Vers des outils de création artistique accessibles inspirés des fourmis

Romain Clair, Nicolas Monmarché, Mohamed Slimane
Université François Rabelais Tours
Laboratoire d'Informatique
64 avenue Jean Portalis
37200 Tours, France

Draft paper of paper published in Technique et science informatiques no 3-4/2013, 313-337

Résumé

Les travaux présentés dans cet article concernent la conception d'outils informatiques de création artistique, accessibles aux personnes handicapées et comportant des systèmes de production d'art génératif. C'est ainsi qu'un programme « instrument de musique virtuel » a été conçu, développé et évalué pour expérimenter ces idées. Il s'agit d'un logiciel accessible, en particulier aux personnes en situation de handicap, permettant de jouer de la musique interactivement. Cet outil, basé sur un algorithme de colonie de fourmis artificielles, est ainsi doté d'un système génératif lui permettant d'improviser en collaboration avec l'utilisateur. Des tests de ce prototype par des utilisateurs finaux, permettent d'évaluer cette approche.

This work deals with the design of accessible softwares for artistic creation, embedding generative art production systems. A virtual music instrument software was designed, developed and tested to experiment these ideas. This software, accessible for disable people, allows to play live music. This tool, based on an artificial ant colony algorithm, can improvise with its user, using a generative system. Tests have been conducted with real users are used to evaluate this approach.

1 Introduction

Dans le cadre de nos travaux sur les algorithmes bioinspirés, tels que les algorithmes basés sur le comportement des fourmis, nous nous sommes intéressés à traduire les comportements de fourmis en manifestations artistiques telles que la génération d'images [2] et de musique [23].

À partir de ces travaux, nous pensons que l'utilisation de ces méthodes est pertinente pour la conception d'outils informatiques de création artistique ac-

cessibles au plus grand nombre de personnes, et en particulier aux personnes en situation de handicap moteur, sensoriel ou cognitif.

L'art génératif, que l'on peut définir comme la création automatique d'œuvres artistiques à l'aide de procédés, en particulier algorithmiques, est au croisement de l'art et de la science et, à ce titre, implique à la fois des artistes et des scientifiques.

L'apport de l'informatique dans ce domaine n'est pas récent. En effet, sans chercher à construire un historique exhaustif, on peut citer [15] qui entreprennent de mettre à profit la puissance de calcul des ordinateurs en développant un système de composition automatique de musique. On peut aussi citer le compositeur Iannis Xenakis qui développe dans son ouvrage « Musiques formelles » [27] une analyse du processus de composition musicale et propose des systèmes, aléatoires dans un premier temps, puis basés sur des chaînes de Markov [16]. Mais les chaînes de Markov ne sont qu'une des nombreuses méthodes utilisées dans l'art génératif. Si on reste dans le domaine probabiliste, on peut citer *OMAX* de [1], qui est un système basé sur des arbres des suffixes probabilistes. Dans un autre formalisme, les systèmes de Lindenmayer, dit L-System [26], sont des grammaires formelles qui permettent de modéliser la croissance des plantes : ils sont utilisés pour la génération d'images numériques de plantes, tels qu'illustrés par les travaux de [17]. On peut également citer les algorithmes évolutionnaires comme outils de création avec par exemple *GenJam* de [4] qui est un système d'improvisation de solo de jazz utilisant des algorithmes génétiques interactifs. Cette interaction mène en particulier à la notion de co-improvisation où le système artificiel interagit avec le musicien. Dans cette catégorie, qui nous concernera dans la suite, on peut citer les travaux de [25] avec son *continuator* ainsi que ceux de [6] qui proposent tous deux des systèmes de co-improvisation.

Les méthodes utilisées pour générer automatiquement des œuvres artistiques étant de plus en plus nombreuses, on assiste à l'apparition d'ouvrages dédiés à ces questions : des ouvrages tels que « Evolutionary Computer Music » [18] ou « Algorithmic composition » [24] dressent un bilan des progrès faits dans le domaine de la génération automatique de la musique.

La notion d'accessibilité considérée dans les travaux présentés dans cet article concerne tout d'abord les personnes en situation de handicap. On distingue généralement les handicaps physiques, des handicaps mentaux.

Les handicaps physiques touchent aux capacités motrices ou sensorielles. D'un point de vue sensoriel, ils englobent toutes les difficultés liées à la perception de l'environnement, qu'elles soient visuelles, auditives, olfactives, gustatives ou tactiles. La perception classique d'une application informatique passant principalement par les deux premiers de ces sens, ce sont donc à ceux-ci que l'on s'intéresse en priorité ici. Les handicaps moteurs, quant à eux, recouvrent la très vaste étendue des problèmes liés aux mouvements du corps et à la fatigue associée. L'utilisation d'un ordinateur concerne principalement les membres supérieurs, de l'épaule aux doigts, aussi nous attacherons-nous plus spécifiquement aux handicaps de ces parties du corps. Il existe bien d'autres handicaps, tels que la mutité, les déficiences viscérales, parfois même certaines déformations morphologiques qui bien que lourds de conséquences dans la vie quotidienne ne sont

pas des freins majeurs à l'usage de l'outil informatique.

Les handicaps mentaux sont liés aux déficiences intellectuelles. Cette catégorie regroupe une grande variété de difficultés qui peuvent affecter les capacités de raisonnement, de mémorisation, de concentration, de communication, etc. Nombre de ces handicaps rendent impossible l'utilisation d'outils informatiques. Il est pourtant des cas dans lesquels des adaptations peuvent permettre un accès à l'informatique. Ces outils participent, alors, au confort voire au développement des capacités des personnes atteintes.

Néanmoins, la notion d'accessibilité que nous considérons s'étend au-delà de la question du handicap, s'inspirant de celle avancée par Tim Berners-Lee, président du World Wide Web Consortium (W3C), dans le cadre du web :

> « Mettre le Web et ses services à la disposition de tous les individus, quels que soient leur matériel ou logiciel, leur infrastructure réseau, leur langue maternelle, leur culture, leur localisation géographique, ou leurs aptitudes physiques ou mentales. » [3]

Ainsi, l'accessibilité telle que nous la concevons cherche aussi à palier les difficultés liées à la langue, au contexte culturel, aux connaissances et aux moyens techniques et économiques des utilisateurs.

2 Les objectifs

L'objectif de ces travaux est de proposer des outils informatiques pour améliorer l'accessibilité de la création artistique. Notre point de départ pour aborder cet objectif est de considérer les méthodes de production d'art génératif comme des techniques permettant cette amélioration. Afin d'expérimenter ces idées, y compris auprès d'utilisateurs finaux, nous avons poussé la conception et le développement jusqu'à obtenir un prototype logiciel fonctionnel.

L'art est un sujet particulièrement vaste. Nous avons choisi de commencer par nous intéresser à la musique, et plus précisément, à la musique improvisée. Il s'agit donc de jouer de la musique « live », sans se conformer à une composition préalable.

Il nous est apparu plus pertinent de placer l'utilisateur comme acteur principal de la création artistique, et de ne pas se substituer à lui. Les méthodes de production d'art génératif viennent donc épauler l'utilisateur-artiste dans sa performance, plutôt que de s'y substituer. Le système cherche à improviser conjointement à l'utilisateur. On parle alors de co-improvisation et l'un des objectifs, dans la conception du système de production automatique de musique, est de le forcer à interagir, en temps réel, avec l'utilisateur.

Les objectifs d'accessibilité de ce logiciel visent, dans un premier temps, le handicap, et particulièrement les handicaps moteurs qui constituent souvent un obstacle majeur dans la pratique de la musique. L'outil de création proposé étant un programme informatique, nous nous intéressons principalement aux déficiences des membres supérieurs. Ainsi les utilisateurs en incapacité d'utiliser leurs bras doivent pouvoir utiliser le logiciel. Nous nous intéressons aussi

aux personnes souffrant de déficiences cognitives, telles que des problèmes de mémoire ou de compréhension.

En ce qui concerne les handicaps sensoriels, nous distinguons le cas des aveugles et malvoyants, à qui nous assurons l'accès au logiciel, du cas particulier des handicaps auditifs. En effet, dans le cas de la pratique de la musique, les déficiences auditives nécessitent un traitement particulier. S'il s'agit évidemment d'un enjeu important et intéressant, il ne fait pas partie des objectifs de ces travaux.

