Framework for Modelling Multiple Input Complex Aggregations for Interactive Installations

Nicolas Padfield
Roskilde University / illutron
HUMTEK, Building 8.1
Universitetsvej 1
Roskilde Denmark
nicolasp@ruc.dk

Troels Andreasen
Roskilde University
CBIT, Building 43.2
Universitetsvej 1
Roskilde Denmark
troels@ruc.dk

***Abstract*—We describe a generalized framework as a method and design tool for creating interactive installations with a demand for exploratory meaning creation, not limited to the design stage, but extending into the stage where the installation meets participants and audience. The proposed solution is based on fuzzy logic and provides a method for variably balancing interaction and user input with the intention of the artist or director. An experimental design is presented, demonstrating an intuitive interface for parametric modelling of a complex aggregation function. The aggregation function unifies hierarchical, importance-weighted and ordered-weighted fuzzy averaging to provide complex combinations of user input.**

***Index Terms*—interactive installations; fuzzy logic; aggregation; parametric modelling; intuitive interface**

I. Introduction

My bedroom floor is completely open to a creative process. One can place the socks in any pattern one wishes – but is it art? Or is it just a mess? The Mona Lisa is widely acknowledged as being art. But is the Mona Lisa interactive? Only if you do as vandals did in 1956 (twice), 1974 and 2009 and throw acid, paint or tea mugs at her. Somewhere, between these two endpoints – my bedroom floor and Leonardo da Vincis famous masterpiece – lies the realm of interactive art. An interactive installation is by definition a combination of design choices taken apriori by the artist and input data generated by the audience, users or participants. Usually, the amount of control afforded the participants is predefined. There is a rigid framework, both physical reality and in software, which defines how and to what degree the participants can affect the work of art. But, what if the division between the power of the artist and the power of the participants was not set in stone, but was itself a variable that can be tweaked in real time, can be experimented with and optimised for a particular setting or particular participants?

In this paper, we describe a generalised software framework which allows an artist or designer to tweak the weighting of each individual input and of groups of inputs. We postulate that this provides three advantages over, e.g., programming the system in a traditional way: it enables non-programmer artists and designers to create complex installations, it enables complexity, and enables real time adjustment of the installation itself. By a complex system, we mean a system where it is not easy to predict the exact output even given knowledge of the input and state of the system. While many of the issues described are applicable to a wide range of interactive systems, we will in the following primarily focus on the particular case of an art installation as an example.

Ideally, we wish our installations to attract participants and be easy to start interacting with – at the same time as providing depth and complexity warranting extended exploration. A good explanation of threshold and ceiling and the desirability and difficulty of achieving both low threshold and high ceiling at once can be found in [13] "Threshold and Ceiling: The 'threshold' is how difficult it is to learn how to use the system, and the 'ceiling' is how much can be done using the system. The most successful current systems seem to be either low threshold and low ceiling, or high threshold and high ceiling. However, it remains an important challenge to find ways to achieve the very desirable outcome of systems with both a low threshold and a high ceiling at the same time." [13]

We posit that enabling complexity makes it more likely that participants will engage with the system for a longer time span, wanting to explore the possibilities, as the output is not obviously tightly coupled to the input, and the output may be diverse even given similar input.

The approach to model interactive installations described here aims at providing flexibility for the user/artist and builds on several cases of practical experience with developing installations, among which are [1], [2]. The ideas evolve from earlier concepts such as physically interactive environments [3], [4], immersive virtual environments [5], and "tangible interfaces" [6], originating in Krugers seminal ideas dating back to the 1980s on computer-controlled interactive spaces [7]. Numerous examples appear in the literature of the development of interactive environments for the general purpose of communication, in areas such as advertising, entertainment, story-telling and dissemination of cultural assets; see for example [8], [9], [4]. One of the main motivations for such developments seems to be that physically interactive environments are perceived to offer the user a greater sense of presence and immersion, allowing the user to engage more actively with the content of the communication.

 ISBN: 978-1-61208-235-6

A. Why intelligent systems

Nowadays, most audience-interactive art installations are controlled by computers. Thus, behind the scenes, there is always some program or engine which determines the behaviour of the system. Such a system can be relatively simple, with output coupled relatively directly to input or it can be very complex, with output being dependent on multiple factors including current input, past input, predefined data, meta data (data about the data), and even machine learned data derived from the interaction history itself.

