On-line Development of Man-Machine Relationships: Motivation-driven Musical Interaction

Peter Beyls
University of Plymouth, UK
peter.beyls@plymouth.ac.uk

Abstract

This paper documents an interactive musical generative system built from a minimal explicit design specification. Given the fact that conventional, responsive rule-based systems cannot cope with large swings in context, a method of on-line development is suggested. The machine develops a dynamic motivation as to whether integrate with a musical context suggested by a human interactor, or in contrast, express a native musical character. Motivations are configured as networks of relationships that provide continuous interpretation of changes in human behaviour. A combination of a simple form of reinforcement learning and genetic evolution continuously optimise motivations in order to accommodate input from an unpredictable human performer. Experimental evidence shows that man and machine may indeed develop common objectives such as mutual agreement during interaction.

1. Introduction

Generative works of art are often thought of as conceptual machines that -- once they are activated -- realise themselves. Given enough critical mass, such works correspond to micro-universes obeying some invented physics of arbitrary complexity. Conventional rule-based systems have been used successfully to implement style specific programs; they rely on the complexity entailed from the combinatorial explosion of the implied rule-base (McCorduck 1991, Cope 2005). In contrast, much artificial life oriented work follows the premise that both interesting morphology and behaviour may follows from the local interaction of simple rules (Sims 1994). Both approaches prove to be effective instruments for managing visual complexity. However, they are both characterised by a *one-way specification* of expertise; the artist implements rules while observing their implicit results. Most significant, the artist thus specifies further rules from the observation of the current behaviour of the program; therefore programming generative systems is a form of artistic introspection. In reality, one-way specification takes place in a creative procedure of circular thinking. The main point here is that the art production system (whether visual, musical or hybrid) is thought of a as a closed container -- complexity of form and behaviour is conditioned by some local rule-base in isolation -- the system is not grounded into the real world and, in this respect, it is not interactive.

In contrast, this paper addresses the issue of open systems, systems that feature a physical connection with the external universe in which we live. Such systems offer

internal generative potential while remaining open to influence from outside. In addition, open systems may develop *autonomous* behaviour rather than reflect the automatic behaviour from the activation of a rule-base in isolation. Autonomous systems are intrinsically *interactive*; they develop their own rules as a side effect of interaction itself. The resulting spatiotemporal patterns observed in biological workspaces speak to the imagination (Bentley 2001), therefore we attempt to identify the essential features of living systems as they may guide us to the synthesis of artificial ones.
In short, a living system contains some form of self-representation stretched out in its DNA, a living system exchanges information and energy with a dynamic environment, they rely on the integrity and relationships between their constituent components, they evolve over many generations while learning in-between points of evolutionary breeding. Finally, living systems develop motivations on-line, what to do next does not follow programmed instructions but emerges spontaneously from a set of competing drives.

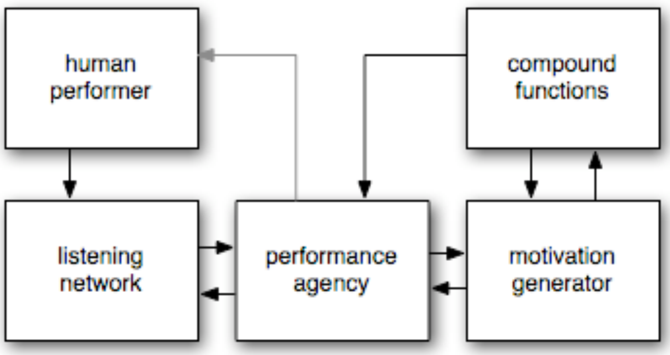

Figure 1: Global systems outline comprising functionality for listening, performance and on-line synthesis of machine motivations

We suggest these features to serve as first principles towards the development of an interactive musical entity, a system that will express its own native character while remaining open to pressure from a single human improviser. Behaviour in such a system follows from the competition between two conflicting forces: either *expression* (create output irrespective of current context) or *integration* (create output that is complementary to the prevailing context and keen to contribute to its further existence). We developed a real-time architecture (see figure 1) to support rewarding man-machine interaction, it consists of three main networks; (1) an evolved sensor-activator network for the purpose of machine listening (Beyls 2005), a distributed player agency equipped with evolved musical processing functions (Beyls 2007) and finally, networks serving the representation and management of internal machine motivations. The latter is the core subject of the present paper.

We take inspiration from the biologically rooted behaviourist theory of motivation (Mook 1987) as it avoids all explanation of actions in terms of internal events such as desires and emotions. In contrast, behaviourist thinking explains complex behaviour in terms of external impact from the environment. One may consider the theory of *autopoiesis* (Maturana and Varela 1992) as a generalisation of this idea; the theory of structural coupling suggested here explains interaction not so much in terms of the complexity or content of a signal but in terms of the kind of structural changes it causes in the receiver. In this light, the model presented here is based on the articulation of activation and inhibition forces in a networked architecture; perpetual renewal of its structure is achievable while, however, its structural integrity (ontogenesis) is guaranteed.

2. Rationale of motivation-based interaction

Provided that our system aspires autonomous behaviour, we cannot accept its identity to be designed exclusively by external, human designed specifications. We hope for a musical personality that maximises diversity, a system spawning many musical trails of great variety. System behaviour should be totally unpredictable while still displaying a coherent personality. Articulate musical patterns should develop from initial randomness. In other words, interaction is seen as navigation in a vastly multidimensional space featuring patterns of great diversity; from relative periodicity to unrestricted chaos. Such dynamic patterns reflect the variable degrees of man-machine understanding within the process of interaction.
A dynamic mechanism is needed that can make up criteria to interpret external agitation in terms of positive (agreement) or negative (conflict) impact. It must be robust and create an opinion by itself according to demands generated by its own internal dynamics. The *drive object* aims to provide such a structure.

