PARTITIONS RETROACTIVES AVEC IANNIX

Guillaume Jacquemin, Thierry Coduys

PARTITIONS RETROACTIVES AVEC IANNIX

Guillaume Jacquemin
Association IanniX
guillaume@iannix.org

Thierry Coduys
Association IanniX
thierry@iannix.org

RÉSUMÉ

Le logiciel IanniX offre la possibilité d'écrire et de composer des partitions exécutées en temps réel qui émettent des messages de contrôle — synthèse, effets, etc. — et reçoivent des messages de structure — ajout, altération d'objets —. La combinaison de ces entrées/sorties et des interfaces disponibles *(scripts, réseau, interface graphique)* permet de composer des partitions de contrôle de paramètres, des partitions réactives à des stimuli, des partitions stochastiques, génératives ou encore interactives où IanniX réalise des *mappings* complexes et temporels.

Les partitions rétroactives développées dans cet article représentent un nouveau champ de composition, où la sortie de la partition est bouclée sur son entrée. Elle entre parfois en résonance, parfois s'emballe mais reproduit globalement les phénomènes observés dans les systèmes bouclés.

L'article présente d'abord l'état des lieux de ce phénomène graphique sonifié par IanniX pour ensuite le proposer comme un nouveau processus génératif d'écriture musicale.

L'article se conclut sur une première réalisation, *Singularités*, développée avec ce mécanisme en utilisant un robot industriel ainsi que le synthétiseur Cosmos*f*.

1. IANNIX

1.1. Objets fondamentaux pour la composition

IanniX est un séquenceur graphique open source, inspiré des travaux de Iannis Xenakis et destiné à la création numérique [1]. Le logiciel propose une écriture polytemporelle d'évènements statiques (cues, déclencheurs, etc.) et dynamiques (variation de valeurs, automations, etc.) destinés à des environnement dédiés (Processing, PureData, SuperCollider, Max…).

Pour composer une partition graphique dans IanniX, une palette restreinte de trois objets fondamentaux et distincts est disponible :

- les *triggers* qui déclenchent des événements statiques (ex : déclenchement d'un fichier son) ;
- les *courbes*, évènements dynamiques qui sont des suites de points dans l'espace tridimensionnel (ex : changement d'un paramètre de synthèse) ;
- les *curseurs* qui évoluent sur des courbes et progressent en fonction du temps.

Ces objets émettent des messages (les *triggers* en émettent lorsqu'ils sont déclenchés, les *curseurs* en émettent périodiquement lorsqu'ils progressent sur une courbe) qui sont envoyés via des *interfaces* vers d'autres applications ou matériels.

1.2. Simplification des interfaces

L'écriture des partitions dans IanniX s'effectue via des *messages* envoyés dans des *interfaces* [2] parmi lesquelles figurent :

- **l'interface graphique utilisateur** (GUI),
- **le réseau** : Open Sound Control (OSC) [3] et UDP brut, pour les applications temps réel ; MIDI, pour les contextes de contrôle simples ou pour l'interfaçage sur des DAW ; RS232, pour le contrôle de hardware ; HTTP GET, pour le contrôle de pages Web ; WebSockets pour l'interfaçage en HTML5,
- **les scripts** JavaScript basés sur la norme ECMAScript (norme ECMA-262 [4]),
- **la boucle locale** qui permet aux partitions de s'envoyer des messages.

L'écriture de partitions repose donc sur l'échange de messages via ces protocoles *(y compris lorsque l'utilisateur manipule l'interface graphique, des messages OSC sont automatiquement générés en interne, de manière transparente pour l'utilisateur)*. Cependant, par souci de clarté, de documentation et aussi pour faciliter l'interfaçage avec des outils tiers, une nouvelle fenêtre, le *Helper*, présenté en figure 1, affiche les messages venant de l'interface graphique et permet de copier dans le presse-papier des snippets pour Processing et Max qui reproduisent l'opération réalisée manuellement dans l'interface graphique.

