

EXEMPLE D'UTILISATION D'I-SCORE POUR L'ECRITURE D'UNE PIECE MUSICALE INTERACTIVE

Pierre Cochard

Scrim / LaBRI
pcochard@labri.fr

Théo de la Hogue

GMEA
theo.delahogue@gmea.net

1. BOIDS

La pièce musicale interactive présentée dans cet article consiste en la sonorisation d'un programme de simulation de nuée d'oiseau inventé par Craig Reynold (« Boids »¹). Chaque *boïd* se déplace au regard de son voisinage selon des règles d'évitement, d'alignement et de cohésion. En se basant sur une implémentation fournie dans les exemples du logiciel Processing², nous avons ajouté la possibilité de fixer une destination cible que la nuée cherche à atteindre (contrôlable via la souris), des analyses statistiques (déviations, position et orientation moyennes) et une détection de collision. Les modalités d'interaction avec la nuée ont été réduites au minimum : un écran tactile permet d'utiliser un seul doigt comme pointeur de souris.

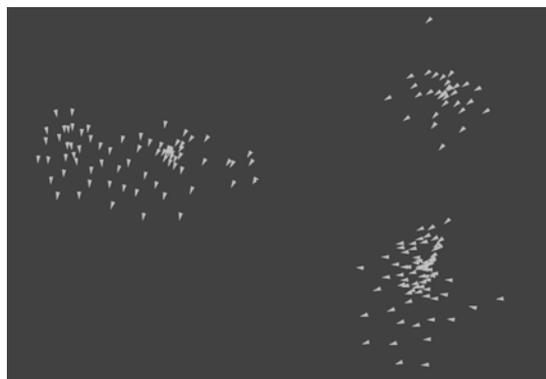


Figure 1 Capture du programme de simulation de nuée réalisé dans *Processing*.

Parallèlement au développement graphique, un moteur audio a été développé dans *SuperCollider* pour sonoriser le vol de la nuée. Celui-ci se base sur différents modules de synthèse (générateurs de bruit, synthèse granulaire, synthèse par modulation de fréquence et synthèse par tables d'onde) dépendants de paramètres de déclenchement (sur le modèle des notes MIDI) obéissant eux-mêmes à certaines contraintes (gammes tonales, ambitus limités) ainsi que de paramètres de modulation (volume, panoramique stéréo, taille et densité de granulation).

¹ <https://fr.wikipedia.org/wiki/Boids>

² <https://processing.org/examples/flocking.html>

2. METHODOLOGIE

2.1 Déclaration des paramètres d'écriture

L'utilisation d'*i-score* pour programmer les interactions entre le moteur graphique et le moteur audio a nécessité une méthodologie particulière. Tout d'abord, chaque moteur de média a dû s'assurer d'être interopérable avec *i-score* via le protocole OSC. Ensuite il a fallu spécifier dans *i-score* les types et domaines des valeurs OSC reçues ou émises par chaque logiciel.

Address	Value	Get	Set	Min	Max
Processing					
sky					
color	[18, 167, 224]	✓	✓		
text					
mouse					
click	false	✓		false	false
flock					
kill	true	✓	✓	false	false
info	false	✓	✓	false	false
collision		✓			
killed		✓			
direction					
y	0	✓		-1	1
x	0	✓		-1	1
position					
y	0,870627	✓		0	1
x	0,265794	✓		0	1
deviation					
y	0	✓		0	1
x	0	✓		0	1
size	0	✓		0	1000
follow					
rate	1	✓	✓	0	5
distance					
max	50	✓	✓	0	100
min	25	✓	✓	0	100
cohesion	1	✓	✓	0	2
alignment	1	✓	✓	0	2
avoidance	1,2	✓	✓	0	2
add		✓			
clear		✓			

Figure 2 Namespace du programme de simulation de nuée réalisé dans *Processing*.

Address	Value	Get	Set	Min	Max
Processing					
SuperCollider					
falconInterface_1					
bassfm_gain	0	✓	✓	0	2
bassfm_note	0	✓	✓	34	46
boïd_density	0	✓	✓	0,1	10
boïd_gain	0	✓	✓	0	2
boïd_grainsize	0	✓	✓	5	500
boïd_note	0	✓	✓	0	127
boïd_pan	0	✓	✓	-0,99	1
create_boïd	0	✓	✓	0	1
granpad_gain	0	✓	✓	0	2
granpad_note	0	✓	✓	0	127
gransects_gain	0	✓	✓	0	2
gransects_note	0	✓	✓	69	127
level	0	✓	✓	-96	12
mix	0	✓	✓	0	1
noise_gain	0	✓	✓	0	2
noise_note	0	✓	✓	95	127
reset_score	0	✓	✓	0	1

Figure 3 Namespace du moteur audio réalisé dans *SuperCollider*

Il faut noter que cette étape laborieuse dans le cas de *Processing* a pu être évitée dans le cas de *Super-*

Collider par l'utilisation du protocole *Minuit*³. Ce système de requêtes/réponses via OSC a permis à *i-score* de questionner le logiciel sur les caractéristiques des messages OSC qu'il reçoit ou émet.