A drive is a computational object that that specifies a simple psychological orientation. It can be considered an abstract suggestive *machine speculation*. It has two options: either integration or expression. *Integration* means that the machine aims to produce music that assimilates well with the last sequence played by the human player. In contrast, *expression* implies that the drive prefers to move away from the musical style suggested by the human improviser. The options are not mutually exclusive, they are viewed as two competing alternatives represented by two fluctuating quantities on a scale of 0 to 100.

The drive object implements a first principle: the appreciation and accommodation of change. Activation of the system happens in terms of changes, that is, *signed intervals* reflecting changes of features of given melodic material[1]. A first order quantity in our system is the current melodic distance between the last sequences produced by man and machine. We simply track if musical similarity (i.e. the inverse of musical distance) between the output of man and machine actually increases, decreases or remains the same over time.

At the very moment the machine just finished playing (detected by an adaptive segmentation algorithm) its most recent response, the effect on the situation can be computed. If the new distance is higher than the previous distance, we interpret this action as a wish to increase musical contrast between man and machine. If the new distance is lower, we know that both parties are musically getting closer together. Intuitively, we may track consecutive differences (delta-similarities) in time. If many

[1] In practice, a feature vector is computed documenting changes of a large number of higher level features in a four dimensional representation of a string of musical events (pitch, loudness, duration and entry-delay), these include, for instance, changes in diversity, regularity, tempo, harmonic tension and entropy.

such consecutive delta-similarities have the same sign, we infer that man and machine are either engaged in an escalating process of incremental contrast (negative sign) or apparent mutual understanding (positive sign).

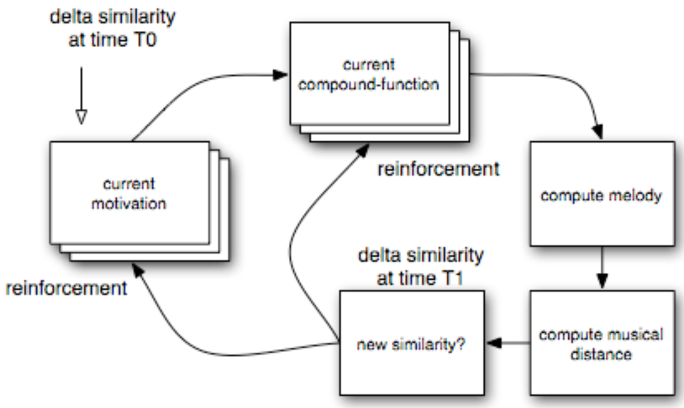

Figure 2: Circular model featuring reinforcement of motivation levels driven by changes in the environment.

Now, the idea is to learn which behavioural motivation (integration or expression) should dominate given a specific perception of behavioural changes in the human performer. From this knowledge, an appropriate musical processing function can be selected to fulfil that specific orientation.

The notion of a *relationship* was developed to specify a qualitative link between observed external changes and internal quantities representing the strength of an internal motivation. Internal motivations and external pressures are thus operationally connected as a complex dynamical system.

3. Relationships

The idea of a *relationship* is inspired on the two-axis theory of personality developed in (Eysenck 1973) and the relationships inside model ecosystems described in (Steels 1995). Eysenck's model suggests a four-quadrant system with the horizontal axis denoting a degree of stability (stable to unstable) and the vertical axis denoting introverted vs. extraverted behaviour. The way the human performer behaves is imagined as being reflected in the two-axis model. Behavioural changes are suggested by specific trajectories in two-dimensional space. Steels' model involves the acquisition of couplings between processes in the environment and internal processes. It studies couplings in order to evolve favourable relationships between a mobile robot, its resources and an unpredictable environment. Four complementary types of couplings are suggested which we view as functionally equivalent to the four quadrants described by Eysenck.

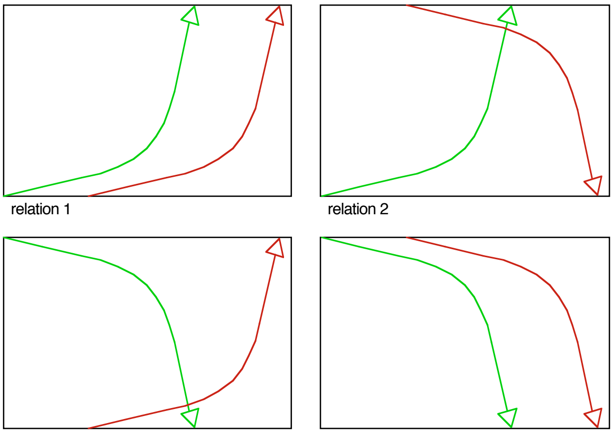

Figure 3: four types of basic relations between changes in an input quantity (green arrows) and the effect in an output quantity (red arrows).

A basic set of four relations exist, we consider them in their most basic form, as linking two quantities by way of a multiplication factor *ƒ*. A more qualitative interpretation follows in the next paragraph. For now we consider the 4 different couplings between changes in a source quantity ΔQ-source(t) at time t, and the value of a destination quantity Q-dest(i+1) at time (t+1).

