Fiche de renseignements processus de création
Projet Kinexperience

Titre : Kinexperience

Date : Juin 2018 (sprint 1)

Image mise en avant

Description courte
L'écriture chorégraphique contemporaine est marquée par les études du théoricien de la danse Rudolf Laban. Dans l'étude de la Choreutique, il présente la *kinésphère* : un icosaèdre qui indique *la limite naturelle de l'espace corporel de chacun* où le mouvement a lieu.

Dans ce projet on propose d'explorer les mouvements de danse dans le cadre de cette spatialisation géométrique. Au-delà de l'intérêt technique et de recherche avec la grille de lecture et d'écriture de la danse proposée par Laban, l'oeuvre joue avec une panoplie d'effets graphiques déclenchés par l'intensité et la direction des mouvements du corps.

Etapes de réalisation

Introduction
La première étape du projet, avant toute considération technique, c'est la naissance des idées ! Dans notre cas le processus a été assez simple : Ariane et moi (Alex) nous sommes rencontrés sur le Mirage Festival 2018 aux Subsistances. Intéressée par l'expérience en réalité virtuelle _*Transition* présentée,

Ariane m'a partagé son approche de la danse à travers les théories de la kinésphère et sa démarche de démocratisation la danse. Passionné d'art géométrique, il ne fallait pas plus que la mention de l'*icosaèdre* pour m'embarquer dans cette nouvelle aventure pluridisciplinaire.

Nous avons commencé à imaginer une expérience interactive où l'on permettrait à chacun de vivre l'oeuvre de son propre corps, d'explorer son expression spontanée dans l'espace pour la première fois. Avec une augmentation artistique et un souci de lever par cela toutes les gênes liées au regard de l'autre, nous avons planifié un sprint de production de deux semaines du 5 au 19 juin 2018. De café en café et avec une gestion itérative des priorités chaque jour, ainsi que la nécessité d'obtenir systématiquement une version incomplète mais terminée - i.e. une application démontrable - nous avons pu dépasser les objectifs initiaux, qui étaient :

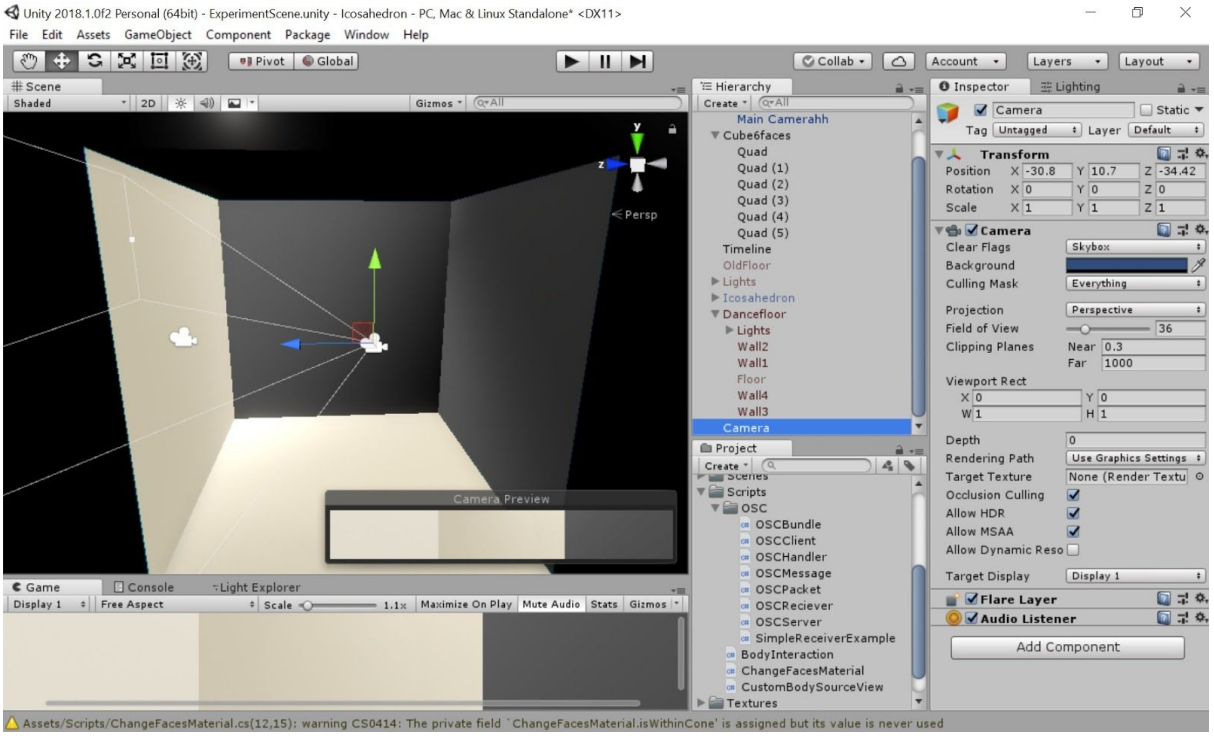
- Définir un univers graphique simple
- Récolter les mouvements du corps avec une Kinect
- Créer un icosaèdre en 3D
- Positionner des particules sur le volume de l'icosaèdre et apprentissage utilisation système de particules KAPPS
- Mettre en place l'interaction avec l'icosaèdre par des sources sur les articulations du corps
- Faire déplacer l'icosaèdre avec le corps en calculant le point moyen des mesures de la Kinect
- Tester le dispositif expérimental sans VR en projection
- Développer avec les outils de VR
- (En fonction de l'avancée, travailler sur l'agrandissement de l'icosaèdre en fonction de la taille de la personne)
- Capturer des images et vidéos pour avoir des premiers contenus
- Fêter ça