La conception de l'outil que nous proposons prend également en compte l'objectif d'être accessible quels que soient la langue, les connaissances musicales et le niveau de maîtrise de l'outil informatique de l'utilisateur. L'aspect économique est aussi important et nous prenons en compte le fait de ne pas imposer de matériels ou de logiciels spécifiques ou particulièrement coûteux.

Il est donc question de concevoir, de développer et de tester un logiciel simulant un instrument de musique virtuel et accessible permettant de jouer de la musique « live » et improvisant un accompagnement interactif.

2.1 Accessibilité des interfaces

L'accessibilité de notre instrument de musique virtuel dépend en partie de l'accessibilité de ses interfaces. Pour l'affichage d'information, notre première approche consiste à rester le plus simple et sobre possible. L'utilisateur ne doit pas être submergé d'informations superflues. Cette première interface pourra être enrichie plus tard, en fonction des besoins identifiés auprès des utilisateurs tests. À terme, l'idéal étant d'obtenir une interface dont la complexité, les fonctionnalités offertes et le niveau d'aide associé s'adaptent automatiquement à l'utilisateur.

Pour les interfaces de contrôle de l'application, nous nous appuyons sur le principe de multimodalité qui consiste à toujours fournir plusieurs moyens différents et indépendants, de procéder à une action. Ainsi, si un utilisateur donné ne peut effectuer une action en utilisant un des moyens fournis, il doit pouvoir le faire avec un autre. Nous nous appuyons donc sur le clavier ou la souris, de façon indépendante, pour les deux raisons principales suivantes. La première est que ces périphériques disposent d'équivalents adaptés aux utilisateurs dans l'incapacité de les utiliser pleinement. Néanmoins, une attention particulière est apportée à offrir une utilisation simple et efficace de ces périphériques, y compris en cas de substitution : le contrôle du programme ne nécessite pas de mouvement rapide, de combinaison d'actions, de maintien de clic, etc. La seconde tient compte des considérations économiques liées à notre définition de l'accessibilité, aussi l'utilisation de ces périphériques grand public, donc peu coûteux est particulièrement appropriée.

2.2 Des connaissances embarquées

Un des objectifs que nous nous sommes fixés est de concevoir un logiciel ne nécessitant pas de formation musicale préalable. Le programme doit pouvoir

guider l'utilisateur, même néophyte. C'est pourquoi, le programme dispose de connaissances musicales embarquées. Ainsi, la notion de tempo est déduite de la musique produite par l'utilisateur. Il ne lui est pas demandé de choisir *a priori* le tempo auquel il va jouer. Les considérations harmoniques, peuvent être considérées comme des règles musicales et sont liées au contexte culturel et à la pratique personnelle de chacun. L'approche adoptée dans ces travaux consiste à imposer des contraintes fortes, dans un premier temps, quitte à les relâcher par la suite si le résultat obtenu n'est pas assez créatif. L'outil doit garantir la production d'une musique ni trop basique ni trop dissonante.

Du point de vue technique, aucun paramétrage n'est demandé à l'utilisateur qui n'a ainsi pas besoin de connaître l'informatique ou l'architecture technique particulière utilisée pour le logiciel. Les messages d'erreur ou situations bloquantes pouvant survenir sont traités automatiquement, et aucune information sur la couche technique/scientifique sur laquelle s'appuie le logiciel pour fonctionner ne vient interférer avec l'expérience musicale de l'utilisateur.

2.3 Une co-improvisation

L'apport à l'instrument de musique d'un système génératif doit permettre d'obtenir un résultat musical étoffé, y compris à partir d'actions réduites de la part de l'utilisateur. Néanmoins, pour que l'utilisateur demeure l'acteur de la création artistique en cours, il convient d'assurer une interactivité forte entre le système et l'utilisateur. De plus, puisqu'il s'agit de jouer de la musique « live », le système doit pouvoir générer son accompagnement en temps réel.

Nous avons décidé de baser notre système génératif sur un algorithme de fourmis artificielles, parce que ce sont des algorithmes rapides et très adaptés aux systèmes évoluant dans le temps. Les algorithmes de colonie de fourmis artificielles naissent du constat que, bien que constituée d'individus au comportement assez simple, une colonie de fourmis est capable de résoudre des problèmes complexes, distribués et évolutifs. On parle d'ailleurs d'intelligence collective. En effet, bien que ne disposant pas d'un contrôle centralisé, la colonie dans son ensemble montre plus de capacités que la simple somme de ce que les individus qui la composent sont capables de faire. Ces constatations sont développées et adaptées à des problématiques en informatique depuis le début des années 1990 [5, 19, 20].

Cette intelligence collective est basée sur les multiples interactions entre les membres du groupe. Ces interactions pouvant être directes, telles que des communications d'individus à individus, ou indirectes, telles que la communication chimique *via* des dépôts de phéromones dans l'environnement. Cette communication indirecte est appelée stigmergie.

Des modèles mathématiques probabilistes permettent de simuler le comportement des fourmis. Ils utilisent des notions de rétroactions positives, notamment *via* le renforcement des pistes de phéromones, ou négatives comme l'évaporation de ces mêmes phéromones. Plusieurs modèles de comportements existent, celui que nous utilisons s'appuie sur les mécanismes de déplacement collectif des fourmis (le recrutement de masse). Dans les algorithmes imitant ce comportement,

on utilise un ensemble de fourmis qui se déplacent dans un environnement formalisé. Dans la suite, nous formaliserons cet environnement sous la forme d'un graphe. À chaque pas de temps, chaque fourmi se trouve sur un sommet de ce graphe (le déplacement des fourmis est donc discrétisé). La communication au sein de la colonie s'effectue par des dépôts de phéromones sur ce graphe. En effet, lors de leurs déplacements, les fourmis déposent une certaine quantité de phéromones sur les arcs sur lesquels elles passent.

Pour se déplacer, une fourmi est amenée à faire des choix entre les différents arcs qui s'offrent à elle. Elle s'intéresse alors aux arcs sortants du sommet où elle se trouve. En effet, on considère qu'une fourmi ne s'intéresse qu'aux informations locales pour faire ses choix, et qu'elle ne prend pas de décision à long terme. Elle procède ensuite à un tirage au sort entre ces différentes directions. Ce tirage n'est pas équiprobable et les probabilités de choisir une direction dépendent des quantités de phéromones trouvées sur chacun des sommets voisins. Plus la quantité est importante, plus la probabilité de choisir ce chemin est forte. Ainsi, une fourmi a plutôt tendance à suivre les directions ayant une forte concentration en phéromones que les autres. Ce choix peut être défini, dans le cas de deux chemins A et B, par la probabilité suivante [5]

$$P_A^{(i+1)} = \frac{(k + A_i)^n}{(k + A_i)^n + (k + B_i)^n} = 1 - P_B^{(i+1)} \tag{1}$$

avec :

- $P_A^{(i+1)}$ (resp. $P_B^{(i+1)}$) : probabilité de choisir le chemin A (resp. B) à l'instant $(i + 1)$;
- k : degré d'attraction d'un chemin non marqué ;
- A_i et B_i : quantité de phéromones sur le chemin A et B à l'instant i ;
- n : paramètre de l'algorithme, permettant d'ajuster l'influence des phéromones.

La fourmi dépose des phéromones sur son passage, ainsi : $A_{i+1} = A_i + 1$ si la fourmi i a choisi le chemin A, $A_{i+1} = A_i$ sinon.

Utilisé dans un environnement avec une source et une cible, telles que le nid de la colonie et une source de nourriture, par exemple, le comportement général de cet algorithme est d'avoir, dans un premier temps, des fourmis qui partent dans tous les sens. Puis, progressivement, les fourmis utilisent toutes, ou presque, le même chemin. Ce chemin a une forte chance d'être le plus court chemin entre la source et la cible. En effet, le grand nombre de fourmis renforce le petit avantage – éventuel mais plus probable – à parcourir un chemin plus court.

Ce comportement permet de résoudre des problèmes d'optimisation combinatoires, notamment ceux consistant à rechercher un plus court chemin dont le problème du voyageur de commerce est l'exemple le plus connu. Le travail de [12] sur *Ant-system* (AS) constitue le point de départ de nombreux algorithmes basés sur des colonies de fourmis, appellés ACO pour *Ant Colony Optimization* [11, 13, 19].