The 4 relations are defined as follows:

```
If ΔQ-source(t) > 0 then Q-dest(t+1) = Q-dest(t) + ΔQ-source(t)*ƒ1
If ΔQ-source(t) > 0 then Q-dest(t+1) = Q-dest(t) - ΔQ-source(t)*ƒ2
If ΔQ-source(t) < 0 then Q-dest(t+1) = Q-dest(t) + ABS(ΔQ-source(t))*ƒ3
If ΔQ-source(t) < 0 then Q-dest(t+1) = Q-dest(t) – ABS(ΔQ-source(t))*ƒ4
```

The four types of relations are visualised in figure 3. Note that every relation { r1 … r4 } features a private multiplicative weighting factor { *ƒ*1 … *ƒ*4 }. Relation type 1 implies that a positive change in a source quantity will introduce a positive change in a destination quantity, the amount of change being proportional to the change at the source modulated by the private weighing factor of relation 1. In relation type 2, positive input changes produce negative output changes: output is inverse proportional to input. Relation type 3 connects negative input changes to positive output changes. Finally, relation type 4 implies that negative input changes produce negative output changes, again scaled by the weighting factor of the given relation. It was decided to keep the weighting factors local to every relation, rather than have individual weights in every relationship in order to limit the state space and create a better chance to monitor the impact of the individual relations.

4. Implementation of motivations

A motivation is implemented as a drive object that is sensitive to three kinds of changes: (1) the first derivates of the similarity between the most recent melody produced by man and machine and (2) the first derivates of the quality and (3) quantity of the contents of the most recent man produced melody. All input changes are computed and normalized in a range -100 to +100.

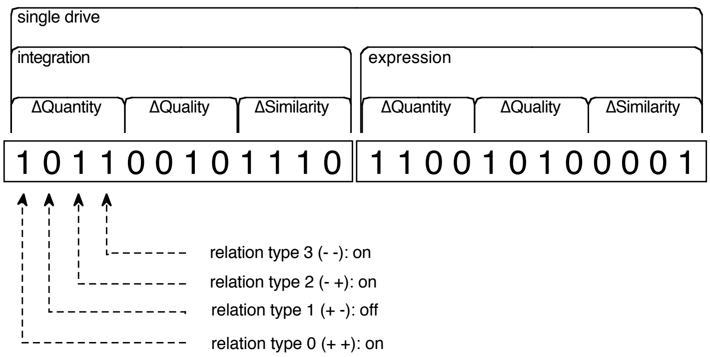

Figure 4: Topology of a single drive object showing three groups of four bits per motivation

Relationships are specified as two 12-bit vectors. Since there are three types of input sensors each feeding four types of relationships, we must accommodate 12 potential effects of external change – 3 blocks of 4 bits. For instance, bits 0~3 account for delta-similarities, bits 4~7 account for delta-quality and bits 8~11 account for delta-quantity. We must include both primitive motivations: integration and expression resulting in a total of 24 bits. If a bit equals 1, it means that its relationship is active, if the bit equals zero, its relationship is not accounted for. Note that many relationships can be active (bit on) in a single block. The output will reflect the contributions of all active relationships. When simultaneous relationships contribute opposite pressure, they may partially neutralise their mutual effect. This phenomenon contributes to non-linearity in the network.

Intuitively, we understand that the density of 'on' bits in the vector will condition the global responsiveness of the drive. Too many 'on' bits may potentially produce over-stimulation leading to erratic output. In contrast, too few 'on' bits lead to under-stimulation, in this case, significant changes in input may get lost. We turn to a learning algorithm that learns to create appropriate couplings between input changes and internal motivations.

```
drive-ID            : 0
nr-runs             : 7
current-orientation : EXPRESSION
relationships Exp   : 1  0  1  0  1  1  0  1  0  0  1  0
relationships Int   : 1  0  1  1  1  0  0  0  1  0  0  0
expression-level    : 80.86634
integration-level   : 31.14185
efficiency-value    :  2.24824
understanding-level : 45.21349
```

Figure 5: Prototype snapshot, list of principal instance-variable inside a drive.

The behavioural motivation of a drive – its current orientation -- depends on the strength of the two competing levels (0~100) for integration and expression. We expect a minimum contrast between both values; we introduce a threshold of 10%. If the difference between the levels for integration and expression is higher than 10%, then the higher value decides on the orientation else the current orientation remains ambiguous.

Notice the current orientation in figure 4 is *expression*. This has a double impact on further computations. First, the expression-vector becomes the source of temporary relationships and second, the output value affected by these relationships is the expression-level. The purpose of the *understanding-level* instance variable is addressed in section 5.2.

Now, as an example, consider the first block of 4 bits of the Integration relationships: (1 0 1 1). Since the first bit is 'on', relationship type 1 (+ +) takes effect. Thus when the input level increases the output level follows. Relationships type 2 (+ -) is not considered since the second bit equals zero. The third bit is 'on' meaning that the contribution of a relationship type 3 (- +) is added to the previous. In other words, when delta-similarity is either positive or negative, the output level will increase. In addition, the relationship type 4 (- -) says that if input level decreases the output level will follow in the same direction.