La seule exception étant le développement de l'expérience en réalité virtuelle. Le résultat actuel propose une vision à la troisième personne de son corps et sa kinésphère, il s'agira dans un second sprint d'ajouter entre autres une expérience à la première personne : à l'intérieur de son propre icosaèdre.

Réalisation

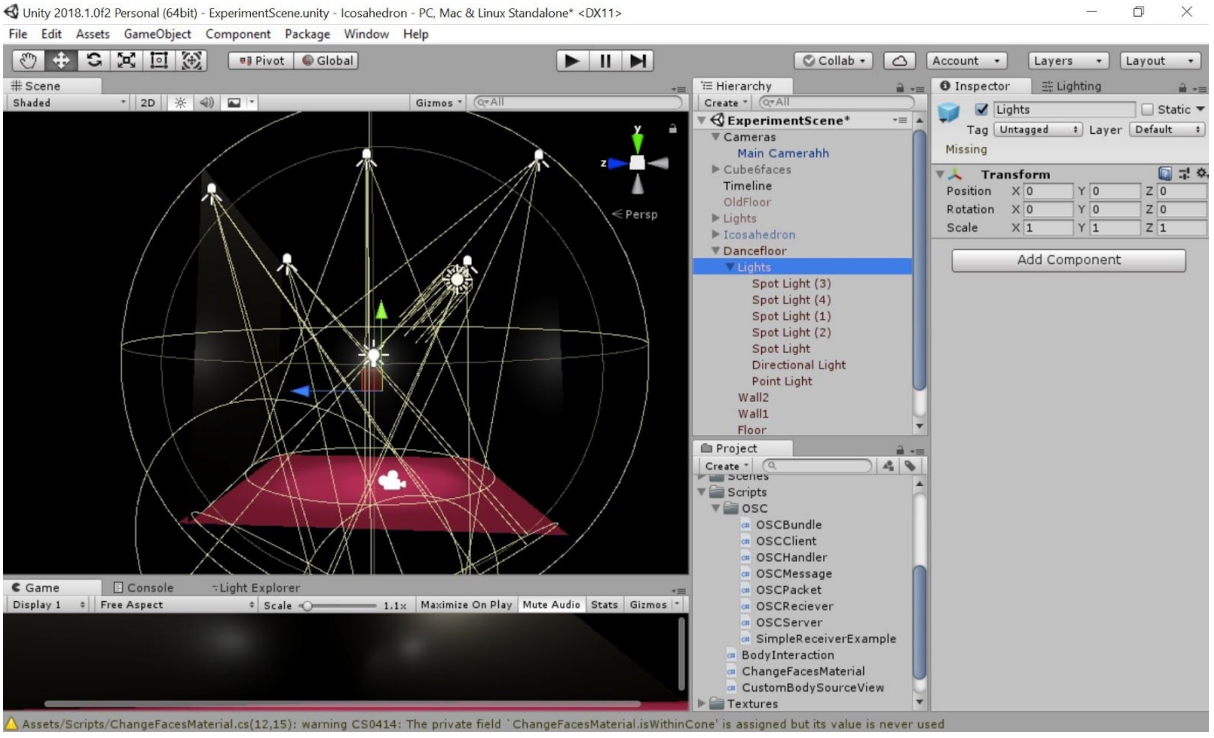
1. Création de formes simples

Think big, start small. L'icosaèdre est le solide de platon à vingt faces, pour commencer nous avons décidé de construire un cube interactif. Sous Unity, il a fallu simplement créer six *quads*, des surfaces dont une seule face est visible au moment du rendu graphique. En orientant ces faces vers l'intérieur, nous avons pu tester quelques scripts