Numéros / n° 2 - automne 2012

« La transduction dans la création de la musique visuelle en temps différé »

Carlos López Charles

Résumé

Le coût de traitement computationnel nécessaire pour transférer des données entre le son, le geste et l'image numériques pose des contraintes considérables sur la quantité, la qualité et les processus de manipulation qu'un artiste peut réaliser sur ces matériaux en temps réel. Cet article introduit des techniques et outils logiciels pour établir des transferts de données entre divers espaces de représentation (comme des signaux audio ou vidéo, des bases des données ou des algorithmes génératifs, par exemple) en temps différé. Trois exemples de l'usage des idées présentées sont décrits pour analyser les avantages que cette approche pour la création audiovisuelle.

1. Introduction

Depuis le xviii^{e} siècle, le terme « musique visuelle » a été utilisé par des artistes et des inventeurs ? comme François Bernard Castel, Thomas Wilfred, Oskar Fischinger, Norman McLaren et Golan Levin ? de très diverses façons. Pour les uns, il fait référence aux oeuvres produites par l'utilisation de moyens qui permettent de combiner le son et l'image dans une unité holistique (Levin, 2000, p. 46) (1). Pour les autres il est une forme strictement silencieuse où les images produisent une « musique pour l'oeil » qui crée des effets comparables à ceux que le son produit sur l'écoute (Moritz, 1986) et, pour certains, il s'agit d'utiliser les moyens de l'image au service de l'expression musicale par la construction de correspondances entre le rythme, la texture, la couleur, etc. Quoi que ce soit, les discussions sur ce sujet restent toujours interdisciplinaires et se concentrent sur l'usage de métaphores et de technologies pour construire des correspondances entre le sonore et le visuel.

Dans cet article, j'examine le potentiel de l'application du concept de transduction dans la création de musique visuelle en temps différé. Bien que le transfert d'information entre divers types de données a ouvert une infinité de possibilités dans ce terrain, les possibilités de l'application de ce principe sous cette perspective ont été peu étudiées jusqu'à maintenant. Dans le but de démontrer l'utilité de cette approche, j'analyserai trois travaux faits en temps différé à partir de la transduction entre divers types de données. Le premier travail, « Étude élastique », est une pièce audiovisuelle où diverses caractéristiques d'une musique préenregistrée sont analysés et utilisés pour piloter un système de génération d'images dans un environnement 3D. Le deuxième, est une animation basée sur l'usage d'un système de particules contrôlées par l'action humaine sur une interface de contrôle physique à plusieurs sliders. Le dernier est une démonstration d'un système où les quantités de mouvement dans différentes régions de la vidéo créée dans le deuxième cas d'étude pilotent les matériaux musicaux d'une bande sonore. Les problèmes posés et les solutions implémentées pour développer ces travaux de création seront décrits, tout comme les outils et les techniques développés pour faciliter les transferts de données entre les médias utilisés.

2. Les moyens de la transduction

Une des caractéristiques de la technologie numérique est qu'elle offre la possibilité d'établir des relations entre différents média. Ces transferts, ou plutôt ces *transductions*, sont des opérations très communes dans plusieurs systèmes qui permettent de créer des rapports audiovisuels très variés, comme on peut

l'apprécier dans la diversité de jeux vidéo musicaux, des instruments audiovisuels et des installations interactives, pour donner quelques exemples [(2)].

Le mot « transduction » vient du latin *transductio*, dont le sens était la transmission (*ducere*, « porter ») à travers (*trans*-) un milieu déterminé d'un objet qui, par le seul fait d'avoir été transmis, est aussi transformé comme conséquence d'entrer en contact ou interagir avec le milieu où il manifeste (Maestro, 2005, p. 2).

Figure 1

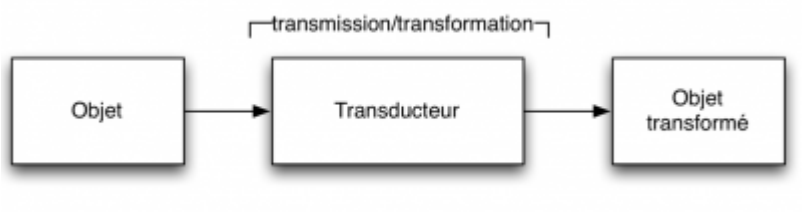


Diagramme d'un processus de transduction

En physique, ce mot est défini comme la conversion d'une quantité d'énergie d'une forme vers une autre forme différente (University of Oxford, 2005, p. 672). Un transducteur électroacoustique, par exemple, est un appareil ou dispositif qui convertit l'énergie électrique en énergie acoustique ou réciproquement. Les microphones à condensateurs comportent des transducteurs pour l'enregistrement sonore. Dans leur fonctionnement inverse, les transducteurs sont utilisés pour convertir l'énergie électrique en énergie acoustique dans les cas des écouteurs ou des haut-parleurs (Stöcker *et al.*, 1999, p. 328).

Sur le plan de l'expression musicale, l'usage des transducteurs nous permet de convertir tout type d'énergie (radiante, mécanique, thermique, électrique, magnétique et chimique) en sons ou images [(3)]. De cette manière, une énergie d'entrée comme, par exemple, celle des *gestes* (énergie mécanique) ou, plus précisément, des actions humaines porteuses d'information (Volpi *et al.*, 2007) [(4)], peut être transformée en énergie électrique et celle-ci, à son tour, peut être convertie en énergie magnétique pour contrôler les variations de pression d'un haut-parleur. Dans ce cas-là, la « trace » de l'interprétation devient un *contrôleur* de la génération du son parce qu'elle permet à l'utilisateur de réguler l'énergie qui sera transduite au dispositif de sortie (le haut-parleur) à chaque moment, au moyen d'un boucle action-perception. Bien entendu, les transductions entre l'énergie électrique et d'autres types d'énergie sont centrales dans la création numérique parce qu'elles permettent aux utilisateurs d'interagir avec diverses média par l'usage de divers types de transducteurs.