5. Learning in the drive object

5.1 Learning to be efficient

It is important to know how *efficient* a given drive actually is. When the external changes are processed by the relationships, they receive a qualitative interpretation because of non-linear couplings take place between the dynamics of external higher level quantities (similarity, quality and quantity) and competing internal behavioural motivations (integration and expression). Given the current orientation, we analyse if man and machine are coming together or drifting apart – according to their melodic similarity. For example, in pseudo-code;

```
if ( current-orientation == integration )
        if (currDist – prevDist) > 0 )
        then ( efficiency = (efficiency * inhibition-weight))
        else
        if (currDist – prevDist) < 0 )
        then ( efficiency = (min 100 (max 1 (efficiency * activation-weight))))

1.11 < activation-weight < 1.50
0.50 < inhibition-weight < 0.99
```

The learning method suggested here is similar to reinforcement leaning (RL) (Sutton and Barto 1998). RL is a form of unsupervised learning; the learning agent receives feedback about how appropriate its actions are in order to achieve a given goal. However, the goal itself is not communicated, the agent aims to approach optima essentially by trial-and-error and learns from positive (rewards) or negative (punishment) feedback – corresponding to the fluctuations in drive efficiency according conditional scaling by activation and inhibition weights. As the efficiency of a given drive increases, so will its probability to be selected in succeeding trials. Eventually, the drives-pool will progress towards values that will maximise reinforcement. RL is typically applied in real-world problems characterised by a huge state space. All of this seems to fit the essence of interactive composing: man and machine must learn to behave successfully without any a priori information about their mutual personalities.

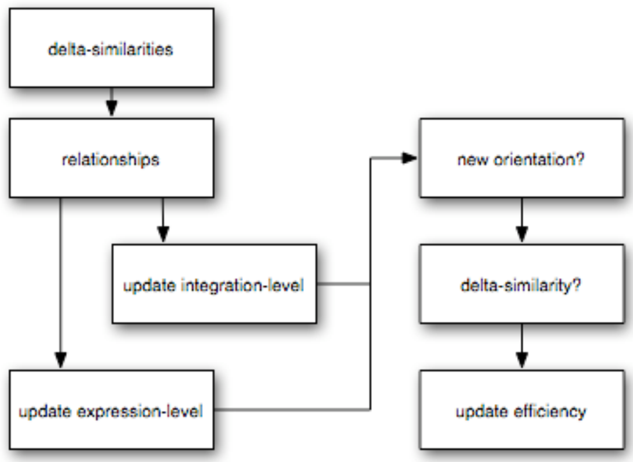

Figure 6: Simplified representation of two-stage learning algorithm: evaluation of the drive's relationships and consequently adjusting the drive's efficiency level.

A two-stage learning algorithm is depicted in figure 6. Stage one updates the levels of integration and expression from the evaluation of the current relationships. Stage two updates the efficiency according to the current orientation. Let us tackle stage one in detail. For every delta value, the respective slot of the relationships-vector is evaluated.

Stage one encloses two nested loops; the arguments are the current gradients (delta-values) in man-machine melodic similarity and the changes in quality and quantity of the contents of working memory. All 24 bits of the relationships-vector are addressed (figure 4). The levels are scaled up (activation) or down (inhibition) according to the type of relationship and the delta value. In the end, the integration and expression levels will reflect the accumulated impact of the combination of vector on-bits and the sign of the respective delta-values.

Stage two evaluates the resulting (potentially changed) drive-orientation, decided on by taking the highest value of the two competing levels as the winning current orientation. Take note that we exploit only the delta-similarity at stage two. This delta value and its sign provide information as to whether the current relationships were helpful to steer the drive towards the optimal orientation. The intended optimal orientation (integration or expression) is the one that is consistent with the last change in man-machine similarity. For example, when the man-machine melodic distance decreases and the orientation is integration, we conclude that the drive is indeed resourceful towards the fulfilment of this drive's orientation – therefore, its efficiency-level is scaled up using the activation-factor. In similar vein, in case the melodic distance increases and the orientation is expression, the efficiency level is also scaled up. Efficiency-level is inhibited when the changes in distance are in conflict with the orientation i.e. either a combination of integration and increasing distance or expression with decreasing distance.

Any given orientation is considered a machine *suggestion* to temporarily approach musical interaction from a given perspective i.e. either a wish for man and machine drifting apart (expression) or narrowing the man-machine melodic distance (integration). The rationale is that a suggestion is first generated at random and subsequently adjusted according to the evaluation of the data gathered during actual interaction – the argument for having a learning component in the first place.

Finally, the obvious question arises of how to compute the similarity of two melodies (the last segment input by the human performer and the current machine response), possibly of unequal length. Different methods were implemented and evaluated, including; the use of 4 transition matrixes tracking the quantised values of consecutive intervals in a sequence of MIDI events in the dimensions of pitch, velocity, duration and inter-onset-time – similarity is viewed as being proportional to the degree of overlap between the respective matrixes. The current implementation calculates similarity indirectly; by comparing the feature-vectors[2] of two given melodies. Further discussion is, however, beyond the scope of the present paper.

5.2 Learning to optimise man-machine agreement

A second higher-level form of learning is introduced. Remember, the implicit goal of our system is to maximise man-machine agreement i.e. having man and machine demonstrating the same *global* orientation. To this purpose, one additional quantity is introduced, the *understanding-level*, acting as an instance variable in the drive object (figure 5). This variable is updated proportional to the degree of conflict or agreement between the current global orientations of man and machine.

The listening module continuously adjusts two quantities: *integration-pressure* and *expression-pressure*, scalars between 0~100. The update is proportional to the current delta-similarity – the amount and direction of change in similarity between the most recent and the previous musical sequence played by the human performer in relation to a given machine output. For instance, when the musical distance decreases (similarity increases), we infer that the human performer wishes to integrate, so *integration-pressure* is scaled up and *expression-pressure* is slightly scaled down. The activation factor is proportional to the absolute value of the similarity interval, formally:

```
activation-weight = 1 + abs(delta-similarity) / 5
inhibition-weight = 0.98

if ( delta-similarity > 0 )
        integration-pressure =
                (min 100, integration-pressure * activation-weight)
        expression-pressure =
                expression-pressure * inhibition-weight

        else
        if ( delta-similarity < 0 )
                expression-pressure =
                        (min 100, expression-pressure * activation-weight)
                integration-pressure =
                        integration-pressure * inhibition-weight
        then
then
```

[2] We analyse the respective lists for global direction (incremental, decremental or stationary), angularity (smooth or angular), regularity and diversity (low or high), and the relationship between the first and last value (interval is positive, negative or zero). This yields two 48 element binary feature-vectors; melodic similarly is considered proportional to the amount of coinciding values in both vectors.