écrits en C# pour détecter l'orientation de la caméra en rapport à la position des faces. Une première expérimentation a été de changer la texture d'une face quand l'utilisateur a cette face dans son champ de vision. On commence ainsi à rendre indépendantes les interactions sur les directions.



2. Création de l'environnement graphique simple

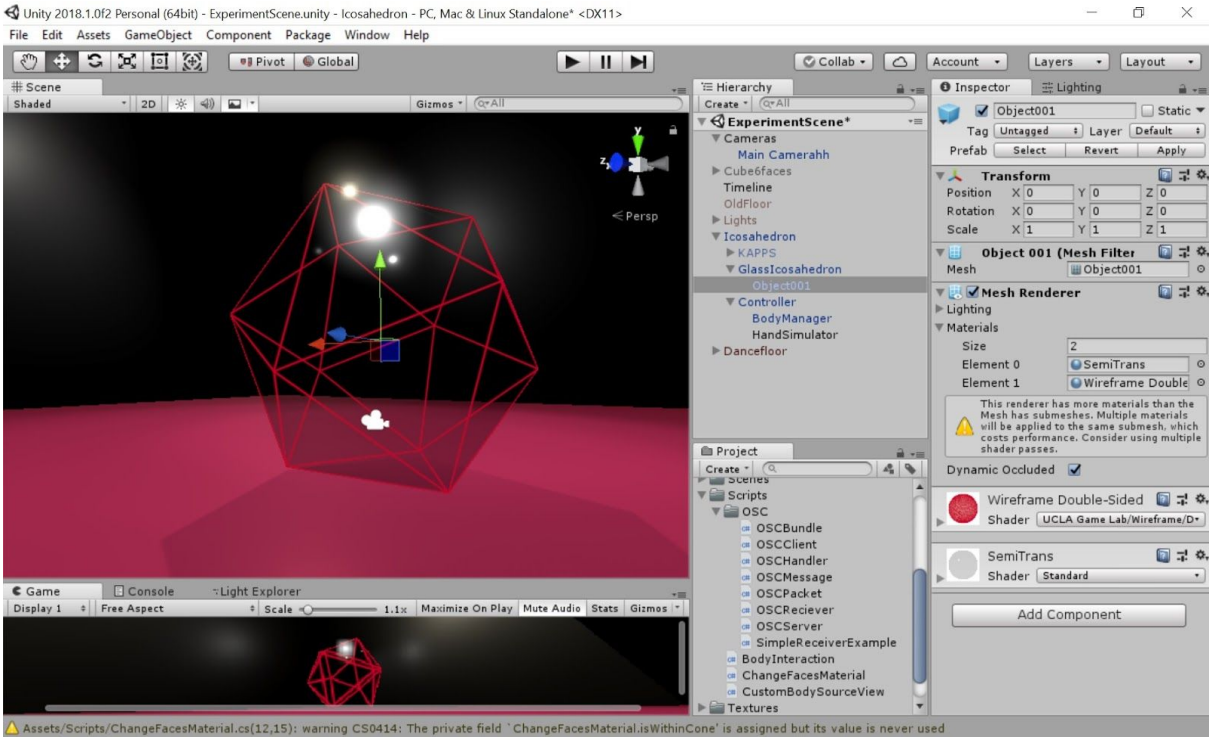
Afin de reproduire une piste de danse dans notre univers, nous avons commencé à construire une scène Unity composée de murs sombres, de sols avec une texture simple et des lumières. Reprenant les plans de feux d'une mise en scène, nous avons choisi d'illuminer la scène avec cinq projecteurs de type *Spot*, tous orientés vers le point central du sol.



