated analysis of data, as well as further automated synthesis of data processing methods.

As for the standard paradigms of object-oriented programming, objects are structures of data and methods for manipulating them. However, a binary representation of an object does not contain semantic information about the data types, their structure, and meaning of the various methods. Semantic information about an object is kept only within the programming language, so it is intelligible only for human programmer, who is fluent in that language, or, eventually, for a programming-support tool within some integrated development environments, serving as a syntax expert.

With the advent of COM programming technique developed by Microsoft, some portion of the semantic information about the object composition is provided within the system to ensure process inter-operability. However, an interface to the object's methods is still an abstract class, which, in its binary form, does not provide any meaning about the class it refers to. To access the methods, the programmer has to include interface declaration in the corresponding programming language. In other words, even the object "knows" how to process the data, using its implicit interface, he cannot tell it – it has no explicit, intelligible interface.

The recently released .NET framework supports semantic information about portions of the code in metadata, with the aim to ensure correct interpretation of arguments in methods originating in different programming languages. Even this seems to be a step in the right direction, .NET lacks a clear, fundamental definition of semantic-enabled data types that can lead to effective automated data analysis and synthesis of procedures.

Despite all the progress in programming techniques, the call for a technique that enables manipulation of virtually any data type structured to deliberate level of complexity, with access to meaning of the structured data, provided by the data itself, was not been resolved so far. Here is proposed a way to achieve this goal: through data encapsulation within one, all-present class, integrating access to unlimited complexity of the data structures, as well as an explicit interface to the data manipulation methods. This encapsulation class is then a *frame*.

(The concept of frame has its special meaning within Artificial Intelligence, as it has perhaps in any scientific area or programming tool. This however does not correspond to our understanding of this concept.)

Thus, in a frame-based system, any data, i.e. *any system-wide obtainable information*, is, or can be provided with its frame. The frame provides, being asked for, information about the data composition, length, history, localization, connections to other data, limitations... then possible methods for manipulation, types of method's arguments, return types... then methods being used to connect data with other data... etc. etc. Clearly, there are types of data that are fundamental, unable of de-structuralization, i.e. bits, bytes, integers, etc.: these were atom frames. *Set*, *vector*, *array*, *image* are clear candidates for structured frames consisting of atoms or other frames. The structuralization can have no limits: Sets of atoms can become elements of other sets, these can become elements of vectors... Because frame-based algorithms strictly follow object-oriented programming rules to relocate any internal action to the objects itself, a vector addition or multiplication can be provided even these are Vectors with never considered structure and composition: the addition or multiplication methods will be grabbed from the frames of vector's elements, then from the elements of those frames... etc., up to atomic elements.

The holy grail of the frame philosophy can be identified as a *binary* library of algorithms, where an abstract algorithm for processing abstract data entities has its *unique* real-world implementation. Or, the count of abstract algorithms and the count of algorithm implementations in the frame-based library is the same number. What a possibility! Consider it again: one correct-written algorithm can be reliably used to process *any type* of data that is meaningful to process, providing the data has its frame.

And this is where frames end and – in our opinion – the real Artificial Intelligence begins. Frames might be 'only' a tool for AI – a tool that enables the unlimited flexibility in operating and manipulating heterogeneous, never-thought sets of data, that enables its processing, mutual comparison, further linguistic analysis, calculus, and reasoning.

Closeness of frame space

Frames offer generalization of objects that has serious implications to the mathematical structure of programs.

First, there is reduction in redundancy of data types and methods due to the fact that frame becomes a universal data type: instead of number of definite types in the function arguments, only one universal type appears.

The second is recurrence, i.e. involvement of frames within an object that is itself encapsulated with a frame.

A premise for closed frame space is a set of prototype objects that can handle frames as their data. For instance, set as a derivation of a container object would be instantiated with general frame as the element data type, and then encapsulated with a frame offering universal interface to the set's methods. Apparently, deliberately deep, dynamic structure can be achieved in such a way. The same procedure would be applied for all other derivations of sets like strings, vectors, images etc.

Another necessity is operative closeness for objects with defined arithmetic and logic operations, and, existence of metric as a measure of distance or non-similarity between objects of the same type.

Frames encapsulating the basic prototype types like sets, strings, vectors etc. offer the same hierarchy structure of the standard OOP (template library) tree, however, technically, all are of the same – frame – type. I.e. the interface to the object behind the frame is constant. Also when we refer to the frame type, the type of encapsulated object should be considered.