The level of *human-global-orientation* is obtained as follows:

```
sum = integration-level + expression-level

if (integration-level > expression-level)
        orientation = ( integration-level / sum ) * 100
        else
        orientation = ( -1 * (expression-level / sum )) * 100
then
```

The resulting global orientation yields a signed value between -100 and +100, negative values denoting expression, positive integration.

The complementary level of *machine-global-orientation* is computed using a similar algorithm; the contrast between levels of integration and expression in the current drive returns a likewise signed numeric result between -100 and +100.

It is now straightforward to compare the formatted orientations of man and machine in order to infer an estimate of *common-understanding* i.e. the nature and strength of mutual orientation between both. In pseudo code:

```
if (machine-global-orientation.signum == human-global-orientation.signum
        common-understanding = Agreement
        common-understanding-level =
                (human-global-orientation.abs + machine-global-orientation.abs) / 2
        else
        common-understanding = Conflict
        common-understanding-level =
                (human-global-orientation.abs - machine-global-orientation.abs) / 2
```

The procedure above returns a signed value reflecting the strength and type of man-machine orientation, *agreement* occurs when both pursue the same orientation i.e. either integration or expression. The interaction is characterized as in *conflict* when both interactors produce levels of divergent sign. Then, the understanding-level of the current drive is further adjusted as follows:

```
if common-understanding-level > 0
        ;; activation
        factor = remap(common-understanding-level 0  100 1.0 3.0 )
        else
        ;; inhibition
        factor = remap(common-understanding-level 0 -100 1.0 0.3 )

understanding-level = min( 100, understanding-level * factor )
```

The reinforcement factor is proportional to the common-understanding-level and its sign; levels of 0 ~ 100 and -100 ~ 0 are remapped to respectively 1.0 ~ 3.0 and 0.3 ~ 1.0). The understanding-level is a second indication of how appropriate a given drive performs given a specific interaction context – it may thus guide the selection of specific drives in forthcoming interactions.

Finally, we might wish to get an impression of the *global* behaviour of the complex dynamical system comprising a single man and a single machine; the dynamics of the interaction as articulated by the fluctuating orientations of both parties. The *system-global-orientation* (-100 to 100) is computed as:

```
system-global-orientation = (human-global-orientation + drive-global-orientation)/2
```

This average continuously documents global systems behaviour at the highest level of abstraction; in essence, it provides an indication of the social processes emerging from the interaction process itself.

6. Managing simultaneous, competing motivations

A critical mass is needed *and* a procedure to maximise diversity and guarantee the potential development of many different types of interactions. We turn to a genetic algorithm (Goldberg 1989) to breed fresh populations of drives by considering the fittest drives (the most efficient drives) as parents to breed the next set of offsprings (see section 7). However, let us first examine how acquired competence may actually be put to good use within the process of interaction.

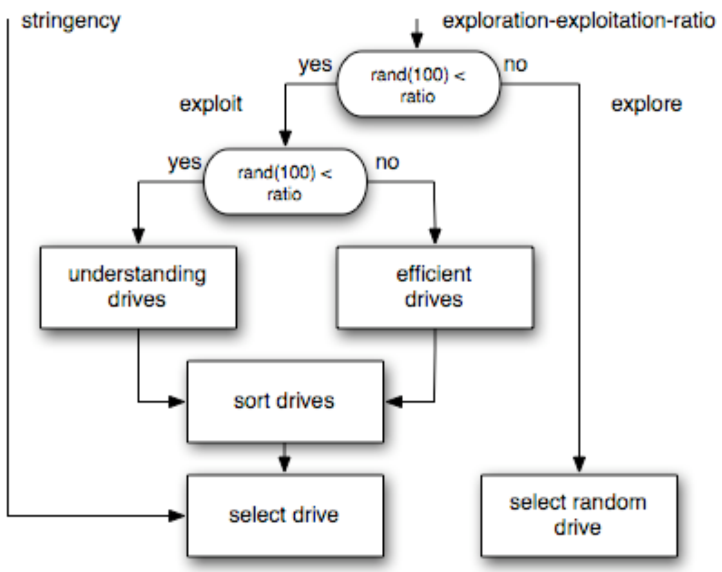

Figure 7: The exploration-exploitation-ratio acts as a probabilistic threshold for selecting the next drive.

In the current implementation, the drives-pool contains between 8 and 30 drives. The initial relationships are random with a density of 50 percent, while both orientation levels receive a random value between 40~60. The rationale is to provide initial momentum for change in either positive or negative directions. A random selection scheme is used, the chance for a drive to be selected being inverse proportional to the number of times it ran in the past. Thus, all drives get a chance to perform but not in any specific order.