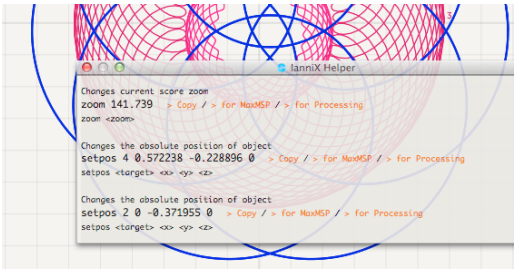

Figure 1. Capture d'écran de IanniX de la fenêtre *Helper* montrant les messages générés par

l'interface graphique utilisateur, ici un zoom, puis deux déplacements d'objets.

Enfin, l'architecture de IanniX 0.9 [5] a aussi été repensée et optimisée pour recevoir et envoyer massivement des messages.

1.3. Conséquences sur la sauvegarde des partitions

La multiplicité des interfaces de création et d'altération de partitions dans IanniX a soulevé le problème de la sauvegarde en fichier.

Prenons par exemple une partition créée dans l'interface JavaScript, sur laquelle le compositeur ajoute et modifie quelques éléments via l'interface graphique et finalement décide de rendre la partition interactive avec des capteurs [6]. Lorsque tous ces facteurs (du code, des modifications dans l'interface graphique et des données de capteurs) altèrent ou génèrent la partition, une simple sauvegarde de l'état de la partition est insuffisante. Par exemple, si le rôle d'un des capteurs est de supprimer une partie de la partition, la sauvegarde après la performance sera inéluctablement partielle…

IanniX propose donc une solution en unifiant le format des sauvegardes : tout document IanniX est désormais un script JavaScript, même si la partition est par exemple créée uniquement par l'interface graphique. Le code généré par IanniX et par l'utilisateur est ensuite ventilé dans quatre procédures en fonction de l'origine des modifications *(procédures appelées au chargement dans l'ordre ci-dessous)* :

- **makeWithScript()** : code JavaScript utilisateur ;
- **madeThroughGUI()** : section auto-générée par IanniX retranscrivant les ajouts et modifications réalisés dans l'interface graphique ;
- **madeThroughInterfaces()** : section auto-générée par IanniX retranscrivant les ajouts et modifications réalisés par des capteurs ou des interfaces réseau
- **alterateWithScript()** : code JavaScript utilisateur permettant d'altérer l'ensemble de la partition juste avant la fin du chargement.

Une rétrocompatibilité est évidemment maintenue pour l'ouverture de documents IanniX créés avant la version 0.9.

2. CLASSIFICATION DES PARTITIONS

Le champ des possibilités d'écriture ouvertes par les messages et protocoles de IanniX peut se classifier selon l'évolution de la partition au fil de son interprétation. Évidemment, le compositeur est amené à hybrider ces méthodes d'écriture dans la conception de son œuvre ; cette classification vise surtout à comprendre les grandeurs mises en jeu au niveau des entrées, des sorties et des interactions avec un environnement.

2.1. Partition de contrôle

Une **partition de contrôle** est une partition qui contrôle une ou plusieurs applications tierces. Elle **ne répond à aucun stimulus** externe. L'interprétation de la partition est autonome, reproductible et déterministe. Ce type de partition est très proche d'une surface de contrôle MIDI.

Ce type de partition permet par exemple d'écrire des courbes de contrôle d'effets, de synthèse (à la manière de l'UPIC) ou de spatialisation (figure 2).

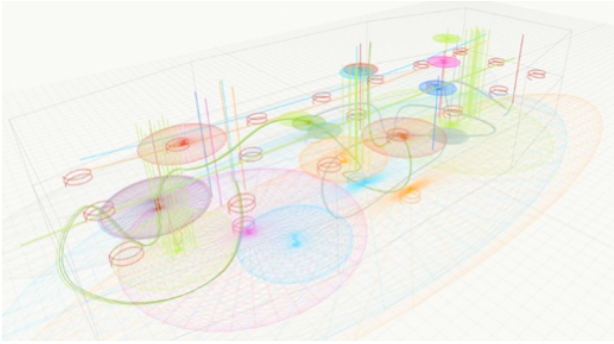

Figure 2. Capture d'écran de IanniX de la partition de contrôle de spatialisation pour *World Expo* de *Charles de Meaux ;* chaque courbe représente le déplacement d'une source sonore.

2.2. Partition réactive

Une **partition réactive** est une partition qui **réagit à des stimuli externes** mais qui ne produit aucun message de contrôle. Elle est interprétée par un humain qui la lit et/ou remplit une fonction esthétique et graphique.