Par rapport aux technologies analogiques, les technologies numériques offrent beaucoup de flexibilité pour le contrôle des matériaux dans la création audiovisuelle. Au moyen des convertisseurs « analogiques-numériques » et « numériques-analogiques », ils permettent de représenter tout type de variations d'énergie comme une collection discrète (numérique) de symboles (échantillons) et, réciproquement, ils rendent possible la génération de signaux électriques à partir de représentations numériques, ce qui permet de définir, d'analyser, et de manipuler les caractéristiques de différents types de média (comme des sons, des images, des textes, etc.), non seulement d'après leur globalité perceptuelle (comme, par exemple, dans le cas de l'extraction de paramètres comme la brillance, le bruit ou l'intensité perceptive), mais aussi d'après leur code, au moyen de la transduction.

Sous leur forme numérique, les différents « média » avec lesquels on travaille ne sont, à la base, que des *espaces de représentation* (Vaggione, 2006, p. 14) où des collections de symboles définis de façon discrète permettent d'atteindre un certain niveau opératoire et morphologique sur les sons, les images et les différents éléments (des patches, des spectrogrammes, des algorithmes de contrôle, etc.) dont un artiste audiovisuel peut s'en servir pour manipuler ses différents matériaux. Alors, dans un contexte numérique, nous pouvons considérer le transfert d'information entre divers média comme une *transductionde données* entre différents modes de représentation [(5)]. Le recours à la transduction comme outil pour *médiation* est efficace dans un milieu numérique puisqu'il favorise la connexion des modes de représentation les plus divers. Du moment où les symboles d'un espace de représentation acquièrent la fonction de générer de nouvelles morphologies, ils prennent, comme le signale Vaggione, *un rôle*

similaire à celui de la notation musicale en tant que partition (Vaggione, 2006, p. 14). Ils deviennent les entités élémentaires d'un langage dans lequel plusieurs caractéristiques de l'objet représenté ? comme l'évolution de l'amplitude d'un son, les couleurs d'une image, le contenu d'une base de données, les données de sortie d'une opération mathématique et ainsi de suite ? peuvent être décrites, manipulées et articulées dans un réseau d'opérations compositionnelles.

Bref, le contrôle dans la création audiovisuelle par ordinateur peut être réalisé, au moins, de trois façons différentes : soit *par l'utilisation de transducteurs d'entrée* (ou capteurs) ? qui convertissent l'énergie d'un contrôleur analogique en énergie électrique dont les variations de grandeur peuvent être numérisées et utilisées comme matériau compositionnel ? ; soit en remplaçant les transducteurs analogiques d'entrée *par un contrôle numérique direct* de l'énergie envoyée aux transducteurs de sortie ? ce qui a l'avantage de permettre d'utiliser des techniques de contrôle plus complexes (Portillo et Polviani, 2007, p. 58) (6) ? ; soit *par une combinaison de ces techniques*.

3. La transduction dans la création audiovisuelle en temps différé

Quant à la mise en place des transductions de données dans la composition audiovisuelle en format fixe ? c'est-à-dire vidéo, cinéma, musique visuelle en temps différé ?, peu de développements technologiques ont encore été faits et le manque d'outils pour créer des travaux basés sur la transduction de données fait que la plupart des artistes qui veulent travailler de cette façon sont forcés à suivre leur démarche avec des outils pensés à l'origine pour la création en temps réel.

Les retombées de cette situation sont considérables. D'abord, le coût de calcul des processus dans ce type d'opérations en temps réel impose des contraintes d'une part, sur la complexité des transferts de données qu'un utilisateur peut prendre en main et, d'autre part, sur la résolution des matériaux avec lesquels il peut travailler. Dans le but de clarifier ce point, il me faudrait peut-être donner un exemple : si nous voulions établir un transfert de données entre le domaine sonore et le visuel pour faire une animation simple et l'enregistrer en temps réel, nous pourrions analyser l'amplitude d'une bande son et utiliser les données issues de cette analyse pour manipuler un nombre réduit de variables d'une animation et, de manière simultanée, enregistrer le rendu final sur une vidéo de basse résolution. Faire une chose comme ça ne pose pas des gros problèmes, car le coût de calcul d'un processus de ce type est relativement léger pour la moyenne des ordinateurs existants actuellement. Par ailleurs, si nous répétions ce procédé, mais cette fois avec une quantité très large de variables, à une résolution très haute et à une vitesse beaucoup plus rapide, le coût de traitement computationnel monterait beaucoup plus et la probabilité de réussir à obtenir une vidéo sans erreurs descendrait, ou même disparaîtrait, tout comme la *capacité transformationnelle* (7) (Vaggione, 1996) de ces données pour être mises en réseau afin de forger de nouvelles morphologies en temps réel. En d'autres mots, l'espace de possibilités pour créer des vidéos ou des sons en temps réel en utilisant la transduction comme principe générateur de forme devient de plus en plus restreint au fur et à mesure que la résolution des matériaux ou la quantité de transferts et de transformations de données dans un flux de travail monte.

Certes, travailler sous les contraintes imposées par la technologie n'a rien de nouveau (on compose toujours avec de possibilités définies par des contraintes techniques), mais il est bien évident que, pour la création audiovisuelle en format fixe, il serait possible de trouver des solutions logicielles qui rendent possible l'établissement de processus de transduction au-delà des limites de ce que les outils temps réel le permettent.