Installations of low complexity can be driven by conventional deterministic procedural programs. But, as complexity increases, these dedicated programs quickly become complicated and have the disadvantages of lacking generalisability and not handling uncertainty very well. Installations that choose elements from a large set of data based on myriad user input variables, of which some may be uncertain or conflicting are good candidates for an engine based on some form of artificial intelligence (AI).

B. Artificial intelligence

While generalised intelligence (strong AI) [10] is still a long term goal for the AI community, applicable AI consists of more specialised systems that are good at a particular task. Artificial intelligence is used for data mining, process control, logistics, diagnosis and in many other areas. The AI field has developed highly successful methodologies for dealing with incomplete or uncertain information, including Bayesian logic [11] enabling probabilistic reasoning in adaptive conditional probability networks, and Artificial neural networks, which are inspired by structural and functional characteristics of biological neural networks, interconnected neurons process information using a connectionist approach.

C. Fuzzy logic

Fuzzy logic is many-valued logic that generalizes conventional Boolean logic, and enables approximate reasoning [14]. Fuzzy truth values range between the extremes 0 to 1 corresponding to "completely true" and "completely false" and Fuzzy sets elements have a degree of membership, described by a membership function, in the range [0, 1], as opposed to crisp sets where membership is bivalent. Fuzzy logic is useful handling a multitude of sensor inputs, often analogue, sometimes pointing in conflicting directions. The usefulness of fuzzy logic derives from the fact that many problems in the real world, especially ones involving human reasoning, are approximate in nature.

One hundred people drumming is not bivalently either completely in time or out of time or fast or slow. As Lotfi Asker Zadeh wrote in his seminal work *Fuzzy sets* [15]: "More often than not, the classes of objects encountered in the real physical world do not have precisely defined criteria of membership" It is no coincidence that one of fuzzy logics main applications is within industrial process control, that bears more than a passing resemblance to interactive installation process control.

II. Introducing a generalized fuzzy logic art support framework

In the following, we describe a generalized fuzzy logic framework that facilitates rapid building and experimenting with interactive art, making it feasible to easily create high complexity interactive installations. A system which provides flexible arbitration, balance between interaction and art. Our aim is to enable non technically inclined artists to model complex behavior based on multiple input. Our approach is to apply highly flexible fuzzy aggregation, more specifically hierarchical, importance weighted, ordered weighted aggregation [18], [19], and to provide an "intuitive" user interface that is easy to grasp without fully understanding the mathematical functions behind it.

Interactive art is a balance, of both the artists' wishes and the participants' actions. Pre-digitally, the artist could control *what* the participant could do. The artist might have decided that there is a welded steel frame (which it is difficult for the participant to alter), and an inviting brass handle (which is inviting and easy to turn). In the digital age, the artist can also decide and adjust *to what degree* the participant may decide. The participant may turn the handle, but at this moment that weighs in at only 0.3, while the artists' wishes weigh in at 0.7. Artificial intelligence based on fuzzy logic is a prime candidate for our use because it supports

- Adaptability – suitable for reaching decisions from a number of heterogeneous, possibly conflicting inputs.
- Narrativity – suitable for supporting narrativity, e.g., for searching a large database and selecting *which* media element to become next in a sequence (comprising a story). This is because attributes of elements will often be humanistic and lend themselves to fuzzy quantification better than to binary quantification or procedural programming; and because presuming a finite number of available elements to choose between and multiple, sometimes conflicting inputs, fuzzy is suitable for choosing the best available element.
- Live data – suitable for taking live sensor data and reaching a decision. One can define a parameterized fuzzy linguistic concept such as "rhythmicality".
- Flexibility in mixing live and predefined data. As fuzzy aggregation can handle both live and predefined data with equal ease, it is possible to use the same aggregations to combine both live and predefined data.
- Realtime adjustment – it is relatively straightforward to tune weighting parameters in real time, giving us ease of experimentation or adjusting the interaction to suit the participants.

A. Raising the interactivity to another level

An installation programmed in a procedural fashion usually lacks the ability to adjust the interaction control in real time. We would like to let the system run as designed and handle all interaction, while at the same time being able to adjust interaction parameter weightings to experiment and achieve

 ISBN: 978-1-61208-235-6

the best interaction scenario for the particular participants and venue. This is a sort of second order version of the Wizard of Oz technique (John F. Kelley, more directly applicably [16]). Ideally not only weights but the very degree of complexity and degree of interactivity should be adjustable parameters. It can be difficult to design perfectly and gauge the audience ahead of time. While the physical characteristics of a work are usually difficult to alter, with suitable algorithms and separation of data and control structures it becomes feasible to adjust the software response in real time. The degree of interactiveness can be a parameter in itself. There are the following possibilities

- 1st. order: the participants input influences the output
- 2nd. order: the participants input alters the installation itself (the machine tunes weighting factors, etc.; machine learning)
- the artists tune weighting factors
- the artists adjust major system parameters such as degree of complexity

III. Fuzzy sets, fuzzy logic, set operations

Say we want to quantify the enthusiasm of drummers. It is a matter of opinion whether 90 beats per minute is "fast" or "very fast" and it would be counter-intuitive to define 89 bpm as being vastly different from 90 bpm. What is needed here is a smooth transition, from what a human observer would call "fast" to what she would call "very fast". This can be described more accurately by fuzzy set theory.