3. Icosaèdre

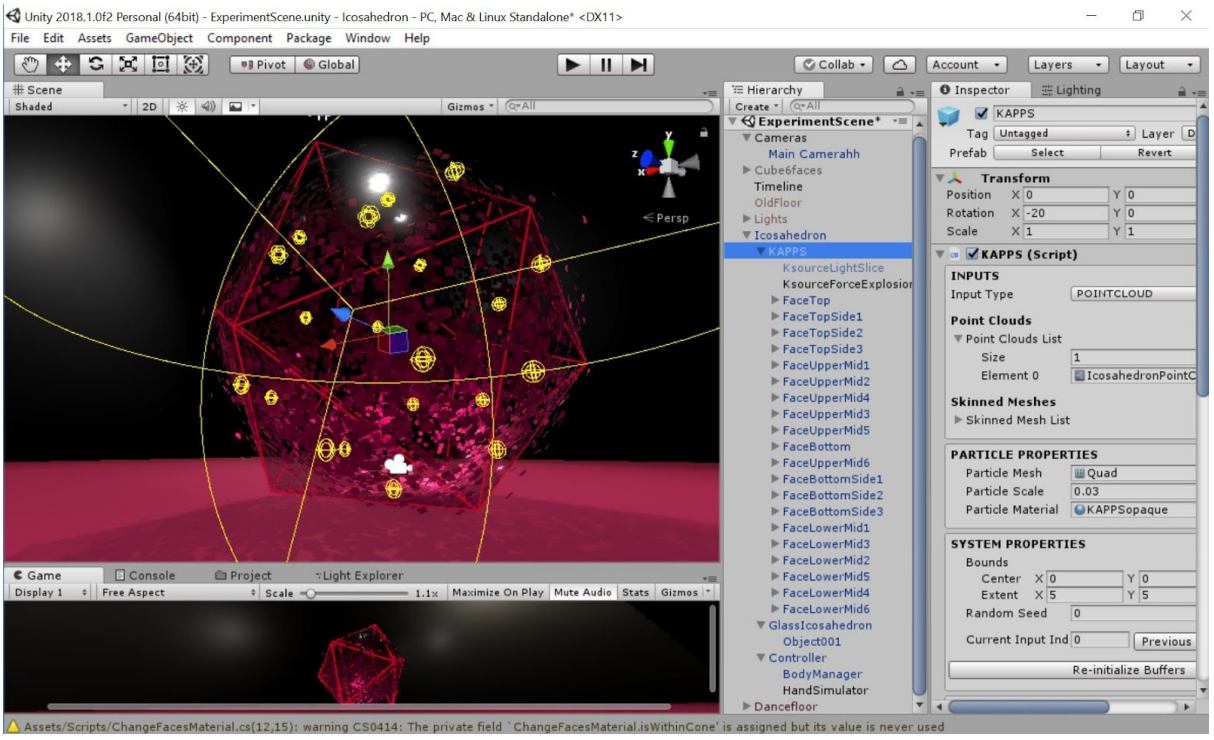
La création de faces interactives indépendantes permet d'imaginer des effets d'éclatement de l'icosaèdre intéressants, ou de pouvoir par exemple repousser une face que l'on touche. En revanche, deux arguments nous ont fait choisir un format simple, le *.obj*, pour l'icosaèdre : d'une part cela nous évite de construire toutes les faces à la main ou par le code, d'autre part cela nous permet a posteriori d'utiliser une fonctionnalité du système de particules KAPPS (par Théoriz, sur la base du travail de Keijiro Takahashi) qui génère des particules sur un unique objet.