Table 1 – Tableau des notes disponibles

Notes	Numéro midi	Octave	Numéro midi
si♭	58	si♭	70
do	60	do	72
ré	62	ré	74
mi♭	63	mi♭	75
fa	65	fa	77
sol	67	sol	79
la	69	la	81
si♭	70	si♭	82

3 Réalisation

3.1 Utilisation de la norme midi

Le développement de l'instrument de musique s'appuie sur la norme midi, pour permettre notamment une approche symbolique de la musique et une gestion des contraintes temporelles. Cette norme nous permet de manipuler directement des notes. Chaque note est caractérisée par sa hauteur, exprimée en nombre de demi-tons et sa durée, exprimée en nombre de *ticks*. Le tableau 1 référence les valeurs utilisées dans la norme midi pour représenter les hauteurs des notes utilisées pour l'instrument.

Le *tick* est l'unité élémentaire de mesure du temps dans la norme midi, sa durée dépend du tempo. L'entité midi permettant la gestion du temps est appelée un séquenceur. Il permet de jouer des séquences complètes de notes, c'est-à-dire d'évènements midi ordonnés dans le temps. La cadence à laquelle les *ticks* sont produits dépend de deux paramètres. Le premier est le tempo qu'on associe au séquenceur. Il peut être exprimé en battements par minute (*i.e.* : noires par minutes). Le second paramètre est la résolution, exprimée en *tick* par battement.

L'utilisation d'un séquenceur permet ainsi de s'affranchir de la gestion des contraintes temporelles pour la restitution des sons. Le synthétiseur midi qu'on lui associe permet quant à lui de s'abstraire des problèmes de production du son.

3.2 Les contraintes musicales

La génération de musique par des fourmis a été expérimentée dans un cadre de « musique expérimentale » [22, 14, 21], et les musiques obtenues étaient librement détachés des contingences harmoniques de la musique classique. Dans le cas du travail présenté ici, l'objectif artistique est différent. En effet, il s'agit de permettre, notamment à des utilisateurs novices en musique, l'obtention de mélodies relativement plaisantes, ce qui implique d'imposer des contraintes fortes

aux possibilités offertes par les fourmis musiciennes. Ces contraintes peuvent être vues comme des connaissances musicales incorporées au programme ou comme des guides qui tentent d'éviter les « fausses notes ».

Ainsi, l'instrument proposé est diatonique. Plus précisément, il offre une gamme de si♭ majeur s'étalant sur deux octaves, soit 15 notes, reprises dans le tableau 1. L'éventail de possibilités créatrices en est ainsi réduit, mais il n'est pas dans les objectifs premiers de cet outil de servir à des musiciens chevronnés, mais plutôt de permettre à des néophytes d'obtenir facilement un résultat satisfaisant. Néanmoins, l'approche adoptée dans la conception du logiciel permet d'étendre facilement les possibilités offertes, par exemple proposer d'autres gammes, plus de notes, passer sur une gamme chromatique, etc.

Les représentations rythmiques des notes jouées sont calculées automatiquement par rapport à un tempo donné. Ainsi, la notion de tempo est totalement transparente pour l'utilisateur. L'instrument fonctionne en interne à un tempo de 60 battements par minute. La conversion est donnée par :

$$D_r = \frac{D_m \times T \times R}{60000} \tag{2}$$

où :

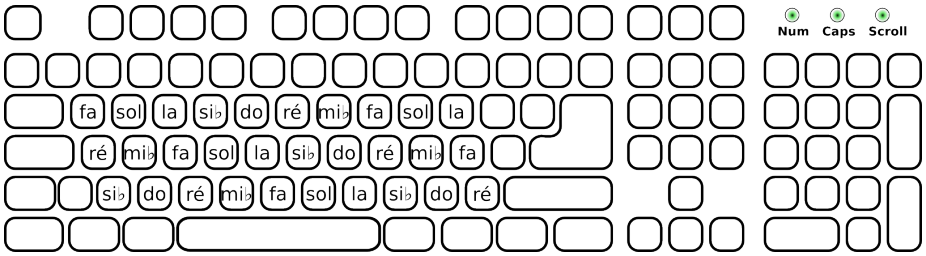
- D_r : durée en nombre de ticks
- D_m : durée en millisecondes
- T : tempo
- R : résolution du séquenceur.

3.3 Les interfaces proposées

Le programme dispose d'une interface graphique minimale. L'un des objectifs est de s'abstraire de la technique. Il n'y a ainsi rien à configurer ou régler. N'ayant donc rien d'important à afficher, dans un premier temps, l'option choisie consiste à n'utiliser qu'une fenêtre vide pour y déplacer la souris. On pourrait évidemment afficher des retours graphiques des interactions de l'utilisateur avec le système. Cependant, ces éléments graphiques pourraient influencer l'évaluation de la production sonore du logiciel : il ne s'agit pas d'avoir des utilisateurs satisfaits utilisant l'outil sans le son ! Conséquemment, le seul retour que l'application fournit est sonore.

Conformément aux préconisations de multimodalité, deux moyens distincts sont fournis pour contrôler l'application. Le clavier est le premier de ceux-ci. Il s'inspire du fonctionnement d'un clavier de type piano. Cette interface relativement intuitive consiste à associer à la pression d'une touche le déclenchement d'une note, celle-ci s'arrêtant dès que l'on relâche la touche. La répartition des touches sur le clavier part du constat que le clavier informatique nous offre plus de touches que ce dont on a strictement besoin. On peut donc avoir plusieurs touches pour produire le même son. La disposition de ces répétitions vise à faciliter les intervalles courants. Les touches qui se suivent de gauche à droite permettent de monter la gamme note à note. Au-dessus de chaque note se trouve sa tierce dans la gamme. Les trois touches alignées verticalement forment donc

FIGURE 1 – Schéma de la disposition des notes sur le clavier



une triade classique tonique - tierce - quinte, et offrent une harmonisation par des accords de trois sons de la gamme. La figure 1 récapitule la disposition des notes sur le clavier.

Le second moyen de contrôle de l'application s'appuie sur un dispositif de pointage. Le postulat de départ est le suivant : il s'agit d'un instrument virtuel. Les interfaces de contrôle peuvent, certes, reproduire le fonctionnement d'instruments réels, mais elles peuvent aussi s'inscrire dans un cadre plus original et explorer les possibilités d'expression d'une interface novatrice. Les contraintes adoptées cherchent à limiter la fatigue et assurer un maximum de compatibilité en cas de substitution par un périphérique adapté. Pour ce faire, les boutons ne sont pas utilisés et seul le déplacement du pointeur est pris en compte.

L'interface de type « souris » s'inspire de la réalité physique du son, où de grandes vibrations produisent des sons graves, quand des vibrations plus courtes en produisent de plus aigus. Le fonctionnement se base sur la distance parcourue par le pointeur. On récupère la position d'origine du pointeur sur la fenêtre et lorsqu'il se déplace, on mesure la distance qui le sépare de cette position d'origine. Tant que le pointeur s'éloigne, il ne se passe rien, mais dès que la distance depuis la position d'origine précédemment définie diminue, une note est déclenchée. La hauteur de cette note dépend directement de la distance parcourue : plus cette distance est grande, plus la note est grave. A chaque note jouée, le point de départ du pointeur est redéfini à la position actuelle du pointeur et on commence à préparer la note suivante.

Les notes les plus graves s'obtiennent donc en parcourant la distance la plus longue possible, soit la diagonale de la fenêtre du programme. L'algorithme 1 explicite les détails de ce fonctionnement.

3.4 Le système de co-improvisation

Un des objectifs principaux de ces travaux est de fournir un système de co-improvisation, pour accompagner l'utilisateur. Plus précisément, il s'agit de générer, en parallèle de la production de l'utilisateur, des phrases musicales s'accordant avec cette dernière. Nous présentons ici les principes généraux, les détails pouvant être consultés dans [8].

Algorithme 1 Algorithme de suivi des déplacements de la souris

1: distanceMax $\leftarrow \sqrt{(\text{tailleFenetreX})^2 + (\text{tailleFenetreY})^2}$
2: tessiture $\leftarrow$ Instrument.recupereTessiture()
3: origineX $\leftarrow$ Position de la souris en X
4: origineY $\leftarrow$ Position de la souris en Y
5: distance $\leftarrow 0$
6: **boucler**
7: positionX $\leftarrow$ Position de la souris en X
8: positionY $\leftarrow$ Position de la souris en Y
9: nouvelleDistance $\leftarrow \sqrt{(\text{origineX} - \text{positionX})^2 + (\text{origineY} - \text{positionY})^2}$
10: **si** nouvelleDistance $>$ distance **alors**
11: distance $\leftarrow$ nouvelleDistance
12: **sinon**
13: jouerNote(arrondi$((1 - \frac{\text{distance}}{\text{distanceMax}}) \times \text{tessiture})$)
14: origineX $\leftarrow$ Position de la souris en X
15: origineY $\leftarrow$ Position de la souris en Y
16: distance $\leftarrow 0$
17: **finsi**
18: **fin boucle**

3.4.1 Principes de base

Le système de co-improvisation est basé sur un algorithme de colonie de fourmis artificielles. Imitant le comportement de déplacement collectif des fourmis, il utilise le parcours des fourmis comme générateur musical. Il s'inscrit en cela dans la continuité des travaux déjà évoqués sur l'étude des capacités créatives des algorithmes de fourmis artificielles (cf. [23]).