Should we wish to determine whether the drumming at a given time or place is fast, we could define "fastness". We define a fuzzy set "fast" as a subset of the set of possible values for drumming speed as specified by $speed(x)$ for an object (a drummer) x by the following set membership function:

$$fast(x) = \begin{cases} 0 & speed(x) < 80 \\ \frac{speed(x)-80}{50} & 80 \leq speed(x) \leq 130 \\ 1 & speed(x) \geq 130 \end{cases}$$

In reality, membership functions are rarely this simple, and are not always based on a single dimension. It might for example be more intuitive and closer to the intention to define enthusiasm(x), based on both speed(x) and amplitude(x). We show a linear function for simplicity, a sigmoid function might be more suitable for many applications.

In general, a fuzzy set membership function $m_A : X \rightarrow [0, 1]$ (X being the universe of discourse) has the following properties:

$$m_A(x) \begin{cases} = 0 \text{ if } x \notin A \\ \in]0, 1[\text{ if } x \text{ is partially in } A \\ = 1 \text{ if } x \in A \end{cases}$$

where A is a fuzzy set (for example, elements that meet a particular criteria) and x is a variable $x \in X$ (for example media elements).

Generally, the intersection of fuzzy sets A and B (visualised in Figure 1) is defined by:

$$m_{A \cap B} = \min(m_A, m_B)$$

while the union is:

$$m_{A \cup B} = \max(m_A, m_B)$$

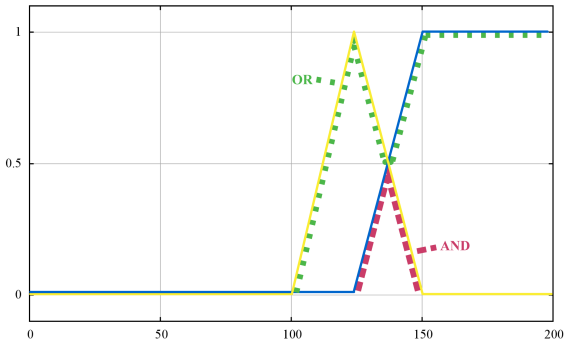

Fig. 1. Red dashed: about 125 AND high. Green dotted: About 125 OR high

The expression $m_A(x)$ can be considered as the truth value of the proposition "x is a member of A" and A can thus be considered as a logic predicate. There is isomorphy between fuzzy set theory and fuzzy logic and for given predicates A and B conjunction A∧B and disjunction A∨B can be defined correspondingly to intersection and union by:

$$m_{A \wedge B}(x) = \min\ (m_A(x), m_B(x))$$

$$m_{A \vee B}(x) = \max\ (m_A(x), m_B(x))$$

A. Set operations and aggregation

An installation may have inputs that are more or less important, that should be logically grouped, and there may be a limited number of choices (e.g., video clips) where we are interested in the best match given a number of different inputs, which may point in different directions. These requirements can be fulfilled with a layered, grouped approach with a combination of three types of aggregation. We are seeking means to combine inputs in flexible ways and are considering fuzzy set operations for this purpose.

A fuzzy set operation is an operation on fuzzy sets. Fuzzy set operations can be considered a generalization of crisp set operations, many generalizations being possible. The "standard" fuzzy set operations, intersection (conjunction) and union (disjunction), defined above immediately generalizes from 2 to n-argument operations such that for instance n cirteria can be combined by a conjunction:

$$m_{A_1 \vee \ldots \vee A_n}(x) = \max\ _{i=1,\ldots,n}(m_{A_i}(x))$$

These operations are encompassed by more general classes of operations. One important such class is the so called Ordered Weighted Aggregation (OWA). We define OWA below and introduce to generalizations taking importance weighting and hierarchical aggregation into account.