Alors, la question centrale serait celle-ci : quelle solution ou solutions logicielles pourraient nous permettre de créer des travaux audiovisuels en format fixe à partir de la transduction entre différents espaces de représentation ? Dans cet article je propose diverses façons de réaliser des transferts de données entre médias différents en temps différé. L'avantage principal du travail en temps différé pour la création audiovisuelle basée sur la transduction est qu'elle permet aux artistes d'élargir l'espace composable de leurs oeuvres par l'incorporation d'une quantité et une variété majeure de transferts de données entre les espaces de représentation et par l'utilisation de matériaux de meilleure qualité : pas seulement des sons et des images avec plus de résolution ou avec des taux d'échantillonnage plus élevés, mais aussi des bases de données plus larges et des algorithmes et des instructions de contrôle plus complexes que ceux que l'on peut utiliser en temps réel. Finalement, et bien que comme le signale Collins (Collins et Olofsson, 2006, p. 8-18), la synchronisation de différentes modalités risque de tomber dans le

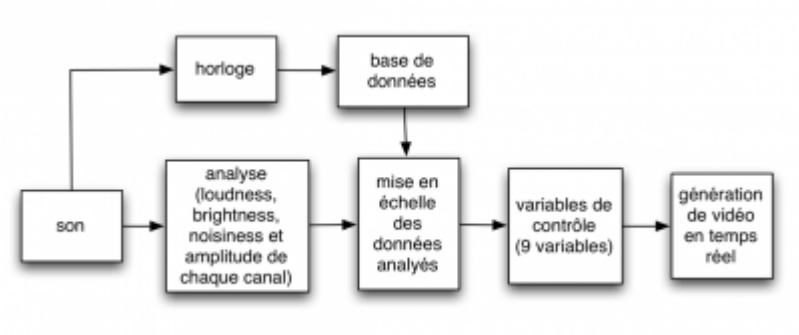
piège consistant à établir des relations audiovisuelles trop directes (« mickey mousing »), elle peut aussi devenir, d'une part, un point de départ pour l'établissement de relations plus subtiles, contrastantes et même combatives et, d'autre part, une préoccupation centrale pour la collaboration audiovisuelle.

4. Cas d'étude n° 1 : le son comme contrôleur

Le logiciel pour la pièce « Étude élastique » a été créé dans l'environnement de programmation Max/MSP (8) en utilisant sa bibliothèque pour la création visuelle « Jitter » et l'objet *analyzer~* de Tristan Jehan (9). Le fonctionnement général du logiciel est celui-ci : d'abord, la bande son de la pièce est joué et analysée en temps réel. Cette analyse est faite par l'objet *analyzer~* ? qui sort les valeurs de brillance, de bruit et d'intensité psychoacoustique ? et par un algorithme qui extrait la différence d'amplitude entre les canaux gauche et droite. Les données issues de l'analyse sont ensuite mises en échelle et utilisées pour piloter neuf variables d'une scène 3D qui montre un système de particules visuelles crée avec les objets jit.p.shiva et jit.p.vishnu. Les variables contrôlées dans la partie visuelle sont la quantité de particules générées, une variation de cette quantité, leur couleur, leur durée de vie, la déviation par rapport au plan vertical, la couleur du fond, l'angle de la caméra et la transparence et la position de la caméra.

Pour rendre plus expressive la mise en relation entre son et image, j'ai intégré un système qui permet de varier ces mises en correspondance au cours de la pièce. Les changements de ces mises en correspondance sont déclenchés par une abstraction qui suit le déroulement temporel de la bande son avec une horloge. Les valeurs minimales et maximales de chaque variable sont gardés dans une base de données à l'intérieur de cette abstraction et utilisés pour moduler la mise en correspondance entre les données d'entrée et les données de sortie dans des moments définis préalablement.

Figure 2

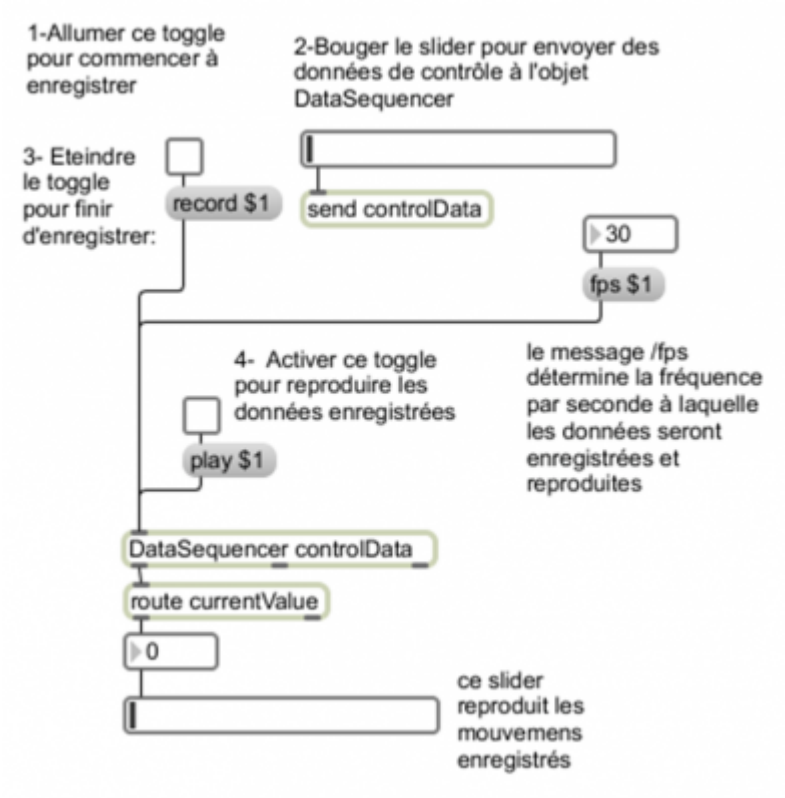


Fonctionnement général du logiciel d'« Étude élastique »

Bien que ce logiciel présente une performance très stable en concert, la consommation de CPU monte considérablement au moment de jouer en raison de la quantité de calculs qu'il faut faire à l'ordinateur seulement pour jouer les images, de sorte qu'il n'est pas possible d'enregistrer l'animation produite sur le disque dur pendant que l'on joue la pièce en temps réel.