Le principe est d'associer une signification musicale à l'environnement dans lequel les fourmis évoluent. Cet environnement prend la forme d'un graphe pour lequel on associe une note à chaque sommet. Les fourmis se déplacent de sommet en sommet en empruntant les arcs du graphe. À son arrivée sur un sommet, une fourmi « joue » la note correspondante.

Le choix de la direction qu'une fourmi emprunte dépend uniquement de l'information locale qu'elle récolte, et s'effectue, sans contrôle centralisé. Cette information locale, équivalente aux phéromones pour les fourmis réelles, est stockée sur les arcs du graphe. Ces phéromones représentent l'attirance qu'exerce un chemin sur les insectes virtuels. Ces phéromones sont déposées par les fourmis à chaque fois qu'elles parcourent un arc. La valeur déposée par chaque fourmi lors de son passage est un des paramètres de l'algorithme, ces paramètres sont récapitulés dans le tableau 2. L'incrémentation de la quantité de phéromones est systématique, chaque fourmi augmente cette valeur à chacun de ses passages sur un arc. Néanmoins, le choix d'une direction à prendre conserve une part de hasard (on peut aussi parler d'exploration), capitale dans ce genre de méthode où l'espace de recherche est potentiellement immense. Le rapport entre la quantité de phéromones présente sur un arc et la quantité présente sur tous les arcs

environnants donne la probabilité qu'une fourmi choisisse cet arc. On la calcule à l'aide de l'équation (3) :

$$P(x) = \frac{S(x)}{\sum_{i=1}^{n}(S(i))} \tag{3}$$

où :

- x : un arc partant du sommet courant, avec $x \in [1..n]$
- n : le nombre d'arcs partant du sommet courant
- $P(x)$: la probabilité de choisir l'arc x
- $S(i)$: la valeur (intensité des phéromones) de l'arc i.

On peut noter que chaque fourmi choisit toujours une direction à chaque pas de temps. Elles sont contraintes de bouger. Aucune notion de distance n'apparaît dans cet algorithme, aussi les déplacements des fourmis sont-ils synchrones. Toutes les fourmis parcourent un arc à chaque pas de temps.

L'émergence d'une intelligence collective apparaît, dans le cadre d'une colonie de fourmis, par les interactions indirectes de plusieurs individus. Aussi, le nombre de fourmis présentes est un autre paramètre important de cette méthode. Dans le fonctionnement décrit précédemment, la multiplication des agents fourmis pose un problème. Plus il y a de fourmis, et plus il y a de notes jouées simultanément. Le risque de surcharge sonore, conduisant à produire un bruit désagréable, augmente très vite. Pour limiter ce phénomène, les fourmis sont divisées en deux catégories.

La première concerne les fourmis, dites musiciennes, qui jouent et construisent la mélodie en passant d'une note à l'autre dans le graphe. Leur nombre caractérise directement la capacité polyphonique de l'accompagnement automatique. Dans le cas d'une seule fourmi, l'accompagnement ne peut ainsi jouer qu'un son à la fois.

La seconde catégorie est constituée de fourmis silencieuses. Elles agissent exactement comme les autres sauf qu'elles ne jouent jamais de note. Leur apport consiste uniquement à favoriser l'émergence de l'intelligence collective. Le tableau 2, récapitulatif des paramètres du système, inclut les nombres de fourmis de chacune de ces deux catégories.

Un mécanisme d'évaporation vient compléter cet algorithme de fourmis. Il consiste à faire progressivement diminuer la quantité de phéromones présente sur chaque arc au fil du temps. Ce mécanisme permet à la colonie une meilleure adaptabilité aux changements de l'environnement, et participe donc grandement à la robustesse de ces méthodes. À chaque pas de temps, un certain pourcentage de la valeur stockée sur chaque arc disparaît. Ce coefficient d'évaporation vient donc compléter le tableau 2 des paramètres de l'algorithme.

3.4.2 Interactivité

La mise en place d'interactions entre le système et l'utilisateur repose sur deux mécanismes : d'une part, l'interaction se fait à travers la construction de la structure du graphe, et d'autre part, le cheminement des fourmis dans le graphe est influencé par l'utilisateur.

Le graphe de notes sur lequel les fourmis évoluent est construit par l'utilisateur En réalité, nous utilisons deux graphes simultanément : un graphe dont les sommets sont les hauteurs des notes et un autre contenant les durées des notes. Le système est muni d'un système de suivi de la musique jouée par l'utilisateur. À l'initialisation, les graphes sont vides, l'accompagnement automatique ne peut donc rien faire seul. Au cours du jeu de l'utilisateur, le programme récolte les notes jouées par l'utilisateur, et ajoute leur hauteur ou leur durée à chacun des deux graphes. Chaque fourmi possède une position dans chacun des deux graphes. Lorsque les graphes disposent d'au moins un sommet et que les fourmis ne sont pas encore placées, elles se voient chacune affectées à un sommet de chaque graphe au hasard. Les fourmis musiciennes fabriquent ainsi leur note en récoltant l'information harmonique du premier graphe et l'information rythmique du second. C'est l'association de cette hauteur et de cette durée qui déclenche une note.

Le système de co-improvisation se base donc uniquement sur les notes jouées par l'utilisateur. Il n'est capable que de variations autour des thèmes du musicien qui l'utilise. Cette séparation en deux graphes permet une amélioration des performances que nous aborderons plus tard, mais permet aussi, d'augmenter la créativité. En effet, les fourmis ont tendance à reproduire les schémas harmoniques et rythmiques joués par l'utilisateur, mais ont la capacité de les associer différemment.

Les deux graphes sont orientés, pour leur donner une signification plus musicale. En effet, les arcs de ces graphes modélisent implicitement les intervalles entre les notes. Or, en musique, un intervalle ascendant, tel que do – mi, n'est pas équivalent à un intervalle descendant, dans notre exemple mi – do. En effet, le premier est une tierce, l'autre conduit à la sixte inférieure. L'homme est d'ailleurs beaucoup plus sensible aux intervalles qu'aux notes elles-mêmes. Une expérience simple permet de s'en rendre compte. Il est aisé de reconnaître une mélodie, quelle que soit la hauteur à laquelle elle est jouée, alors que reconnaître d'oreille une note seule demande beaucoup d'entraînement. Plus que des ajouts de sommets aux graphes, ce sont plutôt des arcs orientés qui sont ajoutés, les sommets n'étant rajoutés que lorsqu'un arc pointe vers un sommet inexistant. L'initialisation de la quantité de phéromones présente sur un arc est un paramètre important de l'algorithme, puisqu'un arc de valeur zéro ne sera tout simplement jamais emprunté par une fourmi.

Le déplacement de l'utilisateur laisse une trace dans les graphes. Le second mécanisme consiste à doter le musicien de la capacité de guider les fourmis. Laissées à elles-mêmes, les fourmis ont tendance à créer au fil du temps un chemin préférentiel qu'elles finissent par emprunter très majoritairement. Ce phénomène est décrit plus en détail dans la section 4. Pour permettre à l'utilisateur de l'instrument d'influer sur le comportement des fourmis, il dépose lui aussi des phéromones sur les graphes pendant son jeu. La quantité qu'il dépose est une mesure de son impact sur le comportement de la colonie. Cette valeur s'ajoute donc aux paramètres du tableau 2.