Ordered Weighted Averaging is a parameterisable class of mean type aggregation operators first introduced in [18]. The parameters are given as a set of so called order weights that

 ISBN: 978-1-61208-235-6

apply in the given order to the most, the second most, and so on, fulfilled criteria. An OWA operator is a mapping $F : R_n \rightarrow R$ that has a collection of order weights $W = [w_1, ..., w_n]$ in the range $[0, 1]$ such that:

$$F(a_1, ..., a_n) = \sum_{j=1}^{n} w_j b_j \quad \text{, where } b_j \text{ is the } j^{th} \text{ largest } a_i$$

with $\sum_{i=1}^{n} w_i = 1$ and where a_i is the degree to which the i'th criteria is fulfilled.

OWA enables us to balance artistic intent with media element (for example video clips) scarcity. While strong control of artistic intent might seem to indicate specific control of which criteria is most important is advantageous, in a real world situation with a limited number of media elements, it may mean the total fulfilledness of most criteria is low, because one was weighted as high importance and no media element was available that satisfied both the high importance parameter and the other parameters. With OWA we achieve both a guarantee for all parameters being included and additionally, as we know the sorted order of parameters, we can with weighting factors easily adjust whether highly fulfilled parameters are given most weight, less fulfilled parameters are given most weight, or all parameters are given equal weight.

OWA can be parameterised between $\wedge$ (pure conjunction) and $\vee$ (pure disjunction). The max, arithmetic average, median and min are members of this class. OWA has been widely used in computational intelligence because of the ability to model linguistic expressions.

While order weights relate to the best fulfilled order, *Importance weighting*, on the other hand, relates to specific criteria. Each input, e.g., from a sensor, is multiplied with a weighting factor, enabling us to define that, e.g., one input is twice as important as another input.

Hierarchical aggregation is a generalized aggregation introduced in its basic form in [19] that allows the combination of different types of aggregations based on the OWA operator and including importance weighting. Each node in the hierarchy can be attached individual parameters for order and importance weighting. The leaf nodes comprise a grouping of the input and the aggregation at each node delivers input to the parent node. Hierarchical aggregation thus allows us to group inputs, aggregate them in a group and send the result up in a hierarchy to a parent aggregation. This is especially useful when it makes sense to group inputs as there are classes of input that are fundamentally different in nature.

But, where does this leave the artist? There can be something very intuitive about an OWA rather than a logical expression, but how to allow an ordinary user to visualise the possibilities? Ideally we need a knob with complete user control unfettered by artist wishes at one end and complete artist control non-interactivity at the other end.

IV. Combining our aggregation operators to offer full parametrisation

We are aiming for fully parametric aggregation, adjustable by the artist by intuitive means. Especially, the order weights are difficult to set due to the requirement that they have to sum up to 1. However, we can define a simple function which provides all n order weights based on a single number between 0 and 1 as follows. Having defined such a function, we can introduce a slider in the interface for adjustment allowing an artist to decide on a scale how fulfilled the different criteria must be:

OR <—> AND

with the left extreme corresponding to "any requirements fulfilled" and the right to "all requirements fulfilled". This is an alternative to requiring traditional logical expressions.

Given a function such as:

$$Q(y) = y^{(\frac{1}{p}-1)} \quad , \quad p \in]0, 1]$$

we can introduce k weights from a single value for p:

$$w_i = Q(i) - Q(i-1) = \left(\frac{i}{k}\right)^{(\frac{1}{p}-1)} - \left(\frac{i-1}{k}\right)^{(\frac{1}{p}-1)}$$

with $i \in \{1, 2...k\}$ and $p \in]0, 1]$ This is described further in [17]. We want a function, that given one number, gives us n weights out, and they must sum to 1. This method of parametrising a function, calculating all n weights at once using only one input is only useful with OWA. The method is to take the difference between function values, given x values of a multiple of $1/n$ where n is the number of parameters and hence the number of weights required. The function must be monotonically increasing in the interval [0,1] and the sum of the weights must be 1. Any function that fulfils these requirements could potentially be used.

$$w_1 = Q\left(\frac{1}{n}\right) - Q\left(\frac{0}{n}\right) \; ... \; w_n = Q\left(\frac{n}{n}\right) - Q\left(\frac{n-1}{n}\right)$$

In the concrete case illustrated in Figure 2, for $n = 4$ and $p = 0.3$, the weights are

$$w_1 = 0.039 \quad w_2 = 0.159 \quad w_3 = 0.313 \quad w_4 = 0.489$$

A. Importance weighted aggregation

Importance weighted aggregation involves giving each parameter a weight, often but not necessarily in the range [0,1]. In an interface, this can be achieved by having a control and display of the importance weight for each parameter - for example sliders or knobs. If all the weights are the same (e.g., 1), the importance weighting is neutral and in effect turned off.