Cette situation m'a conduit à chercher une solution pour enregistrer le rendu visuel en temps différé. Dans le but d'éliminer des processus qui pourraient contribuer à monter la consommation du CPU, j'ai décidé de suivre une démarche où l'on pouvait enregistrer les données issues de l'analyse dans des fichiers texte et, postérieurement, jouer la pièce avec les données enregistrées. Pour faire ceci, j'ai créé une abstraction appelée « dataSequencer » qui permet d'enregistrer les données qu'elle reçoit dans un fichier texte et de les reproduire à une vitesse déterminée par l'utilisateur (figure 3).

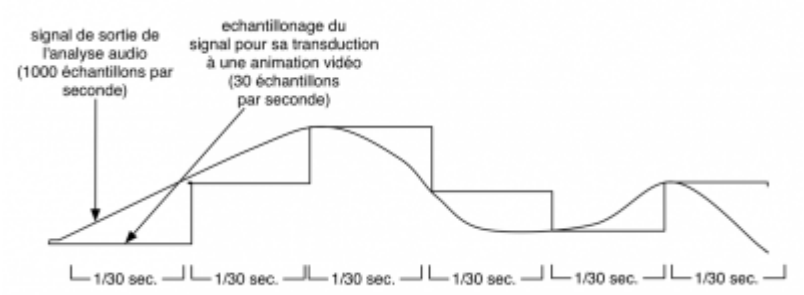
Figure 3



Patch d'aide de l'objet DataSequencer

Dans ce cas, j'ai voulu utiliser les données enregistrées pour faire un enregistrement vidéo de l'animation produite à une vitesse de 30 fps. C'est pourquoi, afin de les reproduire postérieurement, j'ai enregistré les données reçues à une vitesse d'échantillonnage de 1/30 secondes.

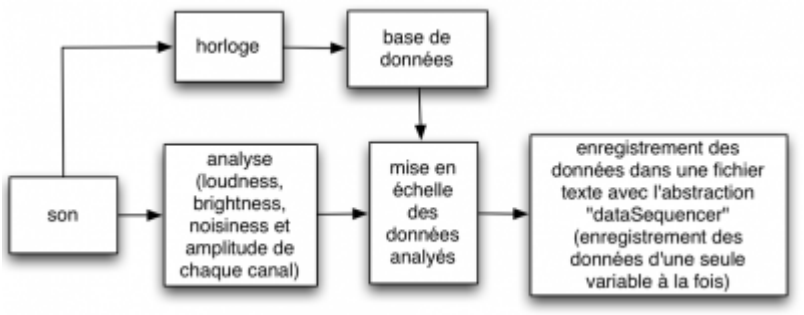
Figure 4



Échantillonnage des données d'analyse à 30 images par seconde

Dans un premier essai, j'ai voulu enregistrer les données de toutes les variables en même temps, mais l'activité du CPU montait trop et, par conséquence, le processus devenait trop instable. Pour cette raison et afin de prévenir des imprécisions dans la capture, il m'a fallu enregistrer les données de contrôle de chaque variable de manière indépendante (figure 5). C'est-à-dire, pour enregistrer correctement toutes les données issues de l'analyse en temps réel sans avoir des inconsistances produites par le sur-chargement du processeur, j'ai dû répéter ce processus pour chacune des variables de contrôle que j'ai utilisée.

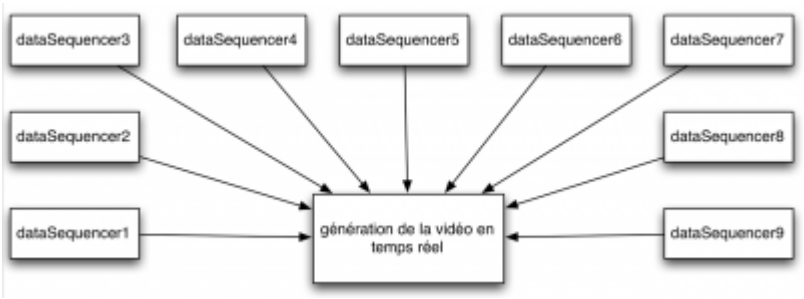
Figure 5



Enregistrements des flux de données sortis de l'analyse de la bande son

Après avoir enregistré les données de chacune des variables, il a été possible de jouer la vidéo en temps réel en obtenant et en reproduisant les contenus des fichiers texte à 30 fps avec plusieurs instances de mon objet dataSequencer (figure 3). Le rendu visuel obtenu a été le même que l'on obtient en faisant l'analyse en temps réel et l'activité du CPU a descendu considérablement, mais pas encore suffisamment comme pour permettre d'enregistrer le rendu visuel sur le disque dur en temps réel.

Figure 6

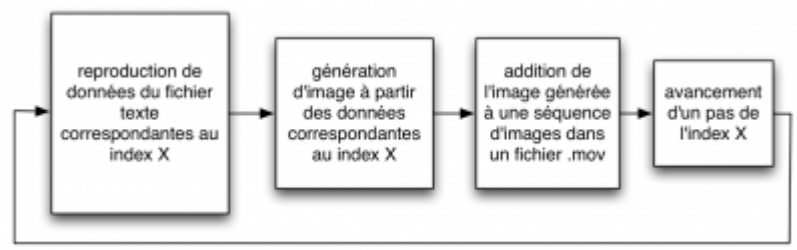


Reproduction des données enregistrées avec plusieurs instances de « dataSequencer »à une vitesse de 30 fps.

La deuxième méthode que j'ai essayée pour enregistrer le rendu dans un fichier vidéo a consisté à générer le film plus lentement qu'en temps réel en enregistrant une image à la fois sur un fichier en format .mov. Pour faire ceci, il a fallu d'abord reproduire les données stockées très lentement (dans ce cas, je l'ai fait à une vitesse d'une image par seconde) et, ensuite, garder chacune des matrices produites dans un fichier .mov à l'objet jit.qt.record (figure 7).