Algorithme 2 Algorithme de suivi de l'utilisateur

```
1: /* Initialisation */
2: sommetCourant(hauteur) ← null
3: sommetCourant(durée) ← null
4: grapheHauteur ← null
5: grapheDurée ← null
6: pour Chaque note jouée faire
7:    On récupère la note jouée,
8:    correspondant aux nouveaux sommets dans chaque graphe
9:    pour Chaque graphe faire
10:      si sommetCourant n'est pas dans le graphe alors
11:         graphe.ajoutSommet(nouveauSommet)
12:      sinon
13:         arcCourant ← arc(sommetCourant, nouveauSommet)
14:         si arcCourant  existe alors
15:            arcCourant.valeur ← arcCourant.valeur + depotUtilistateur
16:         sinon si nouveauSommet  existe alors
17:            arcCourant ← graphe.ajoutArc(sommetCourant, nouveauSommet)
18:            arcCourant.valeur ← arcCourant.valeur + depotUtilistateur
19:         sinon
20:            graphe.ajoutSommet(nouveauSommet)
21:            arcCourant ← graphe.ajoutArc(sommetCourant, nouveauSommet)
22:            arcCourant.valeur ← arcCourant.valeur + depotUtilistateur
23:         finsi
24:         sommetCourant ← nouveauSommet
25:      finsi
26:   fin pour
27: fin pour
```

Du point de vue algorithmique, cette interaction consiste à considérer l'utilisateur comme une fourmi qui possède son autonomie dans ses déplacements mais qui dépose des phéromones de façon tout à fait similaire aux autres fourmis. L'algorithme complet de suivi de l'utilisateur, permettant de construire les graphes et de déposer des phéromones est détaillé dans l'algorithme 2.

3.4.3 Contraintes du temps réel

L'objectif de ce projet est de permettre de l'improvisation musicale, ce qui signifie donc, de jouer de la musique « live ». Cette approche impose des contraintes supplémentaires que ne rencontrent pas les outils de composition. Le maintien de la latence, c'est-à-dire le délai entre l'action qui déclenche une note et la production effective de la note, à une très faible valeur est de première importance dans ce cadre. En musique, on considère qu'une latence de l'ordre de la dizaine de millisecondes est correcte. Lorsqu'elle dépasse 100 ms, la gène occasionnée empêche de jouer.

Table 2 – Paramètres de l'algorithme

Paramètres	Notation
Nombre de fourmis	N_m
Nombre de fourmis silencieuses	N_s
Dépôt standard	τ_f
Dépôt de l'utilisateur	τ_u
Quantité initiale	τ_i
Coefficient d'évaporation	ρ

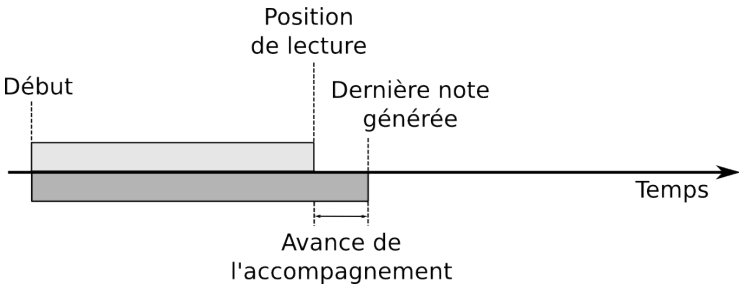
Table 3 – Taille des graphes

Hauteurs	Durée	Nombre de sommets		Nombre d'arcs	
		1 graphe	2 graphes	1 graphe	2 graphes
1 octave diatonique (8)	5	40	13	1 600	89
2 octaves diatoniques (15)	96	1 440	111	2 073 600	9 441
2 octaves chromatiques (23)	96	2 208	119	4 875 264	9 745
Maximum en midi (128)	128	16 384	256	268 435 456	32 768

Pour respecter la politique d'accessibilité du logiciel, et notamment la possibilité pour le programme de s'exécuter sur une machine quelconque, nous faisons le choix de ne pas utiliser de système garantissant des contraintes temps réel. En l'absence de système d'exploitation véritablement temps réel sur lequel nous appuyer, le programme doit donc faire de son mieux pour s'exécuter le plus rapidement possible.

L'algorithme de fourmis proposé ici est relativement simple et n'implique pas de disposer d'un grand nombre de fourmis. La taille des structures de données peut cependant induire des calculs plus coûteux, c'est pour cette raison de l'utilisation de deux graphes au lieu d'un seul a été envisagée. Le logiciel permet de jouer sur deux octaves d'une gamme diatonique, soit 15 hauteurs de notes possibles. Le graphe de notes fait une taille maximum de 15 sommets et 225 arcs. Le graphe des durées peut être un peu plus grand. Les durées sont mesurées en nombre de ticks. La résolution fixée dans le logiciel est de 24 ticks par noire. Cette valeur est calculée pour permettre de jouer des triples-croches, d'une valeur d'un huitième de noire, ainsi que des triolets d'une valeur d'un tiers de noire. En limitant la durée maximale des notes à celle d'une ronde, soit quatre noires, on obtient déjà un graphe maximal de 96 sommets et jusqu'à 9 216 arcs. Si les graphes n'étaient pas séparés, on pourrait avoir à manipuler 1 440 sommets reliés par plus de 2 millions d'arcs, et ce malgré les contraintes fortes appliquées aux possibilités musicales de notre instrument. Dans un cadre plus général, le tableau 3 donne une idée des tailles de graphes pour quelques exemples.

FIGURE 2 – Schéma du mécanisme d'avance de l'accompagnement sur la lecture



Le moteur génératif est exécuté en parallèle de l'instrument, il n'est bien sûr pas question de jouer chacun à son tour. Le parallélisme est mis en œuvre par des *threads* mais cette utilisation de *threads* crée une difficulté supplémentaire. En effet, le système de co-improvisation construit, à sa vitesse, des notes en se basant sur les graphes créés par l'utilisateur. Celui-ci enrichit les graphes au fur et à mesure qu'il joue. Il se trouve donc deux problèmes envisageables. Soit, l'accompagnement automatique prend du retard et ne génère pas les notes assez rapidement par rapport à la vitesse de lecture. Soit, il génère des notes trop vite et n'est plus en accord avec l'état actuel du graphe.

Pour le premier problème, la solution est effectivement déjà traitée, il faut être rapide. Le logiciel a néanmoins été muni d'un mécanisme de recalage, en cas de retard. Ainsi, si l'accompagnement automatique n'est pas assez véloce, les notes générées sont jouées à une date antérieure à la position actuelle de lecture du séquenceur. Dans ces cas-là, la date de début de la prochaine note générée est calée sur la position de lecture actuelle du séquenceur.

Le second problème est rencontré beaucoup plus fréquemment. Il apparaît lorsque le moteur génératif prend trop d'avance. En effet, sans mécanisme de régulation, l'accompagnement automatique est capable de produire plusieurs minutes d'accompagnement dès les premières secondes de jeu. Aussi, n'est-il plus du tout réactif par rapport aux changements que lui suggère, même inconsciemment, l'utilisateur. Le générateur est donc muni d'un paramètre permettant de fixer l'avance maximum autorisée. Ce point est illustré par la figure 2. Si la génération automatique dépasse cette avance, le moteur est mis en sommeil pour un moment, afin de l'empêcher de trop pré-calculer.

Dans la pratique que nous avons mise en place, l'accompagnement automatique proposé joue une basse. Elle est donc monophonique : il n'y a qu'une seule fourmi musicienne. Les notes jouées par l'accompagnement sont transposées deux octaves vers le bas. L'instrument de musique virtuel a fait l'objet d'une présentation lors de la conférence Generative Art en 2008 [10].

4 Évaluation et résultats

4.1 Étude des paramètres

L'idée originale d'utiliser le déplacement des fourmis pour créer des mélodies vient de l'analogie entre la structure répétitive que l'on peut donner à une mélodie, ce qui la rend identifiable par l'oreille humaine, et la construction de chemins par les fourmis. Dans la perspective de l'étude des systèmes auto-organisés, les chemins construits par les fourmis représentent des structures possédant une certaine stabilité dans le temps tout en étant capable d'évolution, ou de bifurcation. Ce sont également ces caractéristiques d'évolution qui permettent d'envisager une évolution pertinente des mélodies.

Le comportement général de l'algorithme se ressent par la création de chemins cycliques stables dans le graphe. Il se traduit par des boucles musicales qui se créent naturellement au cours du déroulement de l'accompagnement automatique. Ce résultat est évident à l'écoute et diffère nettement d'un algorithme purement aléatoire. Il est d'ailleurs possible, pour s'en convaincre d'utiliser l'algorithme de fourmis pour obtenir un résultat totalement aléatoire, en fixant les dépôts phéromonaux ainsi que l'évaporation à zéro.