A rather simplified scheme of the frame space may look like:

- **atom** is a frame that encapsulates no other (sub-)frames
  - atoms are characters, integer and float numbers, and integer vector as a coordinate of a field, with methods involving limitations, arithmetics, metric, mutual conversion etc.
- **set** is a record of frames generally of different types
  - **swarm** is set of the same frame type
    - all methods for the given frame type the swarm is made of are accessible for the whole swarm
    - a method is equally applied for all frames in swarm
  - **field** is two-member set of atom coordinate and a swarm of any type
    - two metrics are defined in a field: metric of the coordinate and metric of the swarm (the type of data swarm is made of)
    - **vector** is 1-dimensional field
      - **complex** is vector of two frames: real and imaginary
      - **color** is vector of three frames: red, green and blue
    - **image** is 2-dimensional field of color frames

This picture is rather illustrative than complete, nevertheless, it can be seen that complex frame features emerge quite easily from basic ones, like the field metric with two components: position and content. Or, consider that an image with red, blue and green atoms as float numbers can be converted with no effort to computer-native integer color image, having such conversion method prepared in float atoms. We could expect similar emergences and shortcuts in other complex frame structures, if the basic building blocks are prepared thorough enough.

The quest for machine intelligibility

Is there a possibility for a pattern-recognition system that can handle virtually any type of data without apriori, pre-programmed information?

Consider an example frame-based algorithm build with only premise: it expects field frame at the input.

First, the algorithm interrogates the input field for number of dimensions. A text source can be 1-dimensional field, as well as sound. Image would be 2-dimensional.

To find a pattern in the data, algorithm has to pick up a sample, relocate and correlate it with other parts of the data, and repeat the process until confidence level is reached. It would need methods for creating subsets from data at given locations, for possible transformations given by available symmetries like rotation, mirroring, magnification etc., and then for correlation of samples in space and data value (position and color, for example). These methods are just that what the field frame of given dimension and structure offers in possible non-loss transformations; the result of correlation in field subsets can be provided automatically as combination of field space and data metrics. Thus, so far, the algorithm can rely on methods supplied by the input frames only.

If patterns occur, correlations are significant in the similar samples, but the level of significance and thresholds for pattern classification depend on many aspects of original data; e.g. number of distinguishable patterns, variations, statistical soundness, noise floor etc. Nevertheless, even this dependence can be universally treated on purely mathematical basis and made part of the universal pattern search algorithm.

To achieve this type of automated, context-aware data manipulation, one has to attach all the structure- and semantic-related information, which usually is only "thought", into the data itself, i.e., to its frame. This is mere the reason why we need possibly all aspects about the data made machine-intelligible.

Application fields

Linguistic analysis of data

Because environment is patterned, symbolic representation of incoming data regularly leads to lower memory load and greater efficiency in usage of data transmission channels of fixed throughput. However, searching for patterns in the data has its price in larger needed computational resources. Nevertheless, any mature communication and audiovisual technology today uses a sort of pattern-recognition, including vocoders in mobile phones, or video players: as far as price for MIPS slips faster than price for communication channels or non-volatile memories, the data compression has its pay-off.

Frames can supply an important technological advantage for intrinsic linguistic analysis of arbitrary data over state-of-the-art systems. Algorithms for frame-based multidimensional pattern recognition can be developed independent on the particular data streams that were processed, a technology leap that liberates software developers of routine tasks while concentrating them to real AI problems.

In particular, image processing in frame domain will supply abstract image decomposition, and further decomposition of the found patterns, up-to a certain level of lowest attainable entropy of the symbolic representation, which, if our idea is correct, resembles natural-language description of the scene. Finding correspondence between patterns and symbols in the representation, the system can automatically derive reverse transcriptions from symbols into image; then, manipulation of a specific pattern on image can be achieved by manipulation of its symbol and following reverse transcription. Because symbolic representation is far less expensive in machine resource terms, optimization tasks with symbolic representation of image data may be far more effective than any direct processing. For example "swap blue and red triangle" can be done far more effectively, when image is represented as chain of symbols for geometric shapes with given colors, where symbols "triangle red" and "triangle blue" will be simply mutually swapped, over an sophisticated direct processing of image data specially programmed for this operation.

Because of the fundamental changes in software paradigms with frame-assisted data processing, we can only guess what further benefits will be, or, where obstacles can lie. What we aim now is attainability of the frame-based tools and ignition of further research in this field.

Signal processing and embedded world

Any signal refers to a measured or modeled physical quantity, therefore, many supplemental information is necessary to provide the actual context for signal data. Signal dimension and unit, absolute minimum and maximum as well as minima and maxima values recorded in a given period of time, the sampling and quantization regularity, particularly the sampling period, intercepted noise or drifts, history record etc., all that can be integrated within a signal frame, particularly a modular one, and supplied through its universal interface.

In such manner, signals of different units or sample rate can be automatically combined and converted, when appropriate. Possible (often necessary) limitations in signal value range are treated by the signal frame itself without treating it especially in the algorithm; and vice versa, an algorithm can learn the signal limitations from the machine-intelligible interface the signal frame offers to optimize its output. This may lead to improved conformance between abstract signal diagram and actual signal processing algorithm, and to graphic design tools capable of on-the-fly application of changes in the signal diagram into the signal path flow on the target device.