 ISBN: 978-1-61208-235-6

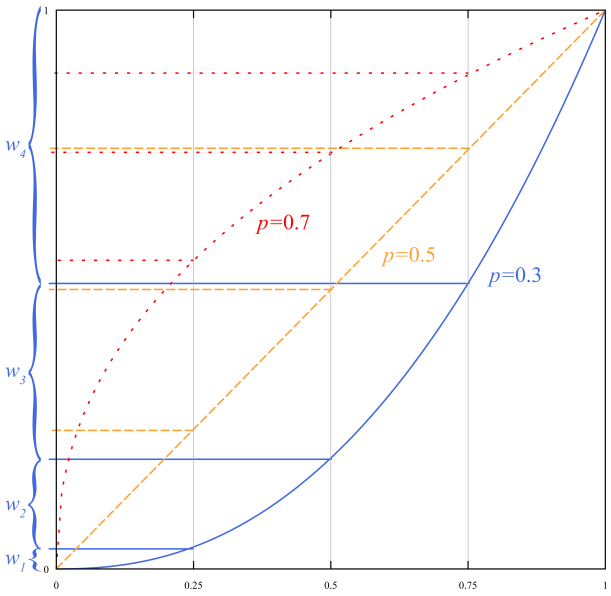

Fig. 2. Finding n OWA weights which sum to 1, for $n = 4$.

B. Hierarchical

Hierarchical aggregation is a layered, grouped aggregation. In our design it is used as a concrete way of combining grouping, OWA and importance weighted aggregation. A hierarchical aggregation is suitable for grouping. If there are very different parameters (inputs), e.g., sound level and geographical position, a hierarchical aggregation will give a natural grouping. See Figure for an example.

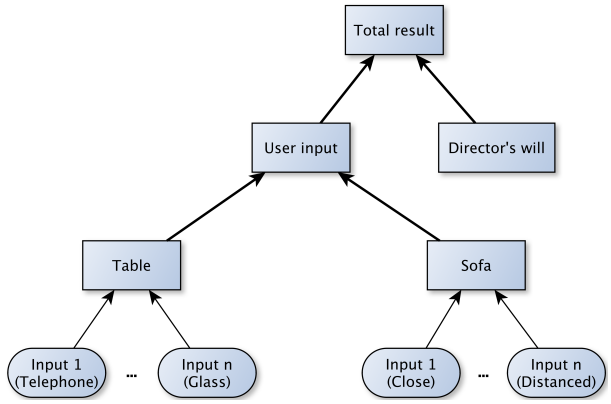

Fig. 3. An example hierarchigal aggregation. Rectangles are aggregations, ovals are inputs.

Here, each rectangle is an aggregation, in our generalised system each is first an importance weighted, then a parametrisable OWA.

C. Combining

It is quite possible to combine one or many different forms of aggregation, either simultaneously or in nodes of a hierarchy. It is not necessary to provide different aggregation objects as the adjustable parameters provide the possibility to "turn off" any unwanted feature, allowing us to keep it simple, using only one type of aggregation object.

An importance weighted ordered weighted aggregation could look like this: in are inputs, imp are importance weights, then sorting by value, $weight$ are the weighting factors we obtained above. OWA is a method that enables using all inputs instead of just one. This is done by adding all the inputs, but as we want the result of the aggregation to be in $[0, 1]$, we first regulate the inputs by multiplying them by weights. Which input is regulated by which weight is decided by the value of the input, so we sort the inputs before multiplying by the weights. For example, given 4 inputs a_1, a_2, a_3, a_4, we sort them by size. The sorted input we call b_1, b_2, b_3, b_4, and we then multiply b_1 by $weight_1$ so: $(b_1 \times weight_1), (b_2 \times weight_2), (b_3 \times weight_3), (b_4 \times weight_4)$

In this design, the above combined OWA and importance weighted aggregation is used as each node of a user-designable hierarchy, providing a ordered weighted importance weighted hierarchical aggregation. This gives the full freedom to weight parameters (e.g., inputs) by importance, to linearly choose from a range from AND to OR for the OWA aggregation, and to group the parameters in any number of groups and steps. The final output is from the topmost aggregation, the node with no parent.

This fully parameterised design allows the artist to play the system at the meta level (i.e., not altering the inputs, which come from the participants, but more subtly adjusting, in real time, how the inputs are processed. This feature can be augmented by implementing standards based real time adjustment of sliders, etc., enabling use of a physical control surface, e.g., a MIDI (Musical Instrument Digital Interface, a de facto industry standard for sound and interactive installation control) controller.