Le résultat obtenu avec cette stratégie ne fut pas entièrement satisfaisant car, bien que la structure générale de la vidéo produite soit restée très proche du rendu produit en temps réel, la synchronisation avec la bande son n'a pas fonctionné correctement. Après plusieurs échecs, je suis arrivé à l'hypothèse que ce problème a été le résultat d'une faille de conception de Jitter. Cependant, je n'ai pas pu le vérifier à cause du fait qu'il s'agit d'une bibliothèque à code fermé. Ceci a été la raison principale pour changer d'environnement de programmation pour l'élaboration du deuxième exemple présenté dans cet article.

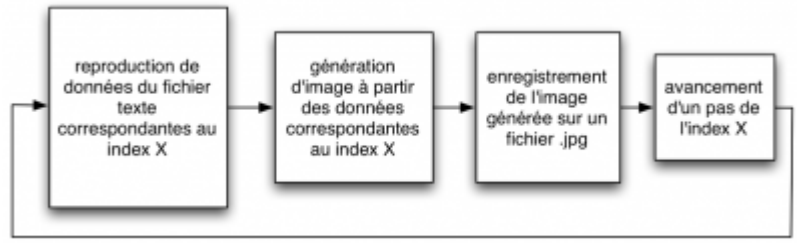
Figure 7



Processus pour l'enregistrement d'une image à la fois sur un fichier vidéo

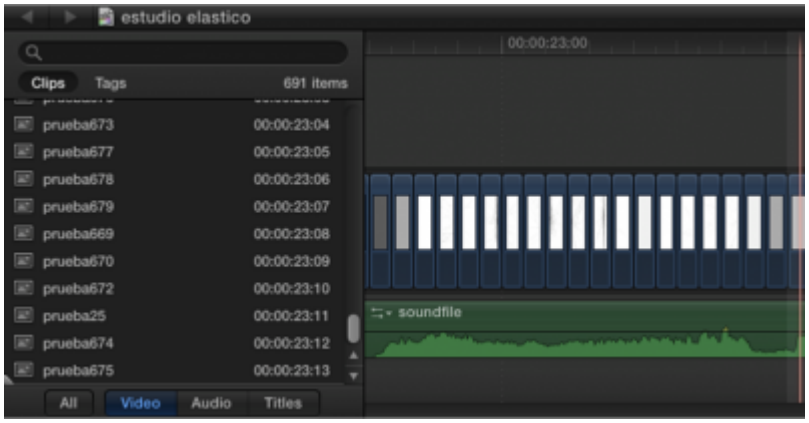
La troisième et dernière démarche que j'ai suivi pour faire la vidéo du rendu visuel a été d'enregistrer chaque image comme une photo en format .jpg (figure 8) et, postérieurement, d'introduire des ensembles de photos ? autant que possible sans saturer le processeur ? sur une session de Final Cut. Ceci m'a permis de faire des ajustements sur la séquence visuelle « à la main » pour résoudre les problèmes de synchronisation (figure 9) et d'obtenir un rendu très fidèle par rapport au rendu en temps réel.

Figure 8



Processus d'enregistrement de chaque image sur des fichiers .jpg indépendants

Figure 9



Montage des images générées

5. Cas d'étude n° 2 : le geste comme contrôleur

L'idée générale de cette expérience a consisté à créer une pièce basée sur l'utilisation de la transduction geste-image comme principe générateur de forme de manière à ce que le rendu visuel ne soit pas possible d'obtenir que par l'utilisation d'une approche en temps différé. Une vidéo pour démontrer les possibilités

de cette démarche a été développée avec un ensemble d'outils créés sur Processing et Max/MSP. Les deux environnements ? Max et Processing ? ont été mis en communication par le protocole OSC.

La transduction entre le geste et la vidéo a été divisée en trois étapes (figure 8) :
a) Enregistrement des données produites par le geste de l'utilisateur avec le patch *PARTIKL_Control_2*,
b) Lecture en temps différé des données enregistrées afin de générer les images d'une animation visuelle avec le sketch *PARTIKL2_demo*,
c) Création d'une vidéo à partir des images générées dans l'animation avec le logiciel *_2_Cargando cuadros a pelicula*.

Figure 10

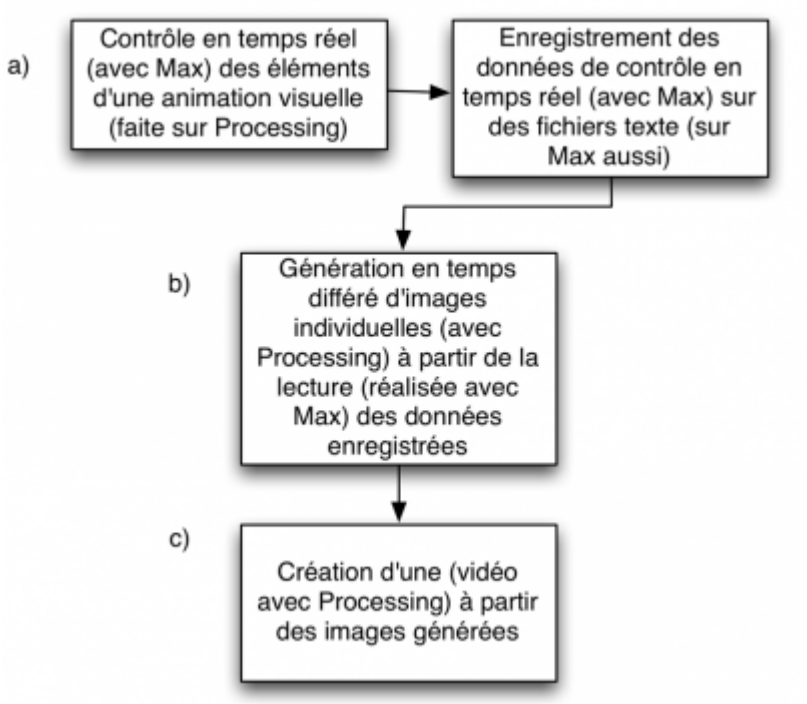


Diagramme des pas suivis dans l'élaboration de la vidéo

5. 1. Description des logiciels utilisés

PARTIKL_Control_2 ? Un patch Max pour la performance, l'enregistrement et la reproduction des données de contrôle qui seront envoyées à Processing.