2.2 Ecriture logico-temporelle

La pièce se compose de trois phases distinctes : une période d'ajout d'individu à chaque touché de l'écran tactile (fig. 4), un moment d'interaction avec la nuée pour guider son vol en déplaçant le doigt (fig. 5) suivi d'une période où les individus disparaissent un à un à mesure qu'il sorte de l'écran (fig. 6, aucune d'interaction requise).

Après avoir initialisé certains paramètres au démarrage de la partition, la première phase consiste en une période de temps indéterminée où le dispositif attend que 300 individus soit créés. Durant cette période, à chaque touché de l'écran, une boucle d'au moins 1 seconde est déclenchée pendant laquelle la commande */flock/add* est envoyée toutes les 50 ms à *Processing*. Une fois le touché terminé, le dispositif attends un autre touché. Parallèlement à ce comportement, la taille de la nuée renvoyée par *Processing* est mise en relation avec le paramètre *boid_density* (la densité – c'est-à-dire le nombre de grains - du module de synthèse granulaire) du moteur audio grâce à une fonction de transfert valable uniquement sur cette première période. Par ailleurs, une deuxième fonction de transfert entre ces deux programmes a été implémentée permettant le suivi panoramique (stéréo) continu de la nuée, avec le paramètre *boid_pan*.

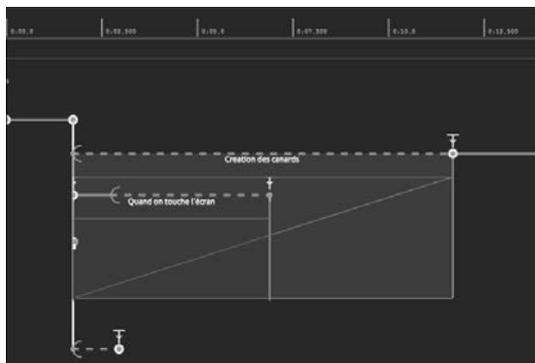


Figure 4 Initialisation puis période d'ajout

La deuxième phase se déroule après quelques secondes d'attente sans interaction possible. Ensuite, dès que le système détecte un touché, la nuée est configurée pour suivre le touché et les paramètres d'évitement, d'alignement et de cohésion sont modifiés progressivement sur 1 minutes. Pendant ce temps, toutes les 3 secondes, la partition vérifie l'orientation moyenne de la nuée et, selon quatre cas possibles, envoie une commande différente au moteur audio.

Après une minute, dès que le système détecte un relâchement du touché, la deuxième phase se termine.

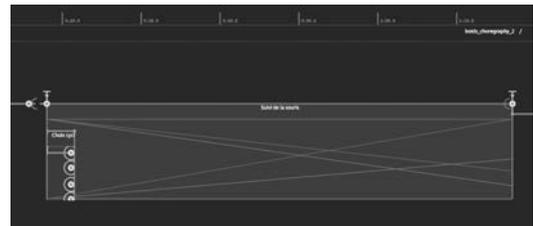


Figure 5 Suivi du doigt par la nuée

La troisième phase n'attend aucune interaction pour commencer. Après avoir reconfigurer les paramètres de la nuée, elle observe l'émission de message relatif à la disparition d'un individu pour déclencher une commande à destination du moteur audio. De même que lors de la première phase, la taille de la nuée renvoyée est mise en relation avec le paramètre *boid_density* du moteur audio.

Cette dernière phase se termine soit lorsque tous les individus de la nuée ont disparu ou au bout d'1 minute et 30 secondes quoi qu'il arrive.

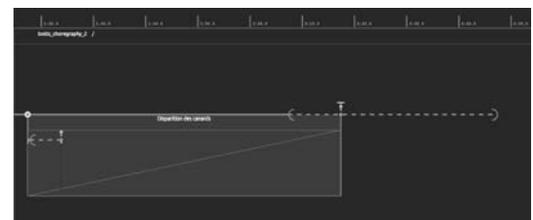


Figure 6 Disparition

3. CONCLUSION

Abstraction faites de tous les problèmes inhérents à un logiciel en phase alpha, l'utilisation d'*i-score* pour l'écriture de l'interaction entre deux médias (ici de synthèse graphique et sonore) ouvre de nouveaux horizons. En proposant un nouveau langage graphique pour formaliser le comportement logico-temporel d'un dispositif interactif, *i-score* apporte des solutions simples à des situations d'interaction habituellement complexes à programmer dans d'autres environnements. Si nous avons programmé ce scénario dans *Processing* et/ou dans *SuperCollider* plusieurs situations d'interactions auraient posé de multiples problèmes d'ingénierie logicielle. Au delà du gain de temps, *i-score* nous a permis de nous concentrer sur des considérations d'ordre esthétique.

Cependant toutes les situations d'écriture n'ont pas pu être écrites dans *i-score*. Certaines étapes plus algorithmiques ont nécessité l'ajout de paramètres dédiés au déclenchement de processus internes à *SuperCollider*. Cependant l'existence dans *i-score* de plugin *JavaScript* laisse espérer de pouvoir écrire ces situations dans les prochaines versions du logiciel. De même les situations d'interaction impliquant des variables ou relations spatiales nous ont fait défaut mais là encore les recherches et développements à venir promettent des services allant dans ce sens.

³ <https://github.com/Minuit/minuit>