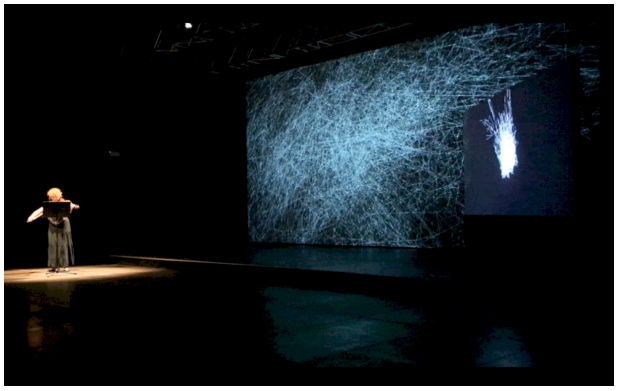

Figure 3. Capture d'écran de IanniX de la partition réactive esthétique pour *Influences* de *Davide Gagliardi* et *Victor Nebbiolo di Castri* ; les courbes sont générées en réponse à l'alto sur scène.

2.3. Partition stochastique

La **partition stochastique** est une partition dont le processus global est prévisible, même si les évènements qui la composent sont aléatoires.

L'usage du JavaScript dans IanniX permet d'écrire les règles de contrôle et de distribution stochastiques qui régissent les grandeurs aléatoires de la partition. Associés à des librairies JavaScript externes, les script IanniX permettent d'écrire rapidement des tirages aléatoires sous contraintes, des implémentations de la théorie des jeux ou des fonctions de distribution [7].

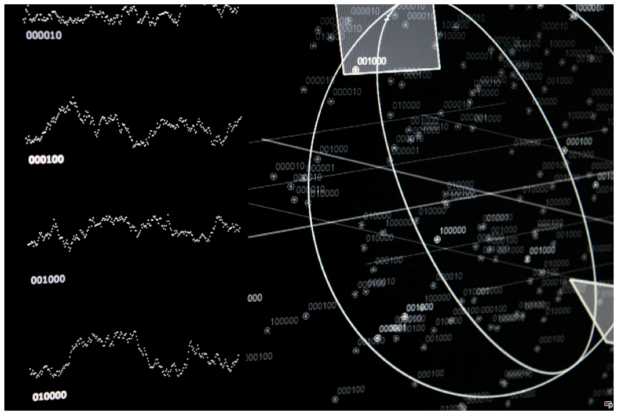

Figure 4. Capture d'écran de IanniX de la partition stochastique pour *Ultimarium* de *Thomas Bouaziz* ; la partition effectue des tirages aléatoires d'hexagrammes Yi King.

2.4. Partition générative

Une **partition générative** est une partition générée par des algorithmes, qui peut également évoluer d'elle-même de manière déterminée à l'avance ou non.

Tout comme les partitions stochastiques, le JavaScript offre une liberté d'écriture générative (figure 5) et permet aussi de reproduire des phénomènes biologiques grâce à des librairies externes dédiées (figure 6).

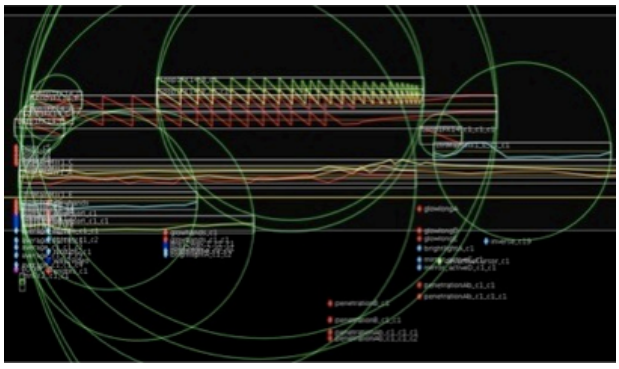

Figure 5. Capture d'écran de IanniX de la partition générative pour *Eros3* de *Joachim Montessuis* ; la partition n'évolue pas au fil du temps.