Dans un sprint ultérieur, il sera intéressant de coupler les deux approches en additif sur le même volume.



4. Implémentation du système de particules

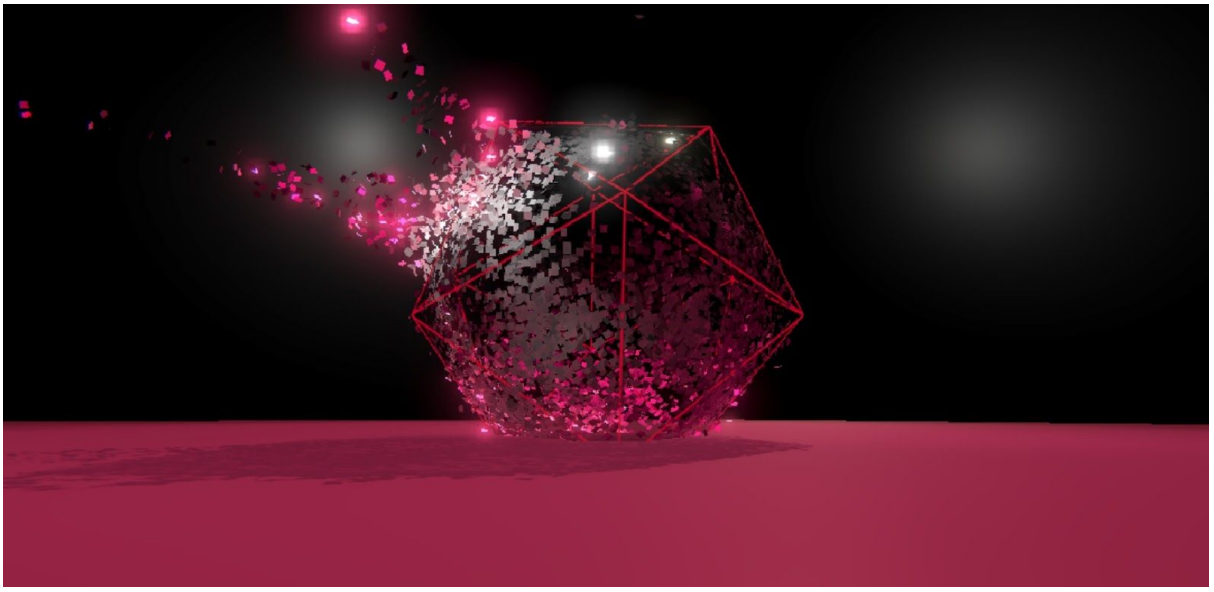
Le système KAPPS permet de positionner des particules sur une surface pour ensuite les déplacer selon des lois physiques. Ayant contribué au module sur certaines forces, nous avons pu utiliser l'une d'entre elles pour repousser les particules de la surface au contact du corps. Il s'agit de créer un *GameObject* dans Unity et de lui appliquer un certain nombre de scripts C# pour qu'il devienne porteur de KAPPS. En créant un *PointCloud* (un ensemble de points de départ pour les particules) à partir d'un mesh, on peut l'appliquer à l'objet KAPPS. On peut aussi choisir le mesh de rendu de chaque particule, c'est-à-dire que chaque particule peut être affichée comme une sphère, un cube, etc. Nous avons choisi le rendu en *quad* simple car c'est le moins gourmand en performances. En contrepartie nous avons pu augmenter le nombre de particules, dans la limite de notre matériel graphique.



5. Ajout de sources

Il est possible d'ajouter des *sources*, qui sont les points d'ancrage de champs de forces physiques comme l'attraction ou le champ magnétique. Une source peut être également une source lumineuse qui va éclairer les particules à proximité ou sur un plan défini passant par la source. Nous avons utilisé les deux types de sources dans notre application finale.