Frames may have a particular role in embedded system's design and test tools, as often data formats in microcontrollers and DSPs are not compatible with data in desktop PCs in which design takes place. Still, one-to-one correspondence of algorithms under tests and in real environments is of highest importance. Usually, this correspondence is found in retyping of primitives, but the underlying hardware complexity may still prepare many unpleasant surprises. Frames may offer an ultimate solution to the portability issue, provided hardware vendor supplies thoroughly tested set of frames for the particular device. But again, from the vendor side, utilizing frame technology with almost automated content extraction will cut design and testing times over manual ad-hoc testing and thus substantially improve times-to-market.

Robotics

Linking devices with unequal data formats or resolution should be no longer a problem with frames. However, the advent of frames in the robotics has quite a wider objective.

In the first instance, a frame-based system that provides information about data format, limitations, history, etc. for an arbitrary signal is by far more comfortable to monitor and control as a system that does not. Telemetry and teleoperation user interface is no longer made-to-measure to the distant system; rather, the interface is generic and grabs all the necessary information about signal types, limitation, initial values etc. within the frames of the system itself.

The second major advantage in usage of frames may be the possibility of automated semantics abstraction for complex systems (complex frames) with no explicit semantic information embedded. For example, complex serial-type robotic manipulator may have semantics defined for mutual positions of each two manipulator segments as there are motors and position sensors, however, the position and orientation of the manipulator's endpoint is too complex to be defined directly. Provided there is a sort of position sensor at the endpoint, e.g. a video camera, the system can learn the relation between multidimensional segment-position vector and results from the video camera to identify *position* and *orientation* parameters for the endpoint, and include these parameters, with their semantic, within system's signals, i.e. include methods for their acquisition and manipulation within the frame that defines robot and its environment. While acquisition is somewhat direct, the manipulation involves an inverse problem, i.e. solving optimization task in multiple dimensions for all the manipulator-segment-vector positions, but again, such optimization task for general vector frame may be included within the mathematic library and its application for the given task is straightforward. Then, a teleoperation interface where position and orientation parameters are present, can be simply linked with the robot's frame, and operator, without notifying the complexity of the problem, can operate the manipulator fully intuitively.

Extensive usage of frame-based automated semantic derivation may evolve into protocols that can connect systems without prior definition of desired connectivity. An enhanced connectivity and inter-operability between diverse systems and interfaces could be achieved, applicable well behind the areas of robotic teleoperation and telepresence: perhaps, one day, we will get a *truly universal* remote control for TV sets.

Relevance to the current research objectives

The glueless, automated coupling between diverse data types that frames offer as a consequence of the machine-intelligible interface philosophy, is not only an advantage; it is a key component – we believe – in a robotic systems comprising never-settled, ever-evolving digital audio-visual standards, communication protocols and environment interfaces – which is the vision of a household robot of the nearing future. Considering need of a flexible, dependable, self-reprogrammable robotic system, aspiring even to higher cognitive functions, an open, flexible, but unitary data-handling standard, encapsulating all the diverse data and not limited to some pre-engineered structure, is *a must*.

Frames, we believe, represent a set of tools that is virtually indispensable for any advanced AI research. The key is in offering an additional degree of abstraction, namely, abstraction over data type and structure, and enabling algorithms for framed data that are once and ever complete, in its binary form. This can lead – for example – to construction of one general analyzer of data capable of delivering machine situation awareness for any incoming sensory data, either they are visual, audio or attitude data, independent of the data composition, structure, precision, numeric types etc.

In addition, in robotics, where complexity of robot's systems comes with several diverse embedded subsystems, an abstraction over various data types used in the particular subsystems is highly affordable for purely technical reasons. We believe, concept of framed data can be well useful even in rather technical, data communication and storage issues, in telemetry, teleoperation and telepresence issues, as well as in higher-level tasks like pattern recognition, optimization and similar AI-related processing.

Potential impact

We can expect that frame-based embedded systems emerge very soon after the first Generic Frames, Frame Explorer and the other communication and manipulation tools are deployed. With the interface intelligibility that comes with frame technology, plus unitary concept of data encapsulation within one common Frame class, design and testing of embedded system will be much easier, and costs for system development will shrink rapidly – which is the factor that will surely fuel the spreading of frame-based systems.

As frames were acknowledged by the large research and industrial community, the frame technology may get an another boost, benefiting from parallel research and cross-utilization of frame-based algorithms. AI algorithms, once developed, can be easily adapted into virtually any frame-based system, because of the data type abstraction that frames offer. We imagine that state-of-the-art in artificial intelligence will be downloadable into virtually any frame-based robotic system, like the today's Internet browsers are kept up to the standards of the World Wide Web.

We can even anticipate development of frame-supporting microprocessors, like the present-day DSP processors are supporting some signal-specific tasks.

Integration of Frames into state-of-the-art information technology, operating systems and consumer devices, supported by standardization authority (world-wide-acting expert group or consortium), will effectively redefine other communication standards and data formats in order of achieving the flexibility the frame philosophy offers.