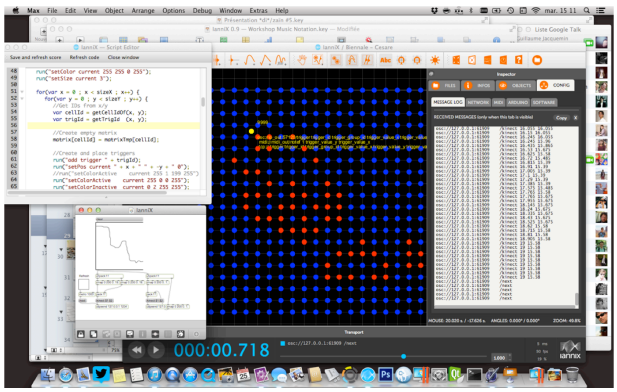

Figure 6. Capture d'écran de IanniX de la partition générative pour *Still Life* de *Cesare Saldicco* ; modèle génératif évolutif reproduisant la propagation des virus.

2.5. Partition interactive

Une **partition interactive** fait intervenir la coopération de IanniX avec plusieurs entités (hommes ou applications) qui agissent mutuellement en ajustant leur comportement.

IanniX est alors l'intermédiaire entre plusieurs entités et régule, par l'écriture et la composition, les interactions entre ces entités. IanniX agit comme un dispositif classique de *mapping* (transformation d'une grandeur vers une autre) mais introduit une dimension de composition et d'écriture du temps fondamentale.

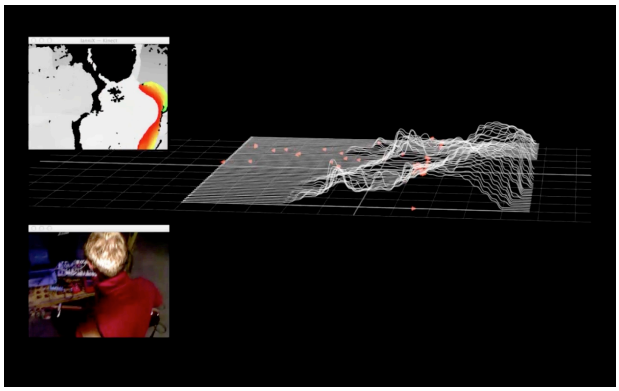

Figure 7. Capture d'écran de IanniX de la partition interactive pour *Fa Octothorp* de *Guillaume Jacquemin* et *Matthieu Ranc* ; une caméra Kinect capte une image 3D qui extrude un maillage de courbes dans IanniX sur lesquelles circulent des curseurs contrôlant la synthèse sonore.

2.6. Partition rétroactive

Une **partition rétroactive ou bouclée** est une partition dont l'interface de sortie est connectée directement à l'entrée, avec un ou plusieurs systèmes insérés dans la boucle.

Les valeurs émises par les objets de la partition à l'instant *t* contrôlent la structure et les objets de la

partition à l'instant *t+1*. Le délai de rétroaction entre deux itérations de calcul est initialement réglé à 5 ms dans IanniX mais reste modifiable par l'utilisateur (de 1 ms à 1 seconde).

Tout comme les systèmes bouclés, l'évolution de la partition *(répétition itérative en fonction du temps)* peut aboutir à plusieurs résultats.

2.6.1. *Rudiments de code IanniX / JavaScript*

Les partitions présentées dans la suite de l'article ont été écrites en JavaScript. Il convient alors de donner quelques rudiments de syntaxe.

IanniX respecte la norme JavaScript ECMA-262 ainsi que ses classes de base (Math, String…) auquel on ajoute une fonction spéciale permettant d'envoyer un message à IanniX : `run()`.

Les messages IanniX permettant de créer, modifier ou supprimer un élément de la partition respectent systématiquement la syntaxe : `<action> <ID objet> <paramètres>`.

Ainsi la commande `run("add curve 1")` permet créer une courbe qui aura l'ID #1 ; la commande `run("setPointAt 23 0 5 8");` permet de modifier le premier point (index n°0) de la courbe #23 et de le placer à la coordonnées 2D (5 ; 8).