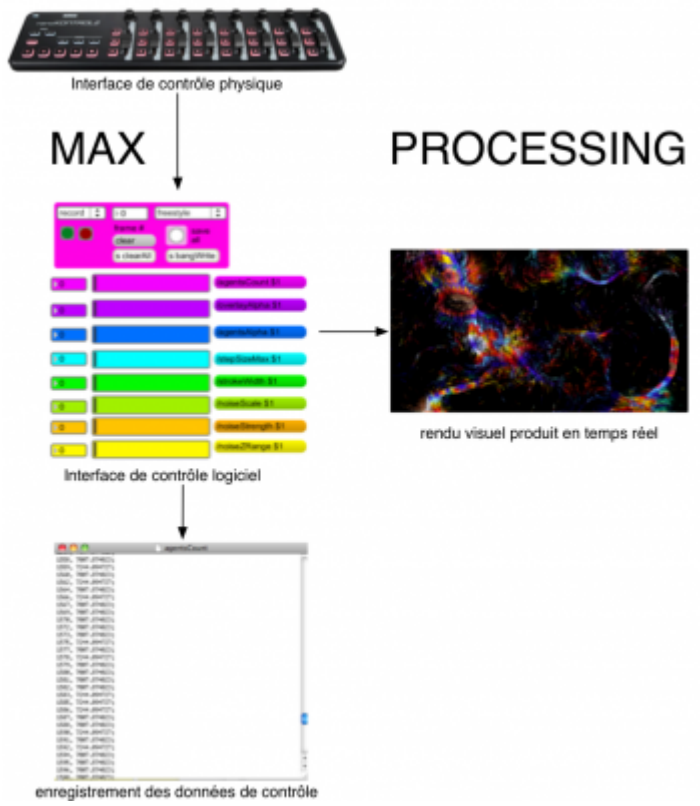
PARTIKL2_demo. Un logiciel fait sur Processing. Il s'agit du générateur de la partie visuelle du système. En le démarrant, il présente un menu qui permettra de choisir entre deux options : mode d'improvisation et mode d'enregistrement des images. Les images produites sont faites par le contrôle d'un système de particules.

Mode d'improvisation du logiciel *PARTIKL2_demo :*
Ce mode permet de montrer le rendu visuel de la performance faite sur le logiciel *PARTIKL_Control_2 ?*, dans Max /MSP. Il existe une grande quantité de variables qui peuvent être contrôlées dans ce système de particules comme, par exemple, leur quantité, leur transparence, leur vitesse de déplacement, la quantité de bruit brownien appliquée à leur trajectoire, etc.

Pour personnaliser le système, j'ai utilisé un contrôleur physique à 8 sliders. Ceci m'a permis de manipuler le patch *PARTIKL_Control_2 ?*, tout en ayant un retour visuel sur l'état des données que l'on

envoie et que l'on reçoit de Processing. Ce système offre un environnement suffisamment manipulable pour composer des séquences d'images en coordination avec le logiciel *PARTIKL2_demo*, fait sur Processing.

Figure 11



Enregistrement des données produites par les actions de contrôle de l'utilisateur (mode improvisation)

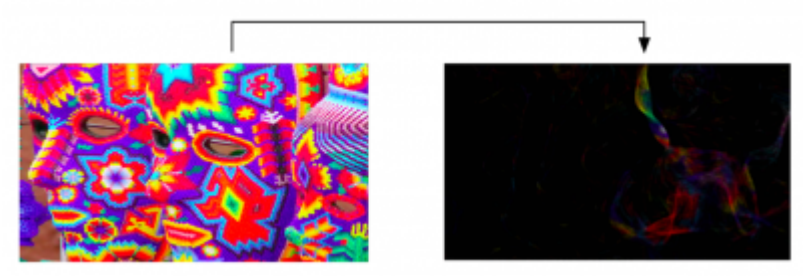
5. 2. Utilisation de système à code ouvert

Si j'ai décidé de travailler cette fois avec Processing plutôt qu'avec Jitter, c'est parce que dans Processing je peux accéder aux codes alphanumériques qui me donnent la possibilité comme utilisateur de comprendre et de modifier les opérations réalisées par chaque objet, ou plutôt sur sa représentation numérique, de manière beaucoup plus détaillée qu'en utilisant les objets à code fermé de Max/MSP.

5. 3. Transduction de couleurs entre deux images

Ce système permet de lire un film en temps réel et de faire une assignation des couleurs du film lu aux couleurs de l'animation crée. C'est-à-dire, si une particule se trouve dans des coordonnées données, elle prendra la couleur que le film aura dans cette même coordonnée. Ceci est intéressant parce qu'il s'agit d'une transduction entre deux espaces de représentation similaires (deux matrices de pixels) d'une même modalité (figure 10).