At the beginning of the learning period, any drive can be selected because none has developed an efficient behavioural orientation. *Exploration* takes place: the pool of drives is sampled at random and the orientation levels are pushed up or down. When a clear contrast gradually emerges, we may decide to actually exploit the knowledge that was acquired online. So first we give many options a chance to develop while later on, the promising ones are applied. We use a probabilistic ranking scheme that proportionally conditions efficient drives to be selected. Once the learning period is finished, the genetic operators are applied. The drive's efficiency-level is viewed as equivalent to fitness. The newly bred generation will thus reflect the knowledge gathered during the learning period.

This situation described above is known as the dilemma of opting for *exploration* or *exploitation*. A global parameter is introduced: the *exploration-exploitation-ratio* (probabilistic selector, 0 = only exploration, 100 = only exploitation, 50 = equal chances) -- the hypothesis is that its value should be congruent with *changes* in responsiveness of the human performer. The level of *human-responsiveness* is another systems variable computed from the normalised sum of 3 items: quality and quantity of the current contents of working-memory (a FIFO structure holding the last 32 MIDI events input by the human performer) and the *current-no-input-gap* (the time since the last event was input). We keep track of the previous and current levels of human-responsiveness. Now, it is imagined that increased responsiveness signifies a readiness to connect with the current context from the part of the human performer; consequently opportunistic *exploitation* of what happens to be available should be favoured over the uncertainty associated with adventurous exploration.
Otherwise, when the interval of current minus previous human-responsiveness is negative, we reason that the human performer is temporarily loosing attention; therefore, we should increment chances for exploration to potentially induce renewed interest in the human performer. In short, the exploration-exploitation-ratio continuously tracks changes in human-responsiveness as to yield a probabilistic threshold between 0 and 100.

Yet one additional concern will guide the selection of a particular drive: its *understanding-level*. The understanding-level of a prospective drive is addressed as this level reflects the short-term efficiency in terms of social conformity during interaction as explained in the previous section. Figure 7 illustrates the twofold application of the exploration-exploitation-ratio in picking the most advantageous drive. The higher the ratio, the more chance for exploitation *and* the more chance for selection of understanding drives. Lower ratios shift towards selection of drives based on efficiency while very low ratios promote exploration i.e. random selection. Before selection, sorting according to respectively positive understanding-level or positive efficiency collects fit drives. The *stringency* parameter (0~100) further constraints the selection process. Given a stringency value of zero, any fit drive is subject to selection irrespective of its fitness level. Given a stringency value of 100, only the fittest drive is a candidate. Values in between exercise variable pressure on the selection process.

7. Genetic optimisation of motivations

Genetic methods are a hot item in computer music research nowadays (Miranda and Biles, 2007). For our purpose, in terms of evolution, the fitness of a drive is equivalent to its efficiency. Genetic optimisation aims to modify the relationships inside the drives to make them better adapted to the variable external pressures i.e. the *changes* in human-machine similarity and the changes in quality and quantity of the material provided by the human performer.

Breeding the next population is organised as follows:

- the current drives population is sorted according to fitness
- the two fittest drives are considered parents
- a new population is created: the relationship-vectors of both parents are considered genotype and new vectors are computed using a single point crossover operator

- a small amount of mutation is applied to all drives in the new population, mutation level is lower than 5% in most experiments
- all instance variables of every new drive are reset and the integration- and expression levels are set to a random centre value between 40 and 60.

In the current implementation, the moments of genetic activity are timed explicitly. Genetic operators should take action when all drives had a chance to build up enough experience during interaction, all drives must be applied at least a few times and gather expertise from the learning process as described above. The breeding-cycle, therefore, is taken as a multiple of the number of drives in the drives-pool. Given a population of 16 drives in the drives-pool, a typical breeding-cycle is 16 * 5 or 80 process cycles. This implies that, on the average, every drive has a chance to be applied 5 times.

8. Experimental results

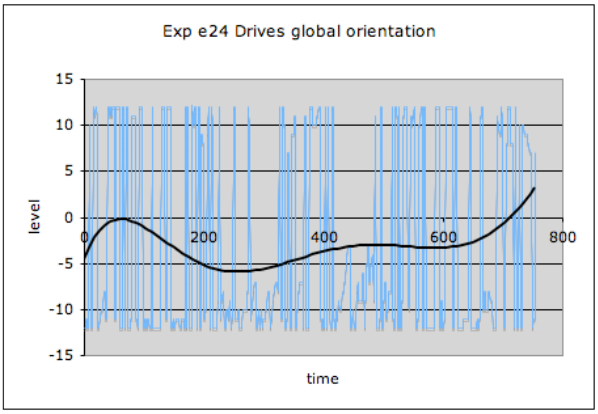

Figure 8: evolution of the drives-pool global orientation, integration (positive values), expression (negative values)

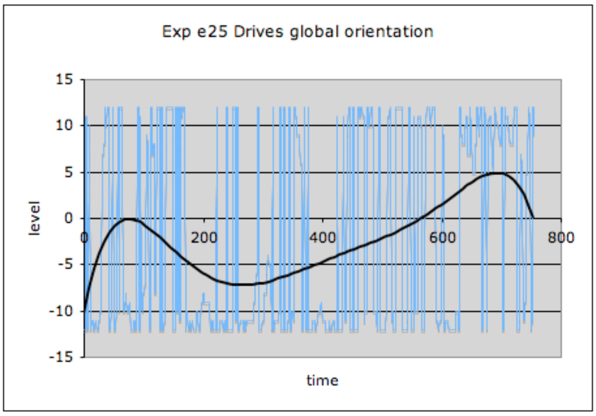

Figure 9: evolution of the drives-pool global orientation, integration (positive values), expression (negative values)