2.6.2. *Amplification continuelle ou extinction progressive*

Dans le code suivant, le curseur fait évoluer en fonction de sa position, le point terminal de la courbe sur laquelle il progresse :

```
run("add curve 1");
run("setPointAt lastCurve 0 0 0");
run("setPointAt lastCurve 1 1 1");
run("add cursor 2");
run("setCurve current lastCurve");
run("setMessage current direct:// setPointAt lastCurve
      1 1 {cursor_yPos+1}");
```

La figure 8 montre que la partition s'étire verticalement au fil du temps tandis que la figure 9 (extraite à l'aide OSCulator) montre que l'agrandissement est linéaire.

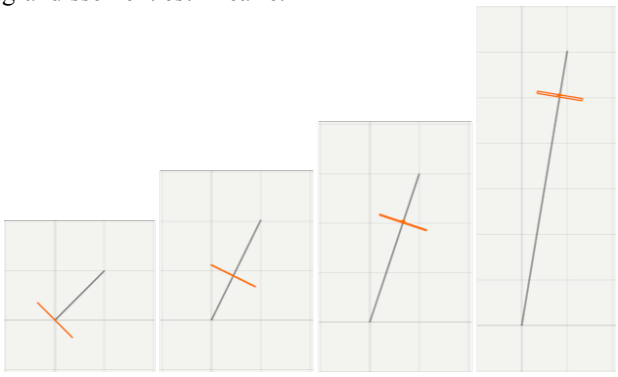

Figure 8. Captures d'écran de IanniX montrant le phénomène d'amplification continuelle à t = 0 sec., t = 1 sec., t = 2 sec. et t = 5 sec.

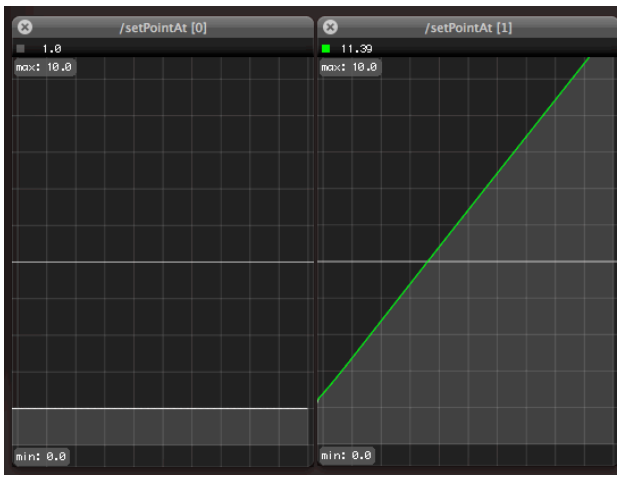

Figure 9. Progression linéaire de l'amplification (évolution des coordonnées *x* et *y* du point).

2.6.3. *Emballement*

Dans la partition suivante codée en JavaScript, le curseur fait évoluer en fonction de sa position, la taille de l'ellipse sur laquelle il progresse :

```
run("add curve 1");
run("setPointsEllipse lastCurve 1 1"); //Rayon = 1

run("add cursor 2");
run("setCurve current lastCurve");
run("setPattern current 0 0 1"); //Joue en boucle
run("setMessage current direct:// setResize lastCurve
      {cursor_xPos+1} {cursor_yPos+1}");
```

Un emballement va se produire car la figure va être redimensionnée dans l'espace des réels négatifs, et va donc se retourner (figure 10) ; de fait, le curseur ira également à rebours et la partition va entrer dans une résonnance non contrôlée (figure 11).

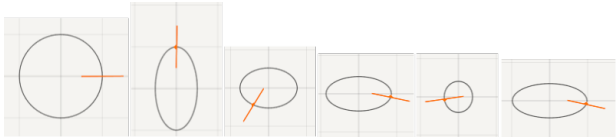

Figure 10. Captures d'écran de IanniX montrant les déformations de la figure dues à l'emballement du système

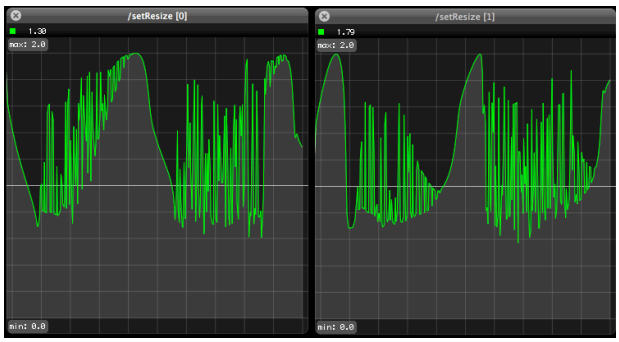

Figure 11. Emballement de la partition (visualisation du redimensionnement en *x* et en *y*).