D. Machine learning

While the initial goal of this work is to enable the artist/director to easily adjust and experiment with the aggregations of a fuzzy logic based artificial intelligence, the logical next step is to implement adaptive fuzzy logic, machine learning. Our design allows any number of aggregations, sensors or outputs can be routed to any number of inputs, and weights are all in the range [0,1] — all that is missing is to allow an output to control a weight.

While actual implementation is for further work, an overview of required aspects includes a paradigm for combining linguistic and numerical information; choice of learning using, e.g., backpropagation, feedback loops, orthogonal least squares or neighbourhood clustering; and possible inclusion of artificial neural networks in a hybrid neurofuzzy system. Adaptive fuzzy filtering will probably be advantageous given the dynamic and nonlinear nature of many of our systems. All these aspects are active research topics. Adaptive fuzzy logic is often used for systems where it is advantageous to learn a system's characteristics and avoid hysteresis, e.g., maximum power point trackers for solar panel arrays - supplanting, e.g.,

 ISBN: 978-1-61208-235-6

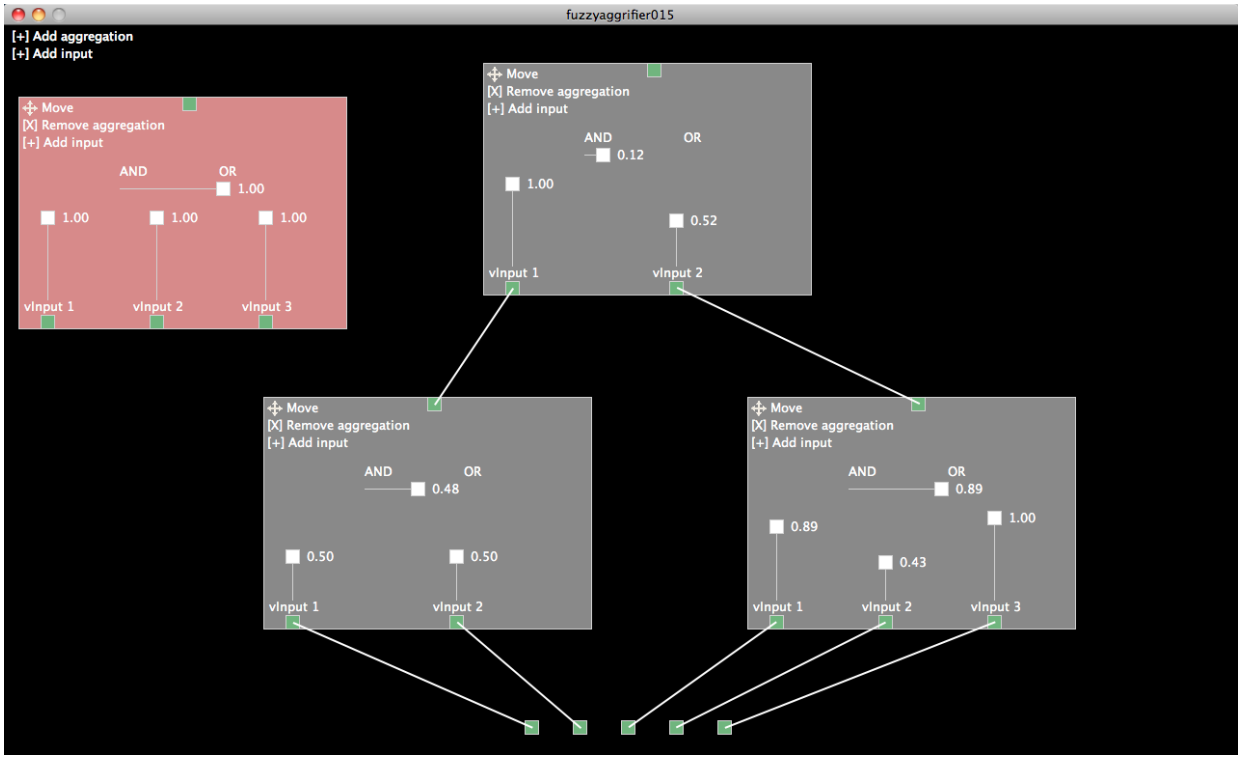

Fig. 4. Experimental interface design iteration 0.8.

PID (proportional-integral-derivative) controllers in diverse applications [12].