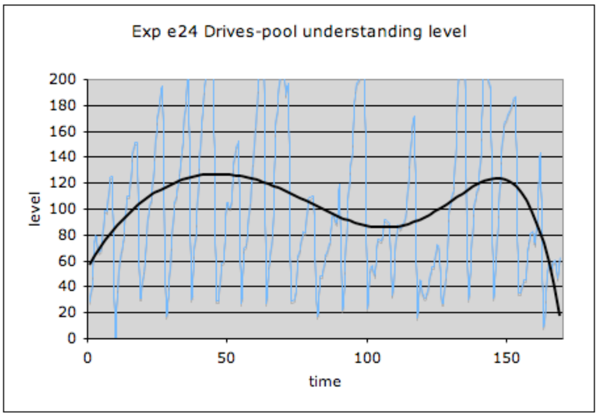

Figure 10: evolution of the drives-pool understanding-level in experiment e24

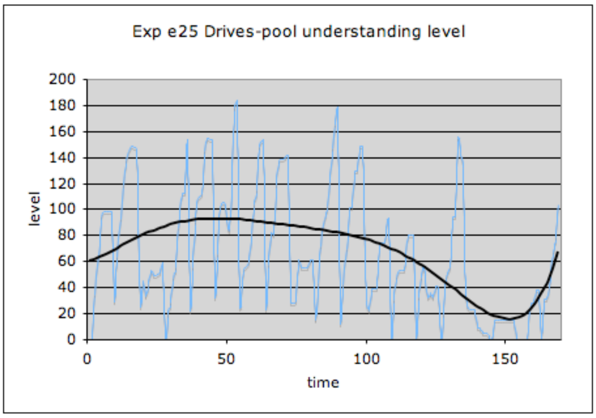

Figure 11: evolution of the drives-pool understanding-level in experiment e25

We conducted a substantial number of experiments to investigate the potential of motivation-driven interaction. Each experiment monitors a considerable number of systems parameters (exactly 50) and saves their momentary values to disk. This yields a large data file documenting an interactive session; the data is further subject to various types of off-line analysis and visualisation. However, only a few aspects directly relating to motivations and global behaviour are included here. Two experiments offer a chance to compare behavioural development in two independent experiments; e24 and e25. The population size is 8 drives, the breeding-cycle is only 16 and the total number of process-cycles is 320 – therefore, these experiment contain exactly 20 epochs of genetic evolution. The experiments respectively take 38'24" and 44'23".

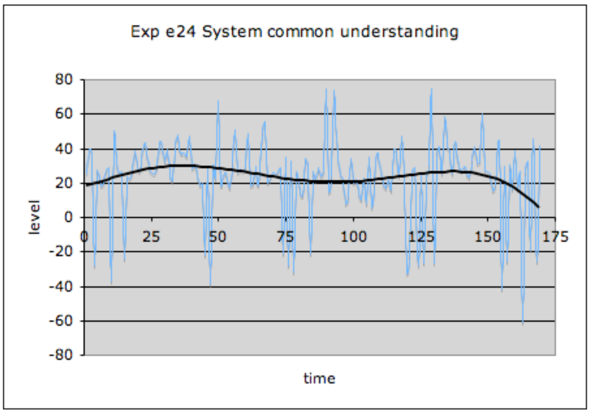

Figure12: evolution of the level of system common understanding in experiment e24. Agreement (positive values), conflict (negative values)

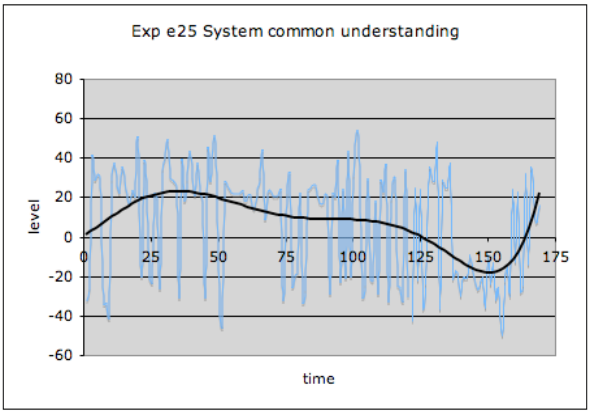

Figure 13: evolution of the level of system common understanding in experiment e2. Agreement (positive values), conflict (negative values)

Figures 9 and 10 document drives global-orientation i.e. the overall result of the competing forces of integration and expression. Both show an incremental data profile, the machine develops momentum to favour integration.

Figures 10 and 11 show the evolution of the drives-pool understanding-level. The polynomial reveals wave-like behaviour; the understanding-level is subject to slow oscillation.

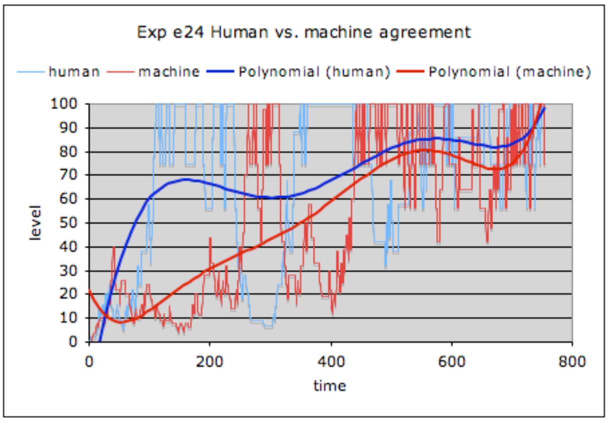

Figure 14: Evolution of the levels of human vs. machine agreement in experiment e24