Le rapport entre la quantité de phéromones τ_f déposée par les fourmis et la quantité fixée à l'initialisation du graphe τ_i permet de faire varier l'influence de l'intelligence collective sur le comportement global. Plus ce rapport τ_f/τ_i est petit, plus l'algorithme aura un comportement aléatoire. Cette organisation émergente est la raison principale de l'utilisation d'un algorithme de ce type dans le cadre de ce projet. La rapidité d'apparition de ces cycles et leur stabilité dans le temps dépendent des paramètres de l'algorithme.

Procédons maintenant à une évaluation de l'influence de ces paramètres sur le comportement du moteur génératif. Dans cette analyse, seules les hauteurs des notes sont prises en compte. Il est en effet beaucoup plus facile de reproduire une suite de notes de même hauteur, plutôt qu'une même figure rythmique. Les effets observés sur le graphe de hauteur peuvent s'extrapoler sur celui des durées.

4.1.1 Le nombre de fourmis

L'impact du nombre de fourmis sur le système est évalué avec les paramètres repris au tableau 4. Pour annuler l'influence de l'utilisateur, la quantité de phéromones qu'il dépose est à zéro. L'expérience consiste à jouer, à chaque fois, une même suite de notes, en l'occurrence la séquence fournie par la figure 3. L'utilisation d'une séquence prédéfinie permet de construire un graphe donné. On mesure la fréquence de chaque note dans la mélodie obtenue. Les résultats du tableau 5 sont obtenus sur plusieurs expériences ayant permis la génération de quelques milliers de notes.

On observe qu'avec 100 fourmis ou plus, une boucle entre si♭ et do est immédiatement créée et ne varie jamais. En effet, plus les fourmis sont nombreuses, plus le comportement résultant est rapidement cyclique et stable. Ce résultat est conforme à l'idée que l'organisation émergente est dépendante du nombre

FIGURE 3 – Suite expérimentale et graphe résultant

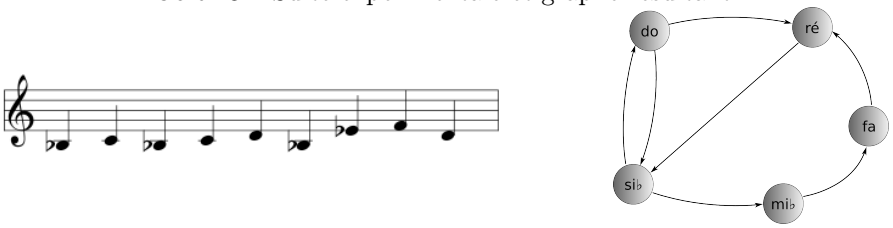


TABLE 4 – Valeurs des paramètres utilisés pour l'étude de l'influence du nombre de fourmis

Paramètre	Valeur
N_m	1
N_s	1 - 9 - 99 - 999
τ_f	1
τ_u	0
ρ	0,1

TABLE 5 – Influence du nombre de fourmis : proportion de chaque note dans la mélodie générée

$N_m + N_s$	2	10	100	1 000
si♭	48,81 %	45,92 %	48,61 %	50 %
do	35,71 %	45,92 %	51,39 %	50 %
ré	4,76 %	4,08 %	0 %	0 %
mi♭	9,52 %	1,02 %	0 %	0 %
fa	1,19 %	3,06 %	0 %	0 %

TABLE 6 – Valeurs des paramètres utilisés pour l'étude de l'influence du coefficient d'évaporation ρ

Paramètre	Valeur
N_m	1
N_s	9
τ_f	1
τ_u	0
ρ	0 - 0,25 - 0,5 - 0,75 - 0,9

TABLE 7 – Influence du coefficient d'évaporation ρ : proportion de chaque note dans la mélodie générée

ρ	0	0,25	0,5	0,75	0,9
si♭	50 %	46,81 %	48,10 %	36,36 %	40,54 %
do	50 %	48,94 %	46,84 %	28,57 %	33,78 %
ré	0 %	4,26 %	2,53 %	15,58 %	18,92 %
mi♭	0 %	0 %	2,53 %	7,79 %	0 %
fa	0 %	0 %	0 %	11,69 %	6,76 %

de fourmis. Dans le cas de notre instrument, il n'est pas recommandé d'utiliser un trop grand nombre de fourmis, pour ne pas dégrader les performances, mais aussi pour ne pas avoir systématiquement un résultat trop prévisible. Dans la pratique, ce nombre est fixé à dix, pour permettre le mécanisme d'intelligence collective sans surcharger la machine et en conservant une part d'imprévu.

4.1.2 Le coefficient d'évaporation

L'influence de l'évaporation des phéromones sur le comportement global de la colonie est évaluée à l'aide des paramètres du tableau 6. Là aussi, l'influence phéromonale de l'utilisateur est nulle, et les conditions de l'expérience sont identiques à l'expérience précédente. Les mesures du tableau 7 correspondent là encore aux fréquences d'apparition de chaque note dans la mélodie générée.

L'évaporation des phéromones a un impact important sur l'organisation produite par la colonie. Plus la valeur du coefficient d'évaporation est élevée, plus les résultats obtenus sont dispersés. L'organisation globale est plus stable dans le temps avec une évaporation faible. De la même manière que pour le nombre de fourmis, pour permettre d'obtenir cette création de boucle mélodique, sans pour autant la rendre trop immuable, le coefficient d'évaporation utilisé pour le logiciel est $\rho = 0{,}25$.

TABLE 8 – Valeurs des paramètres utilisés pour l'étude de l'influence de τ_u, interaction moteur-utilisateur

Paramètre	Valeur
N_m	1
N_s	9
τ_f	1
τ_u	1 - 5 - 10 - 20
ρ	0,25

TABLE 9 – Influence de τ_u, interaction moteur-utilisateur : proportion d'expériences avec une influence notable de l'utilistateur

τ_u	1	5	10	20
Influence	40 %	60 %	80 %	100 %
Pas d'influence notable	60 %	40 %	20 %	0 %

4.2 Efficacité des interactions

L'impact du jeu de l'utilisateur sur le moteur génératif est évalué par l'expérience suivante : une première séquence de notes est jouée, suivie d'une pause de quelques secondes permettant à l'accompagnement de se stabiliser. Cette séquence très simple, si♭ – do – si♭, induit une boucle dans la mélodie générée. Une seconde séquence, si♭ – ré – si♭ – ré, vient tenter de perturber le cycle. Cette expérience est répétée de nombreuses fois, en utilisant différentes valeurs de τ_u. Les paramètres utilisés sont récapitulés dans le tableau 8. Les résultats obtenus, repris sur le tableau 9, mesurent le pourcentage d'expériences au cours desquelles l'accompagnement est sorti de sa boucle initiale.

Ainsi, permettre à l'utilisateur de déposer lui aussi des phéromones sur le graphe a un impact majeur sur le comportement de l'algorithme. En fait, plus que la quantité déposée, τ_u, en elle-même, c'est le rapport entre celle-ci et celle déposée par les fourmis, $\frac{\tau_u}{N \times \tau_f}$, qui caractérise l'influence de l'utilisateur. Des expériences avec des ratios similaires pour des quantités différentes produisent, d'ailleurs, les mêmes résultats.

4.3 Évaluation de terrain

L'outil a pour objectif de jouer de la musique. Une évaluation objective des résultats obtenus est, au moins périlleuse, pour ne pas dire impossible. L'évaluation de la pertinence de l'outil passe donc par des enquêtes subjectives auprès de véritables utilisateurs.

Le public cible dans le cadre de cette évaluation peut être regroupé en plusieurs grandes catégories :

– personnes handicapés ;
– thérapeutes ;
– enfants ;
– musiciens ;
– grand public.

Les rencontres avec ces utilisateurs tests sont de deux types. Les premières sont des entretiens, individuels, d'une durée d'une à deux heures suivant les cas. Ces entretiens concernent des personnes avec qui un rendez-vous a été fixé auparavant, et qui ont donc déjà consenti à procéder à un test de l'outil. Ils débutent par une présentation générale de l'instrument de musique virtuel comportant :

– une présentation de l'équipe de recherche ;
– le contexte dans lequel le programme a été conçu ;
– ses objectifs généraux ;
– les principes de base du fonctionnement de l'outil.

À la suite de cette présentation, une première proposition est faite de procéder à un essai. Dans certains cas, une demande de renseignements complémentaires amène à aborder le fonctionnement interne de l'accompagnement automatique, c'est-à-dire l'algorithme de colonie de fourmis, ainsi que le fonctionnement des deux interfaces de contrôle.