Il a fallu créer et positionner les 20 sources différentes dans l'espace, sur le point central de chaque face de l'icosaèdre. La force de répulsion de type *blackhole* utilisée pour les effets a aussi nécessité l'orientation à la main des vingt axes principaux, de façon orthogonale à chaque face.



6. Kinect

Au cours d'une journée de tests au LABLAB rendue possible par Theoriz, nous avons pu accéder à une Kinect et mettre en application tout ce que nous avions préparé pour l'interaction. L'interfaçage a été plus rapide que prévu, si bien que l'on a pu développer les effets graphiques et l'utilisabilité du système.

Pour faire fonctionner une Kinect classique en USB, il faut un adaptateur spécial qui n'est pas commercialisé mais qui peut se fabriquer avec un poste à souder et quelques manips. Le téléchargement de la SDK Kinect for Windows 2.0 donne accès à des batteries de test qui permettent de vérifier le bon fonctionnement de la Kinect, ainsi qu'à Kinect Studio, une interface qui affiche ce que voit la Kinect.

Côté Unity, le package UnityPro 2.0.1410 présente une scène simple avec affichage des membres du corps par des lignes et des articulations par des cubes. C'est en adaptant le script C# de base de cette scène que l'on peut créer notre application interactive. ll a fallu par exemple imposer une translation à tous les objets de base car le corps était dessiné tel que le buste se trouvait au niveau du sol… pas très pratique.

Le gros du travail a été de choisir quelle articulation du corps allait déclencher quel effet. En programmant un script qualifié de *contrôleur* qui fait le lien entre la détection du mouvement et l'activation des sources du système de particules (et la transmission de certaines variables dites *publiques* entre les classes C#), on a pu ainsi jouer avec un ensemble de paramètres visuels.

```
6
7   public class BodyInteraction : MonoBehaviour {
8
9       public CustomBodySourceView bodySourceView;
10      public GameObject facesParent;
11      public GameObject icosahedron;// Object which will be translated
12      public GameObject simulatedHand;
13      public bool simulationMode;
14      public bool lightMode;
15      public bool blackholeMode;
16      public bool explosionMode;
17
18      private List<GameObject> faces;
19      private List<GameObject> facesKlightSources;
20      private List<GameObject> facesKforceSources;
21      private GameObject icosahedronExplosionSource;
22      private Vector3 rH;
23      private Vector3 lH;
24      private Vector3 rF;
25      private Vector3 lF;
26      private CustomBodySourceView body;
27
28      void Start () {
29          // Get icosahedron faces children
30          faces = new List<GameObject>();
31          facesKlightSources = new List<GameObject>();
32          facesKforceSources = new List<GameObject>();
33
34          foreach (Transform child in facesParent.transform) {
35              if (child.tag == "face") {
36                  // Register face Game Object
37                  faces.Add(child.gameObject);
38
39                  // Get faces' KAPPS light source and force source Game Objects
40                  foreach (Transform grandChild in child.gameObject.transform) {
41                      if (grandChild.tag == "k-light") {
```

7. Tests et ajustements sur les effets

Trois effets ont été créés dans la première version de Kinexperience. Le premier est un effet de lumière tel que quand le danseur passe ses mains et ses pieds devant une face, on active une source lumineuse du système de particules sur cette face. On obtient ainsi un effet de trace simple. Deux options ont été testées sur la relation entre la distance entre l'articulation (main/pied) et le centre de la face : soit on active à 100% la source quand l'articulation se rapproche en deçà d'un certain seuil de proximité, soit on calcule à chaque instant l'intensité lumineuse de chaque face selon une loi exponentielle décroissante en la distance. Nous avons retenu la dernière option, pour laquelle il suffit

ensuite de calibrer la distance critique (constante de la fonction exponentielle) pour ne pas tout illuminer à la fois, ou ne rien illuminer.