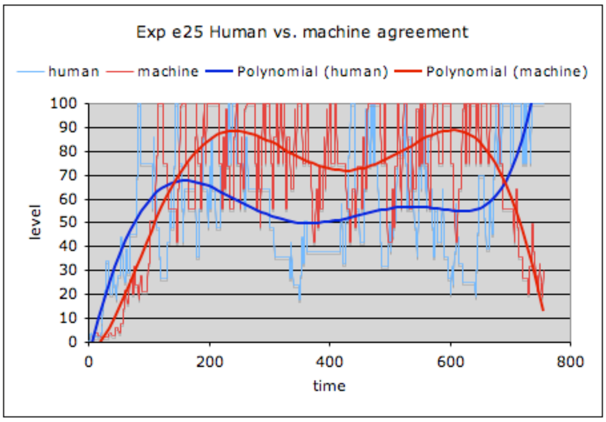

Figure 15: Evolution of the levels of human vs. machine agreement in experiment e25

Figures 12 and 13 documents system behaviour at the highest level of generalization. System common understanding shows areas of relative stability and areas of oscillations constrained between specific upper and lower levels. Both experiments maintain a significant interaction climate characterized by man-machine *agreement*. A strong correlation exists between the incremental nature of the drives-pool understanding level (figures 10 and 11) and a similar profile in system common understanding level.

All experiments keep track of two additional systems variables: human-agreement and machine-agreement (figures 14 and 15). *Human-agreement* is updated each time the segmentation algorithm considers the human performer 'just finished playing'. *Machine-agreement* is updated every time the machine just finished playing its most recent response. The rationale is to have quantities that respond immediately (in contrast to accumulated changes inferred at the end of a learning period) to changes in the musical distance between man and machine. The levels of agreement are scaled up or down according to the interval in similarity. For instance, human-agreement goes up when the most recent human input sequence manages to be more similar to the current machine output than the previous human input sequence. In contrast, given an increase musical distance, human agreement it is scaled down. Exactly the same principle articulates machine-agreement.

Remarkably, figure 14 and figure 15 (somewhat less so) demonstrates a steady increase in agreement for both man and machine. When both values are highly similar *and* of a high value there is evidence that both parties managed to develop musical functionality to perform in a common effort with shared objectives. In other words, man and machine expose adaptive behaviour. As a result, these observations reveal emergent goal directedness as a side effect of leaning in the drives.

9. Conclusion

In conclusion, a drive implements machine motivations -- a facility to generate temporal machine suggestions. The drive object advocates a method to avoid explicit design, which typically characterizes conventional mapping procedures in interactive systems design. In contrast, a drive is a flexible data structure that adapts its integration and expression levels according to its relationships and the accommodation of external changes during the process of interaction itself. In addition, a drive features learning components: long-term efficiency and short-term understanding.

The fluctuations in system common understanding -- representing the top-most impression of the global systems behaviour – reveal the dynamic qualities of the interactive process. The systems' implicit intention is to develop networks and musical processing functions that are optimised towards generating agreement between man and machine as musical partners. *Agreement* implies that both man and machine show competence to develop functionality that contributes to sustaining the current system-global-orientation, irrespective of whether it is integration or expression. The experiments reported here show strong evidence that the current systems architecture manages to support such functionality successfully.

A short note on implementation: a first version was written in Macintosh Common Lisp using MIDI functionality provided by Common Music (Taube 2005). The most recent version is written in SuperCollider 3.2 (McCartney 1996) in a style of object-oriented programming, the complete system is comprised of about 100 classes.

Acknowledgements

I am grateful to Eduardo Miranda and everyone at ICCMR for discussions and constructive feedback. In addition, I thank George Lewis, Robert Rowe, François Pachet and Joel Chadabe who contributed to shaping the ideas expressed in this paper.

References

Beyls, Peter: 2005 *Evolving Adaptive Sensors in a Synthetic Listener*, Proceedings of the International Computer Music Conference, Barcelona, Spain,

Beyls, Peter: 2007 *Interaction and Self-Organization in a Society of Musical Agents*, Music & Artificial Life Conference, Proceedings of the European Conference on Artificial Life, Lisbon, Portugal

Cope, David: 2005 *Computer models of Musical Creativity*, MIT Press, Cambridge, MA

Eysenck, Hans: 1973 *The Inequality of Man*, Temple Smith, London, UK

Goldberg, David: 1989 *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA

McCartney, James: 1996 *SuperCollider: A New Real time Synthesis Language*, Proceedings of the ICMC, Hong Kong

McCorduck, Pamela: 1991 *Aaron's Code: Meta-Art, Artificial Intelligence and the work of Harold Cohen*. WH Freeman and Co. New York

Maturana, H and Varela, F: 1992 *The Tree of Knowledge, The Biological Roots of Human Understanding*, Shambhala Publications

Miranda, Eduardo and Biles, John (Eds.): 2007 *Evolutionary Computer Music*, Springer, Heidelberg, Germany

Mook, DG: 1987 *Motivation: the organization of action*. WW Norton and Co. Inc. New York

Sims, Karl: 1994 *Evolving 3D morphology and behavior by competition*, Proceedings of Alife IV, Brooks and Maes (Eds.), MIT Press/Bradford Books

Steels, Luc: 1995 *A selectionist approach to behavioural development*, AI Memo 95-06 VUB, Brussels, Belgium

Sutton, Richard and Barto, Andrew: 1998 *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA

Taube, Heinrich: 2004 *Notes From The Metalevel Introduction to Algorithmic Music Composition*, Taylor & Francis Group plc, London, UK