Le second type de rencontre s'effectue lors de manifestations ouvertes au public et prend la forme de présentations sur un stand. Le temps disponible étant moindre, la présentation et les discussions qui s'ensuivent sont donc beaucoup plus succinctes. Le programme a ainsi été présenté lors de la Fête de la science 2009, des Journées portes ouvertes 2009 de l'université François Rabelais de Tours, du festival Univercités 2010 organisé par l'association les Petits débrouillards de Région Centre. Tourné vers un public plus spécifique, le Challenge Handicap et Technologie [1] qui a eu lieu à l'université de Metz les 15 et 16 mai 2009, a aussi servi de terrain d'expérimentation. On a d'ailleurs pu y découvrir que, même s'il n'était pas prévu à cet effet, le programme était tout à fait utilisable par une personne non voyante, principalement *via* l'utilisation du clavier, la souris étant plus délicate à utiliser en l'absence du retour visuel de sa position sur l'écran. Le challenge regroupait 24 projets, l'instrument de musique virtuel accessible y a remporté le premier prix « Défi Loisirs ».

Les conférences scientifiques ayant permis des communications sur le projet ont souvent été, elles aussi, des lieux de tests. La rencontre avec une personne handicapée d'un bras a permis de constater la possibilité d'utiliser l'outil à l'aide d'une seule main. Il s'agit d'ailleurs d'un point d'intérêt fort du logiciel car les instruments de musiques habituels utilisables avec une seule main sont très rares, aussi les personnes ne pouvant utiliser pleinement leurs deux mains y trouvent une façon aisée de pratiquer la musique.

Les essais du logiciel se font systématiquement à l'aide d'un casque audio. En effet, le test ne se fait jamais tout seul, puisqu'il se trouve toujours, au moins, l'expérimentateur à côté de l'utilisateur. L'idée est d'évaluer si la musique

1. `http://challenge-ht.univ-metz.fr/`

Table 10 – Résultats des tests de terrain

	Grand public	Enfants	Musiciens	Thérapeutes	Personnes handicapées	Total
Entretiens	0	10	10	6	3	29
Stand	100	150	20	2	10	282
Apparemment satisfaits	65 %	89,4 %	46,6 %	100 %	76,9 %	77,2 %

jouée par l'utilisateur lui plaît. Il n'est donc pas souhaitable qu'interviennent des rapports sociaux implicites. Les interrogations d'un utilisateur, sur le fait que ce qu'il joue plaît aux autres auditeurs en plus de lui-même, risquent d'interférer avec le rapport personnel de ce testeur avec l'outil. C'est d'ailleurs pour cette même raison que nous avons conservé une version du programme qui affiche uniquement le cadre dans lequel la souris bouge, afin que les résultats graphiques des actions de l'utilisateur n'empiètent pas sur son expérience sonore.

L'observation de testeurs au cours de la phase d'essai constitue une importante partie de l'évaluation de l'outil. Poursuit-il son essai dans le temps ? Quelle attitude a-t-il ? Quelles sont les expressions de son visage ? Ce sont des questions de ce genre qui nous intéressent à ce moment. Cette phase est, en général, suivie d'une phase de questions informelles, si possible, en invitant l'expérimentateur à faire des retours sur l'expérience, sa satisfaction globale, les points qui lui semblent à améliorer, et les usages possibles du programme.

Ces observations subjectives et les échanges qui suivent nous permettent de distinguer trois catégories d'utilisateurs. Les premiers sont ceux qui ne sont pas immédiatement intéressés et qui n'ont pas la curiosité de pousser l'expérience plus loin. Leur essai est très court, de l'ordre de quelques secondes, et ils ne sont manifestement pas convaincus par l'outil. Parmi les raisons de cet échec, on trouve sans doute le fait que le résultat qu'ils obtiennent ne correspond pas à l'idée qu'ils se sont faite *a priori*, ces personnes ayant souvent voulu savoir de quoi il s'agissait précisément avant d'essayer. La seconde partie des testeurs, environ deux tiers d'entre eux, est celle des personnes qui, intéressées par l'outil, poussent leur expérience plus longuement. En général, ils ne cèdent leur place que parce que d'autres utilisateurs attendent, et parfois même, demandent à récupérer le programme. Lors des rencontres avec un jeune public, la proportion de ces utilisateurs intéressés devient très majoritaire, de l'ordre de 90 %. En effet, les enfants sont plus prompts à expérimenter et trouvent immédiatement un côté ludique dans l'utilisation du logiciel. La troisième partie des utilisateurs englobe ceux qui restent indécis. Nous considérons qu'ils ne sont pas pleinement satisfaits par l'outil et les assimilons à la catégorie des personnes déçues. Les résultats obtenus sont regroupés dans le tableau 10.

Ces chiffres sont complétés par quelques mots sur certains des entretiens que

nous avons menés. Nous sommes ainsi allés à la rencontre de thérapeutes pour recueillir leurs sentiments sur ce projet. Dans l'ensemble très enthousiastes, ces personnes nous ont accordé souvent plusieurs heures de leur temps. Le médecin chef du centre de ressources sur l'autisme de Région Centre, par ailleurs musicien, voit dans une utilisation tranquille de l'outil un côté relaxant. Sans lui avoir précisément décrit le comportement du moteur génératif, il a naturellement eu l'impression d'être « suivi par une ombre musicale ».

Un musicothérapeute de Touraine avec qui nous avons longuement discuté s'est montré particulièrement intéressé. Dans sa pratique, il organise des séances de musique collective avec des patients en situation de handicap mental. Certains de ces patients, qui présentent des troubles de l'apprentissage, ont de grandes difficultés à participer à ces séances. Seuls des instruments très simples, type percussions, sont à leur portée et ils ont rapidement tendance à faire plus de bruits désordonnés que de réelle musique, ce qui nuit à la communication de l'ensemble du groupe. Les contraintes intégrées au logiciel permettent à ce type d'utilisateurs, même lorsqu'ils font des mouvements désordonnés, de conserver un résultat « musical », et donc de rester intégrés au groupe.

Le 9 février 2010, une présentation à l'équipe d'encadrement du centre d'accueil spécialisé de Ballan Miré (37) a permis de déployer le logiciel sur leur parc d'ordinateurs et le mettre ainsi à la disposition de leur vingtaine de résidents. Parmi les constats de cette rencontre, le manque d'information du personnel sur les moyens d'adaptation au handicap d'ordinateurs standard est flagrant. Néanmoins, la connaissance du public handicapé de l'équipe a permis de proposer des améliorations du programme.

Les résultats de cette évaluation ont été présentés lors du congrès Handicap 2010 [9]. D'une manière générale, les conclusions en sont les suivantes. Les objectifs de ces travaux récoltent l'adhésion quasi générale. Le logiciel en lui-même, sans être complètement consensuel, suscite majoritairement l'intérêt de ceux qui l'essaient. La principale raison d'utiliser le programme semble tenir à son côté ludique et interactif. La curiosité et l'étonnement laissent rapidement place à l'amusement sur le visage des gens qui tentent l'expérience. D'autres usages émergent de cette campagne de test. L'utilisation de l'outil paraît opportune dans le cadre de séance de relaxation. Un maniement lent de l'instrument tend à produire des sons calmes et reposants. Il s'adapte, aussi, aux personnes ayant des troubles de l'apprentissage, qui peuvent ainsi accéder à une création musicale particulièrement chaotique par ailleurs. Ces rencontres ont aussi permis de dégager un ensemble d'améliorations ou d'évolutions possibles du logiciel. Parmi ces améliorations, certaines sont déjà intégrées, telles que le démarrage automatique de l'accompagnement lorsque l'utilisateur se met à jouer, ou la possibilité d'utiliser simultanément le clavier et la souris.

5 Conclusion

Ces travaux permettent de valider qu'il est possible de créer des outils accessibles de création artistique. L'incorporation de systèmes interactifs générateurs

d'art à ces outils est pertinente dans le cadre de l'accessibilité. Les résultats des évaluations montrent d'ailleurs un vif intérêt du public pour ce genre d'outil. Le choix d'un algorithme de colonie de fourmis artificielles dans ce cadre s'est montré judicieux. Nous avions besoin d'un algorithme à réponse instantanée. La rapidité d'exécution de ces algorithmes permet d'y répondre. Il nous fallait aussi un système capable de s'adapter à des changements des données qu'il manipule. La robustesse des algorithmes de fourmis s'est, là aussi, montrée appropriée.