2.6.4. *Régulation stable*

Pour cette dernière partition rétroactive, nous allons placer deux cercles avec deux curseurs. Le curseur du *cercle #1* contrôle le diamètre du *cercle #2* ; et le curseur du *cercle #2* contrôle le diamètre du *cercle #1*. Avec certaines conditions initiales *(c'est à dire le diamètre et la position initiale des cercles)*, les cercles entrent graphiquement en oscillation *(code fourni en annexe pour n cercles)*. La figure 12 propose quelques extraits graphiques de la partition tandis que la figure 13 montre l'évolution des rayons des cercles à l'aide d'OSCulator.

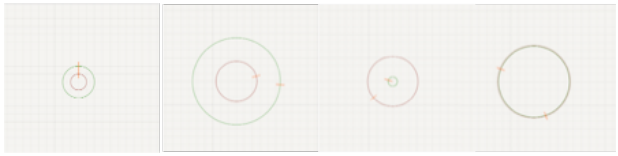

Figure 12. Régulation stable de la partition rétroactive, les cercles entre en oscillation / capture d'écran de IanniX

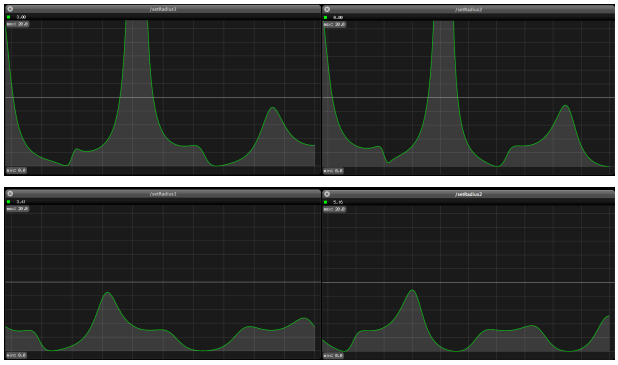

Figure 13. Régulation stable de la partition (visualisation des rayons à deux instants donnés).

2.6.5. *Du phénomène observé au processus d'écriture génératif*

Récurrences a été la première œuvre composée avec IanniX qui utilisait la rétroactivité. La pièce utilise six jeux de 8 courbes + curseurs qui contrôlent chacun un oscillateur indépendant. Chaque curseur d'un jeu de courbes contrôle l'amplitude et la durée de la courbe du jeu de courbes suivant.

La pièce a été jouée au centre DATABAZ d'Angoulême et la partition est distribuée librement avec l'application IanniX.

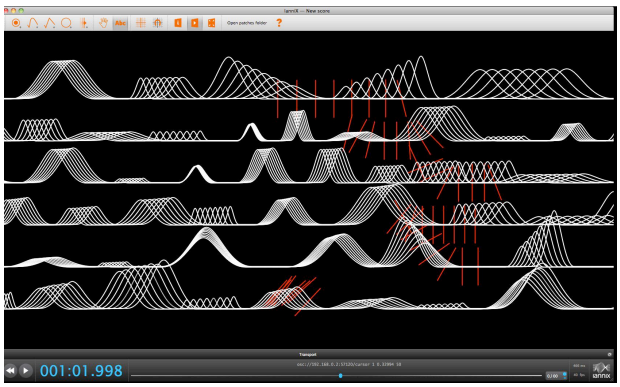

Figure 14. Capture d'écran de IanniX de la partition rétroactive *Récurrences* de *Thierry Coduys*.

3. SINGULARITES

3.1. Contexte

Les recherches menées sur les partitions rétroactives nous ont conduits naturellement aux systèmes bouclés et aux asservissements. Dans l'optique de créer une pièce dédiée à ces mécanismes, deux pistes nous ont paru intéressantes :

- l'asservissement dans le domaine de la robotique industrielle ;
- le feedback utilisé en synthèse sonore.