V. Solution: What we want for artists - Interface design rationale and experiments

The requirements for the system are that it:

- Must process the data in real time
- Make it easy to enter data for different projects - separation of data, meta data and control logic is required.
- Handle complexity so great the output is not recognisably deterministic.
- Be second order adjustable in real time (i.e., allow weightings etc. to be adjusted)
- Preferably have easy enough controls (e.g., a hardware control surface) that it invites "playing" and "tweaking" once configured.
- Preferably be understandable by a motivated artist
- Preferably be capable of adjusting itself if so wished

The objective is to design an interface for artists which makes complex fuzzy operations reasonably intuitive. The interface should enable a non computer scientist to design a fuzzy AI system, and tune it in real time. What might seem a daunting task is made easier by the fact that understandable metaphors are available for most steps of the process. Grouping can be symbolised with boxes, hierarchy with lines between these boxes, importance with sliders like an audio mixer or knobs like volume. While a complete system including databases, all kinds of input and output options and machine learning is a major undertaking, the system initial design and programming is progressing well and the interface design has already been through several iterations.

The interface is designed thus: at the bottom are the inputs, any number can be instantiated by pressing "[+] Add input". At the top of each aggregation is an output; if there is no parent this is considered the final output. Any number of boxes=groups=aggregation operations can be instantiated by pressing the "[+] Add aggregation" button. Each box is an object which contains an arbitrary number of inputs, including a slider to define an importance weight for each input. An input can be added by pressing the "[+] Add input" button. It contains methods for an OWA aggregation, a slider for variably parametrically adjusting this aggregation from AND to OR, and one output.

The box is by its nature a grouping. It is often opportune to be able to group inputs. For example, inputs which are in their nature fundamentally different, e.g., microphone volume and geographical position. Lines can be drawn between objects by clicking on a symbolic output jack and then on a symbolic input jack. A sensor can be routed to any number of inputs and the output of an aggregation can be routed to any number of inputs.

Each node in the hierarchy has its own individually controllable OWA and importance weights, just as its place in the hierarchy and what inputs it receives is user controllable. It is possible to mix - a node can perfectly well take input both from sensors and from an aggregation lower in the hierarchy.

Adjusting a slider from AND to OR may be intuitive for some artists and not for others – an investigation of this

 ISBN: 978-1-61208-235-6

is for future work – but it is certainly more user friendly than requiring a non technical user to *a priori* choose from a predefined selection of aggregations, e.g., MAX; MIN; AND; OR; AVG; MEDIAN. Tuning an importance weight is easily understandable, it corresponds to turning the "value" or "volume" of a sensor or input up and down. A hierarchical aggregation is a natural way to support grouped input, and a visual hierarchy is a natural way to visualise the hierarchy and grouping.

A. Complete solution

The above implemented interface is useful for experimenting with adjusting aggregation; in practical applications, a few more object types are required. Time is a factor, this can be dealt with in three ways: the system can be input driven, request driven or clock/bang driven. Input driven means that when any data arrives, the whole calculation is performed. Request driven would be that another part of the system requests an answer. This could be suitable for, e.g., a video installation, where the system is aware that the present video is nearing its end, and could ask for what video clip to show next. Clock/bang driven would mean that there is a source that propagates a command "do it now" to all relevant objects "Bang" is a term (used in PureData and MaxMSP) for an irregularly timed pulse, e.g., sent only on certain conditions.

The above still leaves us with the aggregations being the result of a snapshot, a moment in time. It would be advantageous to be able to aggregate over time, just as it would be advantageous to be able to let earlier results affect later aggregations - adding feedback and state to the system. Both goals can be accomplished with a multifunctional FIFO (first in first out) object. The FIFO object is a bucket chain, each bucket storing a result. The user can determine the number of buckets and therein the maximum delay. There is an input at the beginning of the FIFO, an output at the end, which provides the input, in order, but delayed by n cycles. Additionally, there is an output from each stage or bucket. It is optional to use these outputs, they are there to enable the aggregation of inputs over time. By making a ten stage FIFO and connecting the ten stage outputs to a ten input OWA aggregation, it is easy to aggregate the last ten sensor inputs over time. Using the FIFO, some "sensors" can be, e.g., metadata about the current media element and some can be state.