Des améliorations restent encore à développer. Parmi les évolutions possibles, la plus évidente est l'ajout d'un retour graphique aux actions de l'utilisateur. Il peut prendre de nombreuses formes (affichages des graphes et des fourmis, tracé des mouvements de la souris, affichage de la partition, etc.), ayant chacune des objectifs et un intérêt différents.

Citons aussi des améliorations des capacités musicales de l'outil. À l'heure actuelle, l'instrument utilise toujours le même timbre. Il joue un son de violon car c'est un son qui peut durer aussi longtemps que nécessaire. La possibilité de choisir le *patch* midi peut être ajouté pour permettre à l'utilisateur de choisir le timbre de son instrument. Dans l'éventualité d'augmenter ou de diversifier les possibilités musicales, il est envisageable d'offrir plusieurs types de gammes (mineure, pentatonique, chromatique, etc.) et un moyen de choisir parmi elles. De la même façon, le clavier peut proposer des touches pour produire directement des accords. Ce genre d'évolutions vient rapidement complexifier l'interface et nuire à la simplicité d'utilisation, aussi ouvrent-elles la porte à une interface adaptative qui se personnalise automatiquement en fonction des besoins de l'utilisateur.

En se basant sur une conception modulaire de l'interface, regroupant des fonctions connexes au sein d'éléments d'interfaces pouvant être affichés ou masqués, une interface qui s'enrichit progressivement lorsque l'utilisateur s'habitue au logiciel constitue un exemple d'interface adaptative. L'utilisateur débute alors avec une interface minimale, qui offre de plus en plus d'options avec le temps. L'avis de l'utilisateur sur l'interface peut être collecté, en récupérant la façon dont il dispose les différents éléments, et en dotant ces éléments d'un moyen pour l'utilisateur de les valider ou de les désapprouver. Un algorithme peut alors choisir, à partir de ces informations, quels autres éléments proposer dans l'interface.

D'autres évolutions sont envisageables au niveau du moteur du logiciel. Une détection automatique du tempo permet une adaptation plus fine au jeu de l'utilisateur. L'accompagnement automatique peut être enrichi, en lui ajoutant d'autres instruments, ou en testant d'autres méthodes génératives. L'algorithme de fourmis artificielles déjà développé peut incorporer plus de règles musicales, par exemple par l'utilisation de seuils. Cette forme d'hybridation avec les mécanismes de partage de tâches chez les fourmis permettrait d'inclure un mécanisme de préférence indépendant des phéromones, et pouvant favoriser certains intervalles ou certaines durées, comme par exemple une tendance à compléter exactement une mesure.

Ce ne sont là que quelques pistes de développement qui ouvrent la voie à plus d'adaptations et de personnalisation d'un outil déjà prometteur.

Le développement d'autres outils, s'appliquant à d'autres champs artistiques

est aussi un point intéressant. Nous travaillons sur un atelier de dessin accessible basé sur les mêmes idées et qui a été présenté à la conférence Generative Art de 2009 [7].

Références

[1] Gérard Assayag, Georges Bloch, and Marc Chemillier. OMAX-OFON. In *Sound and Music Computing (SMC) 2006*, Marseille, France, 2006.

[2] Sebastien Aupetit, Vincent Bordeau, Nicolas Monmarché, Mohammed Slimane, and Gilles Venturini. Interactive evolution of ant paintings. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 1376–1383, Canberra, 8-12 december 2003.

[3] Tim Berners-Lee. Définition de l'accessibilité du web. Site web d'accessiweb : `http://www.accessiweb.org/fr/Label_Accessibilite/`, juin 1997.

[4] John Al Biles. Genjam : A genetic algorithm for generating jazz solos. In *International Computer Music Conference (ICMC'94)*. The Computer Music Association, 1994.

[5] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence : From Natural to Artificial Systems.* Santa Fe Institute Studies on the Sciences of Complexity. Oxford University Press, October, 21 1999.

[6] Giordano Cabral, Jean-Pierre Briot, and François Pachet. Incremental parsing for real-time accompaniment systems. In *The 19th International FLAIRS Conference, Special Track : Artificial Intelligence in Music and Art*, Melbourne Beach, USA, 2006.

[7] Romain Clair, Nicolas Monmarché, and Mohamed Slimane. Accessible art creation tools, a generative arts application. In Celestino Soddu, editor, *12th Generative Art Conference (GA2009)*, page 9, Milano, Italy, 15–17 decembre 2009.

[8] Romain Clair, Nicolas Monmarché, and Mohamed Slimane. An interactive ant algorithm for real-time music accompaniment. In *Évolution Artificielle, 9th International Conference on Artificial Evolution*, page 12, Strasbourg, France, 26–28 octobre 2009.

[9] Romain Clair, Nicolas Monmarché, and Mohamed Slimane. Accessibilité de la création musicale par ordinateur : l'exemple d'un instrument virtuel avec accompagnement automatique. In *Handicap 2010*, Paris, juin 2010.

[10] Romain Clair, Nicolas Monmarché, and Mohamed Slimane. Interactions between an artificial colony of musical ants and an impaired human composer : towards accessible generative arts. In Celestino Soddu, editor, *XI Generative Art conference (GA2008)*, page 10, Milano, Italy, 16–18 decembre 2008.

[11] Marco Dorigo and Gianni Di Caro. Ant colony optimization : A new metaheuristic. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99)*, volume 2, pages 1470–1477. IEEE Press, 6–9 Juillet 1999.

[12] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. The ant system : Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B : Cybernetics*, 26(1) :29–41, 1996.

[13] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. Bradford Books. MIT Press, Cambridge, USA, July 2004.

[14] Christophe Guéret, Nicolas Monmarché, and Mohamed Slimane. Ants can play music. In *Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2004)*, volume 3172 of *Lecture Notes in Computer Science*, pages 310–317, Université Libre de Bruxelles, Belgique, 5-8 September 2004. Springer-Verlag.

[15] Lejaren Hiller and Leonard Isaacson. *Experimental music : Composition with an electronic computer*. McGraw-Hill, New York, 1959. réimprimé en 1979 par Greenwood Press.

[16] Andreï Andreïevitch Markov. An example of statistical investigation in the text of eugene onegin illustrating coupling of tests in chains. In *Proceedings of Academic Scientific St. Petersburg*, volume VI, pages 153–162, Saint Petersburg, Russie, 1913.

[17] Jon McCormack. A method for generating phyliotaxis over surfaces of revolution. Technical Report 2003/149, Monash University, Melbourne, December 2003.

[18] Eduardo Reck Miranda and John Al Biles. *Evolutionary Computer Music*. Springer, 2007.

[19] Nicolas Monmarché, Frédéric Guinand, and Patrick Siarry, editors. *Fourmis Artificielles, des bases de l'optimisation aux applications industrielles*, volume 1 of *Traité IC2*. Hermès-Lavoisier, 2009.

[20] Nicolas Monmarché, Frédéric Guinand, and Patrick Siarry, editors. *Fourmis Artificielles, nouvelles directions pour une intelligence collective*, volume 2 of *Traité IC2*. Hermès-Lavoisier, 2009.

[21] Nicolas Monmarché, Isabelle Mahnich, and Mohamed Slimane. Artificial Art made by Artificial Ants. In Juan Romero and Penousal Machado, editors, *The Art of Artificial Evolution : A Handbook on Evolutionary Art and Music*, Natural Computing Series, pages 227–247. Springer Berlin Heidelberg, 2007.

[22] Nicolas Monmarché, Mohamed Slimane, and Gilles Venturini. Fourmis artistiques ou l'art artificiel pictural et musical. In *Hypertextes et Hypermédias, Réalisations, Outils & Méthodes (H2PTM'03)*, pages 109–118, Paris, France, septembre 2003. Hermes.

[23] Nicolas Monmarché and Romain Clair. Des fourmis artificielles pour l'art artificiel. In Nicolas Monmarché, Frédéric Guinand, and Patrick Siarry, editors, *Fourmis Artificielles, des bases algorithmiques aux concepts et réalisations avancés*, Traité IC2. Hermès-Lavoisier, 2009.

[24] Gerhard Nierhaus. *Algorithmic composition*. SpringerWienNewYork, Wien, 2009.

[25] François Pachet. Interactions réflexives. In Yann Orlarey, editor, *Actes des rencontres musicales pluridiscipinaires*, Lyon, France, 2006.

[26] Przemyslaw Prusinkiewicz and Aristid Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

[27] Iannis Xenakis. *Musiques formelles*. Stock, 1981.