Le second effet va déclencher une expulsion brutale des particules vers l'extérieur de l'icosaèdre de façon orthogonale. Il s'agit là encore d'activer l'intensité d'une force - sur un modèle inspiré des trous noirs - selon une fonction de la distance. Plus l'articulation est proche du centre de la face, plus l'intensité de la source de force et la vitesse d'expulsion sont grandes.

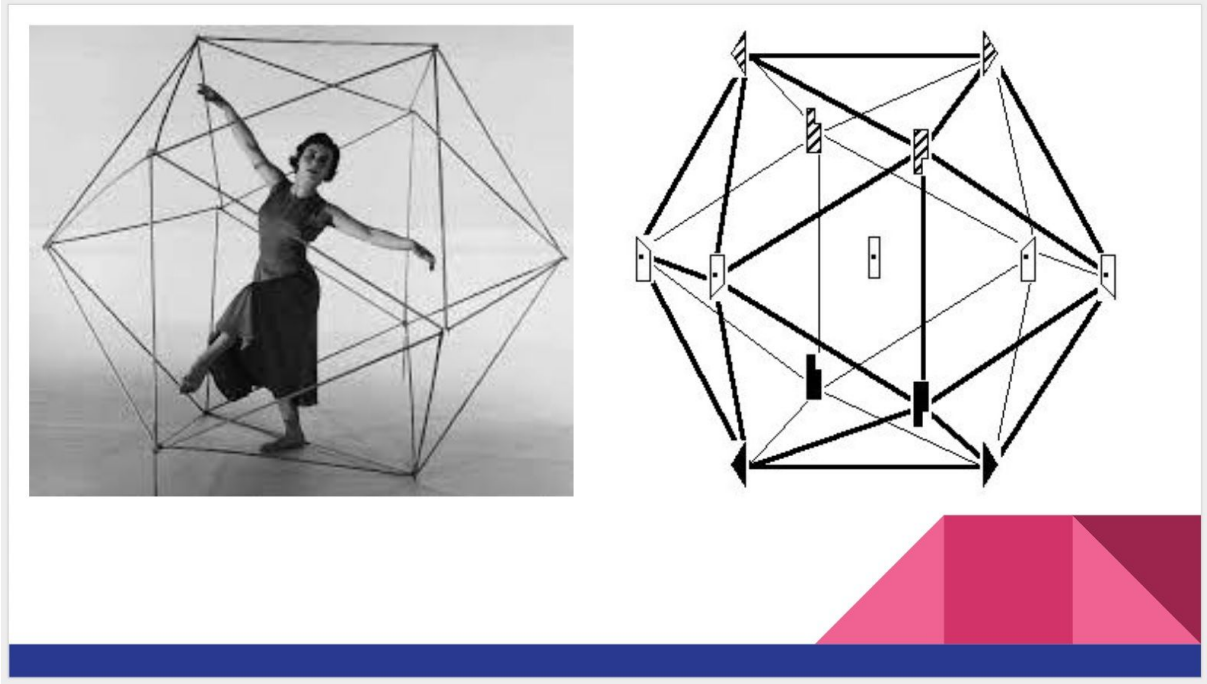
Le dernier effet s'intéresse uniquement à la distance absolue entre les mains. En positionnant un attracteur gravitationnel au centre de l'icosaèdre, on va pouvoir soit appliquer une force de répulsion radiale à toutes les particules, soit l'équivalent avec une intensité négative ; c'est-à-dire une attraction vers le point central. L'activation du changement se fait lorsque la distance entre les mains passe un certain seuil, qui correspond à peu près à la largeur des épaules.