Après plusieurs séances de travail avec le compositeur *Sinan Bökesoy* (concepteur du synthétiseur stochastique inspiré des travaux de Iannis Xenakis, Cosmos*ƒ* [8] et auteur d'une publication sur l'utilisation des robots industriels en performance artistique [9]), une résidence autofinancée à *Büyükada* en Turquie s'est tenue du 29 juillet au 12 août 2013 afin d'esquisser les grands principes de la pièce et ses enjeux majeurs.

Figure 15. Capture d'écran du plugin Cosmos*ƒ*.

3.2. Intentions

La pièce *Singularités* amorcée lors de cette résidence va exploiter le principe de *singularité*.

En mathématiques, une **singularité** est un point, une valeur, dans lequel un objet mathématique n'est pas défini, par exemple une valeur où une fonction d'une variable réelle devient infinie.

En physique, une **singularité gravitationnelle** est un point spécial de l'espace-temps au voisinage duquel certaines quantités écrivant le champ gravitationnel deviennent infinies.

La **singularité technologique** est un point hypothétique de l'évolution technologique où il n'y a plus de progrès mais une explosion exponentielle de la science et des techniques.

En robotique, **les singularités** sont des points de l'espace que le robot ne peut atteindre. Contraint par ses moteurs et la rigidité de ses axes, le robot essaye d'atteindre les positions spatiales données par son opérateur tout en évitant les postures impossibles (les singularités). Ainsi, en donnant comme consigne au robot, des positions aux voisinages des singularités, il est contraint d'emprunter des itinéraires très complexes.

Ces singularités se retrouvant également dans les partitions rétroactives de IanniX (points, valeurs ou structures graphiques qui entraînent un emballement graphique de la partition), un dispositif très simple a été imaginé :

- IanniX prendra le rôle de l'opérateur et contrôlera les mouvements d'un robot (chorégraphie) ;
- le robot industriel ABB IRB120 sera équipé de capteurs embarqués ;
- Cosmos*f* sera utilisé pour la synthèse sonore en temps réel.

Figure 16. Robot industriel ABB-IRB120.

La rétroaction sera la suivante :

1. IanniX pilote le robot et donne des ordres de positions absolues,
2. le robot calcule les itinéraires pour atteindre ce point tout en évitant les singularités,
3. les capteurs placés sur le robot mesurent les rotations, accélérations, mouvements des axes (figure 17) et envoient ces informations vers Cosmos*f* qui va sonifier les mouvements du robot,
4. les mesures des capteurs du robot sont également renvoyées vers IanniX et altèrent la partition initiale, bouclant le système en rétroactivité.

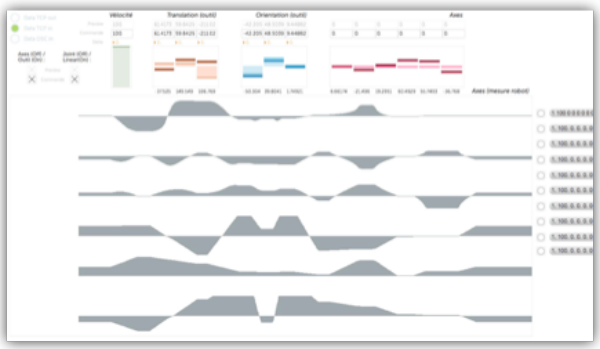

Figure 17. Mesures temps réel des capteurs sur le robot pour cinq consignes (déplacements) données au robot.

3.3. Prototypage

Dans le cadre du prototypage de la pièce, une caméra et un laser placés sur le robot permettent également d'obtenir un retour vidéo (caméra) et une projection du mouvement du robot sur un plan (laser).

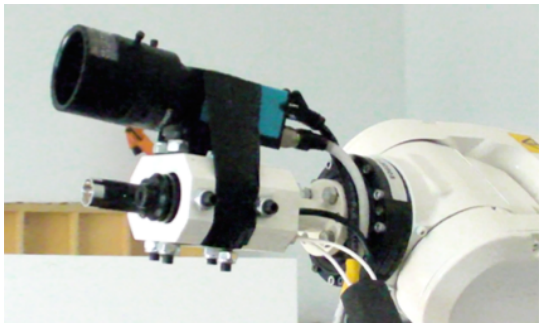

Figure 18. Caméra et laser sur la « main » du robot.