VI. Conclusion and future work

We have demonstrated a design for a user friendly interface enabling non technical users to define and adjust in realtime a set of completely parameterizable complex aggregations. We believe an intuitive interface for a hierarchical, importance weighted, ordered weighted aggregation is an original idea. While simple user friendly interfaces offering a flexible alternative to logical expressions (e.g., AND, OR) consisting of one slider in conjunction with a search field [17] have been demonstrated, enabling a user to easily design a complex aggregation and adjust it in real time is to our knowledge new. We have sketched an experimental prototype of the generalized fuzzy logic interface; field evaluation remains for future work.

References

[1] T. Andreasen, H. Christiansen, J. P. Gallagher, C. Jacquemin, N. Møbius, N. Padfield, S. Siggaard, D. L. Strand, and P. R. Sørensen *Havhingstens Tur til Irland: en interaktiv oplevelsesplatform.* CBIT, Roskilde University, 2011.

[2] T. Andreasen, J. P. Gallagher, N. Møbius, and N. Padfield. *The Experience Cylinder, an immersive interactive platform: The Sea Stallion's voyage: a case study*. In: R. Emonet, A. M. Florea, (ed). AMBIENT 2011, The First International Conference on Ambient Computing, Applications, Services and Technologies. ThinkMind, 2011 pp. 25-31.

[3] J. Montemayor, A. Druin, A. Farber, S. Simms, W. Churaman, and A. D'Amour. *Physical programming: designing tools for children to create physical interactive environments.* CHI. 2002.

[4] C. S. Pinhanez, J. W. Davis, S. S. Intille, M. P. Johnson, A. D. Wilson, A. F. Bobick, and B. Blumberg *Physically interactive story environments.* IBM Systems Journal 39, no. 3 & 4, 2000.

[5] M. Slater and S. Wilbur. *A framework for immersive virtual environments (FIVE): Speculations on the role of presence in virtual environments.* Presence 6, no. 6, 1997 pp. 603-616.

[6] J. M. Sales Dias *Natural and tangible human-computer interfaces for augmented environments.* Proceedings of the 26th annual ACM international conference on Design of communication, SIGDOC '08. ACM, 2008, pp. 181-182.

[7] M. W. Krueger *Environmental technology: Making the real world virtual.* Comm. ACM, 1993, pp. 36-37.

[8] M. Danks, M. Goodchild, K. Rodriguez-Echavarria, D. B. Arnold, and R. Griffiths. *Interactive storytelling and gaming environments for museums: The interactive storytelling exhibition project.* Technologies for E-Learning and Digital Entertainment, Second International Conference, Edutainment 2007. Lecture Notes in Computer Science, 2007, pp. 104-115.

[9] D. Grigorovici *Persuasive effects of presence in immersive virtual environments.* Being There: Concepts, effects and measurement of user presence in synthetic environments. Ios Press, 2003.

[10] B. Goertzel and C. Pennachin, eds. *Artificial General Intelligence*, Springer, 2006.

[11] A. Darwiche, *Modeling and Reasoning with Bayesian Networks* Leiden : Cambridge Univ. Press, 2009.

[12] D. Misir, H. A. Malki, and G. Chen, *Design and analysis of a fuzzy proportional-integral-derivative controller*, Fuzzy Sets and Systems. Volume 79, Issue 3, 1996, pp. 297 - 314.

[13] B. Myers, S. E. Hudson, and R. Pausch, *Past, Present, and Future of User Interface Software Tools*, Carnegie Mellon University, ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1 Volume 7 Issue 1, March 2000. Open access: http://dl.acm.org/citation.cfm?id=344959 [retrieved: 7, 2012]

[14] G. J. Klir and B. Yuan, *Fuzzy sets and fuzzy logic: theory and applications*. Upper Saddle River, NJ: Prentice Hall 1995.

[15] L. A. Zadeh, *Fuzzy Sets*, Information and control (8), pp. 338-353 http://www-bisc.cs.berkeley.edu/Zadeh-1965.pdf [retrieved: 7, 2012]

[16] B. Buxton, *Sketching User Experiences: Getting the Design Right and the Right Design*, Interactive Technologies, Morgan Kaufmann 2007.

[17] T. Andreasen, *Query Aggregation Reflecting Domain-knowledge*, Intelligent systems for information processing - from representation to applications, Bernadette Bouchon-Meunier, Laurend Foulloy, Ronald R. Yager (eds.) Elsevier 2003, pp. 107-116.

[18] R. R. Yager, *On order weighted averaging aggregation operators in multicriteria decision making*, IEEE Transactions on Systems, Man and Cybernetics, 18(1), 1988, pp. 183-190.

[19] R. R. Yager, *A hierarchical document retrieval language* Information Retrieval 3, 2000, pp. 357-377.

 ISBN: 978-1-61208-235-6