Figure 12

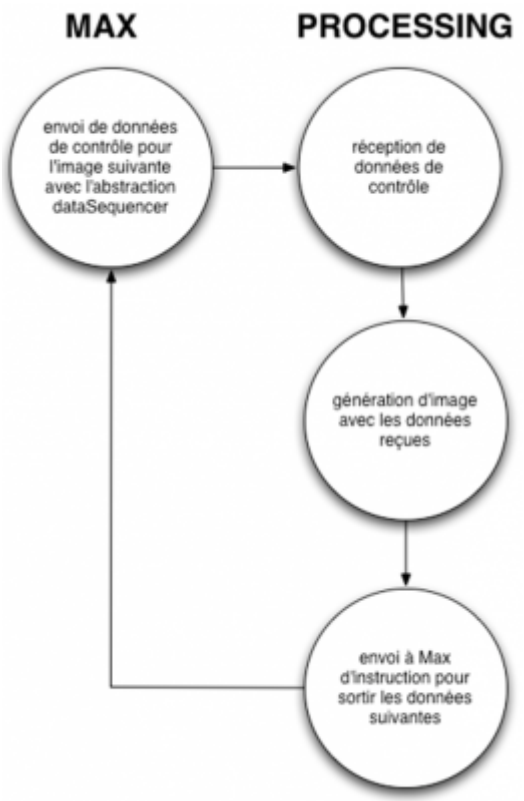


Transduction de données entre deux matrices de pixels

5. 4. Mode d'enregistrement du logiciel *PARTIKL2_demo*

Il sert à sauver sur des fichiers .jpg les images du rendu visuel produit par les données de contrôle enregistrées avec le mode d'improvisation. La communication entre Max et Processing par OSC devient un élément clé dans cette opération. Comme je l'avais mentionné antérieurement, l'abstraction dataSequencer, fait sur Max/MSP, permet non seulement d'enregistrer des séquences de données sur des fichiers texte, mais aussi de reproduire ces données. La reproduction de données sur Max se réalise en coordination avec la génération et l'enregistrement des images sur Processing, ce qui permet d'établir un boucle où, en premier lieu, Max envoie à Processing les données correspondantes aux valeurs des variables dans un instant donné ; ensuite, Processing génère l'image respective à partir des données reçues ; à continuation, Processing enregistre sur un dossier l'image générée dans un fichier .jpg et, finalement, Processing envoie à Max l'instruction de lui envoyer les données de contrôle pour l'image suivante... et ainsi de suite jusqu'à ce que Processing arrive à la lecture de la dernière image de la séquence (figure 12).

Figure 13



Coordination d'opérations nécessaires pour l'enregistrement d'images à partir de la reproduction de

données enregistrées dans l'improvisation

5. 5. Réduction de la performativité

Le rendu dans ce second cas d'étude devient beaucoup plus difficile à calculer en temps réel pour l'ordinateur à cause du nombre d'opérations qui doivent s'effectuer pour gérer toutes les opérations nécessaires pour rendre la vidéo (surtout quand on augmente la quantité de particules). Ceci fait que le rendu en temps réel devient plus lent et, ainsi, l'utilisateur perd un certain degré de ce que Golan Levin a bien appelé *performativité*, un terme qui fait référence à l'usage de l'action humaine pour la manipulation dans l'interaction homme-machine et qu'il définit comme *le degré dans lequel un système permet d'établir une boucle de rétroaction avec leurs utilisateurs pour qu'ils explorent leur potentiel en tant qu'acteurs dans une oeuvre ouverte* (Levin, 2009) [(10)].

Comment pourrait-on résoudre ce problème ? Il me semble que la seule option est de trouver des compromis. Dans ce cas en particulier, il faudrait donner à l'utilisateur quelques options pour descendre le coût de calcul quand il est en « mode improvisation ». On pourrait faire ceci par exemple, au moyen d'un menu avec des fonctions qui réduisent soit la résolution de l'image rendue, soit le nombre de plans de couleur utilisés ou, encore, une combinaison de ces deux techniques. Dès lors, l'utilisateur pourrait choisir sur quelle ou quelles variables il voudrait agir à un moment spécifique de son processus compositionnel. Cependant, l'impossibilité de ne pas pouvoir gérer tous les processus en temps réel à partir d'une certaine limite suppose déjà un handicap incontournable.

Enregistrement des images générées avec le logiciel *_2_Cargando cuadros a pelicula :*

J'ai développé ce logiciel pour faire un film à partir des images produites dans l'étape antérieure. Quoi qu'on puisse penser à enregistrer le rendu visuel directement sur un fichier vidéo, il n'a pas été possible de le faire dans ce cas-là à cause d'une contrainte dans la quantité de vidéos que l'on pouvait jouer simultanément avec la bibliothèque de Processing que j'ai utilisée (il me faudrait probablement rappeler ici que, pour la génération d'images, nous avions déjà une vidéo qui tourne pour affecter les couleurs de particules). Bref, il m'a fallu suivre cette approche d'enregistrer les images de façon individuelle parce qu'une vidéo n'aurait pas pu être produite en temps-réel avec les outils utilisés.

6. Cas d'étude n° 3 : l'image comme contrôleur

Après avoir fait la vidéo, j'ai choisi de suivre une approche où l'on pouvait, d'abord, diviser l'image en deux régions, ensuite, extraire la quantité de mouvement dans chacune et, finalement, utiliser les données extraites pour contrôler la vitesse de lecture d'un échantillon sonore dans chaque canal.

Parce que le temps de calcul du mouvement dans la vidéo ralentissait le processeur, il n'était pas possible de transduire les données issues de l'analyse à Max avec une vitesse constante. Ceci produisait, en conséquence, un ralentissement du résultat sonore et une impossibilité de synchroniser la musique produite avec la vidéo originale.

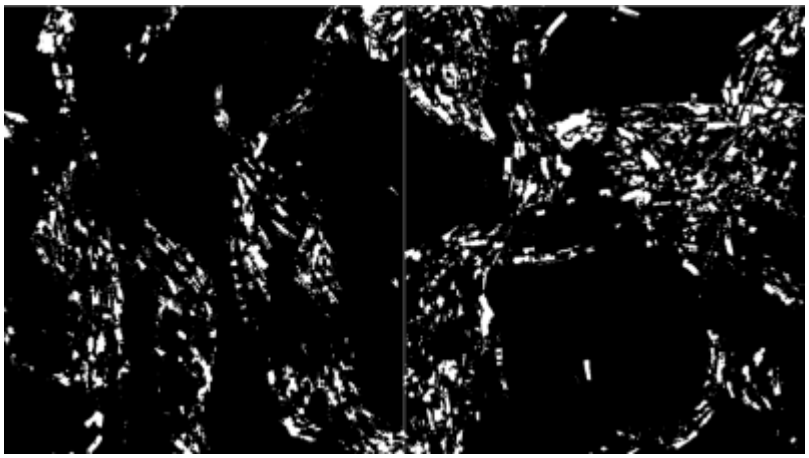
Pour résoudre ce problème, j'ai utilisé une approche en temps différé similaire à celle utilisée dans le premier exemple : j'ai enregistré les données issues de l'analyse d'image dans des fichiers texte avec l'objet dataSequencer et, postérieurement, j'ai reproduit ces données à la vitesse correcte et enregistré la musique produite dans deux fichiers son avec Max/MSP. Finalement, j'ai fait un montage de la vidéo et de la bande son avec Final Cut.