Enfin, le fonctionnement du robot étant extrêmement bruyant, l'outil RoKiSim [10] a permis afin de simuler et d'identifier les singularités, et ainsi d'écrire des partitions sans tester systématiquement avec le robot.

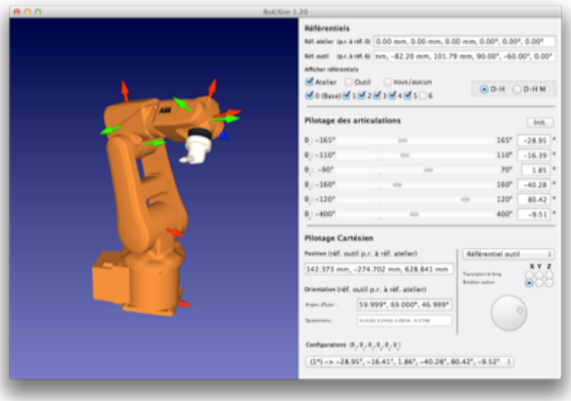

Figure 19. Logiciel *RoKiSim* permettant de simuler les comportements et singularités du robot.

4. CONCLUSION

Les partitions rétroactives ont été envisagées dans IanniX depuis quelques années. Elles permettent un auto-contrôle de la partition allant du simple événement rétroactif *(trigger qui arrête la partition ou saute à un timecode précis)* jusqu'à la mise en oscillation complexe des éléments constitutifs de la partition.

Au départ mal comprises, les partitions rétroactives constituent pour l'équipe de recherche IanniX une formidable opportunité de trouver de nouvelles approches dans la composition, même si l'écriture de telles partitions reste encore difficile à appréhender.

Au travers de l'amorce de la pièce *Singularités*, un premier socle prometteur [11] a été produit et la viabilité technique et esthétique est prouvée. L'environnement de travail étant économiquement lourd à cause du robot, des partenariats et des financements sont en cours.

5. REFERENCES

[1] Coduys, T. et Ferry G., "IanniX, aesthetical / symbolic visualisations for hypermedia composition", Sound and Music Computing, 2004

[2] Jacquemin, G., Coduys T. et Ranc M., "IanniX 0.8", Journées d'Informatique Musicale, Mons, 2012

[3] Wright, M. et Freed, A. "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers" Proceedings of the International Computer Music Conference 1997, Thessaloniki, Hellas, pp. 101-104.

[4] Norme ECMA-262 http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

[5] IanniX 0.9, http://www.iannix.org

[6] Jacquemin, G. et Coduys T., « Simone Beneventi + IanniX », 56ème Biennale de Venise, octobre 2012, http://www.labiennale.org/en/mediacenter/video/beneventi-int.html

[7] RandomJS, http://simjs.com/random.html

[8] Bökesoy S., "Synthesis of a macro sound structure within a self-organizing system", DAFX07 (Digital Audio FX Conference), Bordeaux 2007

[9] Bökesoy S., "1city 1001vibrations: development of a interactive sound installation with robotic instrument performance. ", NIME, Oslo 2011

[10] RoKiSim, http://parallemic.org/RoKiSim.html

[11] Vidéo du projet *Singularités*, http://vimeo.com/76981622

ANNEXE

Code de la partition rétroactive stable pour *n* cercles

```
function makeWithScript () {
   run("clear");

   var iMax = 2;
   for(var i = 0 ; i < iMax ; i++) {
       run("add curve " + (100+i));
       run("setPos current 3 3 0");
       run("setEquation current polar radius, TWO_PI*t,
              theta");
       run("setEquationParam current radius " + (1+i));
       run("setEquationParam current theta  0");
       run("setColorHue current " + map(i, 0, iMax, 0, 255)
              + " 255 128 255");

       run("add cursor " + i);
       run("setSpeed   current lock " + map(i, 0, iMax, 0.2,
              0.3));
       run("setCurve   current lastCurve");
       run("setPattern current 0 0 1");
       run("setBoundsSourceMode current 2");
       run("setMessage current 5, direct:// setEquationParam
              " + (101+i) + " radius cursor_xPos ");
   }
   run("setMessage current 5, direct:// setEquationParam " +
              100 + " radius cursor_yPos");
}
```