6. 1. Détection du mouvement

L'estimation de la quantité de mouvement a été faite avec le logiciel *_7_Detectando_movimiento_dos_markers_test7*, qui a été programmé avec Processing. L'algorithme de

détection de mouvement que j'ai utilisé dans ce programme est basé sur la comparaison entre les pixels d'une image avec ceux de l'image antérieure. Il s'agit donc d'une lecture de toute la matrice pixel par pixel afin d'obtenir la différence des valeurs représentant les couleurs des deux images. Si, par exemple, la différence entre les valeurs représentant les couleurs d'un pixel dans une image et les valeurs représentant les couleurs du même pixel dans l'image antérieure dépassent un certain seuil, ce pixel sera marqué en blanc (figure 13). La quantité de mouvement de chaque région sera, donc, la quantité de pixels marqués en blanc.

Figure 14



Division de l'image en deux régions pour la captation de mouvement

Le logiciel *_7_Detectando_movimiento_dos_markers_test7* (fait sur Processing) envoie les données issues de l'analyse par OSC au patch de Max/MSP *recibiendoMovimientoDeProcessing_ dosMarkersConDosBuffer8_test7.maxpat*, qui les enregistre, à son tour, dans deux fichiers texte au moyen de l'abstraction « dataSequencer ». Une fois enregistrées, ces données peuvent être lus *à tempo* ? c'est-à-dire, à une vitesse de 30 données par seconde ? pour piloter n'importe quel processus sonore. Dans la vidéo de démonstration, j'ai utilisé ces données pour piloter la lecture de deux échantillons (joués dans les canaux gauche et droit, respectivement) avec Max/MSP pour créer un espace sonore stéréo qui réagit au mouvement de la vidéo.

7. Conclusions

Le potentiel que la transduction ouvre pour la création audiovisuelle ? qu'elle soit en temps réel ou en temps différé ? offre des *modèles* pour la création de rapports structurels entre différents espaces de représentation. Il s'agit d'une approche artistiquement viable car elle instaure des critères spécifiques d'engendrement. En d'autres mots, l'utilisation de la transduction pour la création de musique visuelle a comme but *la création de quelque chose de neuf* au moyen d'un processus génératif où la forme et le comportement d'un objet sont utilisés comme des repères morphologiques dans une composition.

Comme Emmerson le signale (Emmerson, 2007, p. 42), les choix compositionnels qu'un artiste doit faire pour utiliser des modèles dans son processus créatif sont, au moins, quatre : l'identification du modèle spécifique à utiliser et de son comportement, le choix des variables qui vont correspondre à celles du modèle choisi et la définition de l'échelle de leurs relations. Mais par l'usage des stratégies compositionnelles basées sur la transduction de données, la prise de décisions concernant le compromis entre la performativité d'un système et la qualité (ou résolution) des matériaux utilisés devient incontournable.

Avec l'ordinateur, on peut simuler toutes sortes de modèles et de comportements qui vont au-delà des limitations physiques, mécaniques et optiques imposées par les instruments de musique visuelle traditionnels (Levin, 2000, p. 33) [(11)] mais, en temps différé, son utilisation facilite la création de polyphonies de transductions entre diverses modes de représentation et la génération de matériaux ? tant

de détail que de globalité ? qui ne pourraient pas être créées autrement. Une approche de la transduction en temps différé apparaît pour le moins comme un bon compromis pour le cas des musiques visuelles.

Exemples (logiciels et vidéos) : http://www.carloslopezcharles.com/default/ideas.html

---

1. Version électronique disponible sur www.flong.com (consulté le 29 juin, 2012).

2. Levin (2009) donne un état de l'art très complet de ce type de systèmes.

3. Emmerson, (2007, p. 117-142) donne un état de l'art très complet de l'évolution des transducteurs utilisés dans la musique électronique.

4. Version électronique disponible sur: http://www.enactivenetwork.org (consulté le 15 juin, 2012)

5. Vaggione (1995, p. 2) considère les différents modes de représentation (graphiques ou textuelles) comme des interfaces des « traductions », ou plus précisément, des « réécritures » entre les divers niveaux opératoires accessibles au compositeur et celui des codes numériques manipulés par la machine.

6. Version électronique disponible sur : http://www.enactivenetwork.org (consulté le 15 juin, 2012)

7. Version électronique disponible sur http://jim.afim-asso.org/jim96/actes/vaggione/VaggioneTEXTE.html#Tahil

8. Logiciel disponible sur le site http://cycling74.com/products/max/ (consulté le 15 juin, 2012)

9. Objet téléchargeable sur le site http://web.media.mit.edu/~tristan/ (consulté le 15 juin, 2012)

10. Version électronique disponible également sur http://www.flong.com/texts/essays/see_this_sound_old/ (consulté le 30 décembre, 2010).

11. Version électronique disponible sur www.flong.com (consulté le 29 juin, 2012)

Pour citer ce document:
Carlos López Charles, « La transduction dans la création de la musique visuelle en temps différé », *RFIM* [En ligne], Numéros, n° 2 - automne 2012, Mis à jour le 28/09/2012
URL: http://revues.mshparisnord.org/rfim/index.php?id=202