Enfin, nous avons pu détecter quelques points d'amélioration à court terme. La kinésphère se déplace en même temps que le corps de la personne, et il arrive que certaines faces soient inatteignables car nos membres sont trop courts ! Le problème reste le suivant : si l'on réduit l'icosaèdre pour pouvoir toucher toutes les faces, on a la tête qui dépasse…
Pour la suite on pourra réfléchir à des solutions pour prolonger virtuellement les membres, s'intéressant ainsi à l'orientation plutôt que la position des bras.
Une autre amélioration que nous aimerions apporter est la sensibilité à la rapidité du mouvement. Dans cette version les effets sont déclenchés uniquement par la position des articulations. Que l'on approche doucement d'une face ou que l'on vienne la percuter avec force, l'effet visuel est le même. Prendre en compte cette vitesse pour différencier les effets est un chantier à explorer, soit par la comparaison successive des positions (la dérivée première de la position donne la vitesse), soit par l'utilisation d'autres moyens de captation électroniques comme des accéléromètres.

```
// Compute closest point (minimal distance)
float d = Math.Min(d1, Math.Min(d2, Math.Min(d3, d4)));

/*** LIGHT ***/
// Exponential law for light (multiple light sources lit)

KAPPSSource lightSource = (KAPPSSource) facesKlightSources[i].GetComponent(typeof(KAPPSSource));
if (lightMode) {
    lightSource.intensity = 15.0f * ((float) (Math.Exp((double) (-d / 1.5f))));
} else {
    lightSource.intensity = 0.0f;
}

/*** BLACKHOLE FORCE ***/
// Threshold law for blackhole forces (active or inactive)

KAPPSSource forceSource = (KAPPSSource) facesKforceSources[i].GetComponent(typeof(KAPPSSource));
if (blackholeMode) {
    //float maxIntensityForForce = 60.0f;
    //float activationDistance = 3.0f;

    forceSource.vortexDownstreamIntensity = 120.0f * ((float) (Math.Exp((double) (-d / 1.5f))));
} else {
    forceSource.vortexDownstreamIntensity = 0.0f;
}

/*** EXPLOSION FORCE ***/
// Mapped to distance between hands
KAPPSSource explosionSource = (KAPPSSource) icosahedronExplosionSource.GetComponent(typeof(KAPPSSource));
if (explosionMode) {
    Vector3 fromRightToLeft = lH - rH;
    float eDist = fromRightToLeft.magnitude;
    float distForExplosionZero = 5.0f;

    // Forbid inwards attraction
    explosionSource.intensity = Math.Min(0.0f, -5.0f * (eDist - distForExplosionZero));
```

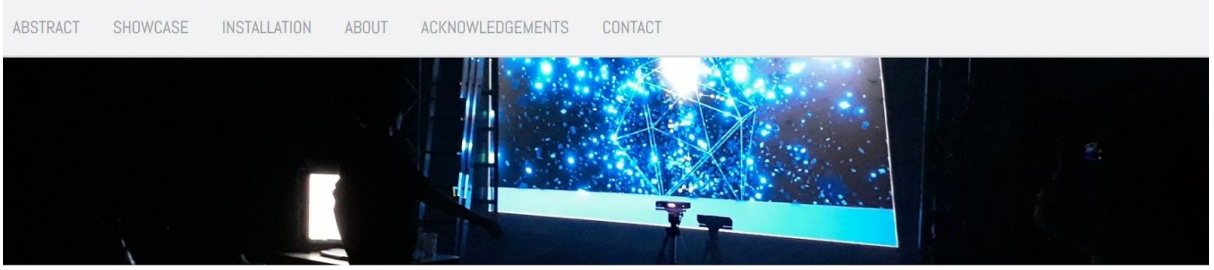
8. Présentation du projet et théorie

A la fin de la journée de développement au LABLAB, nous avons présenté le projet à l'équipe sur place. Nous avons rédigé quelques slides de présentation pour poser le cadre théorique de Rudolf Laban et présenté nos objectifs. Les danseurs de circonstance nous ont donné leur retour sur l'interaction et les rendus graphiques, faisant avancer nos réflexions et mûrir nos idées pour la suite.



9. Captures et site web

Les photos prises lors des expérimentations et des captures d'écrans pendant la conception nous ont permis de construire une page web dédiée au projet Kinexperience. Avec une charte graphique simple, on y présente les résultats du premier sprint et nos remerciements. Cette page sera mise à jour au cours de la vie du projet !



Kinexperience

Abstract

Modern choreographic composition is strongly influenced by dance theorist Rudolf Laban. He explored the field of Choreutics with the concept of Kinesphere, an icosahedron representing the natural boundaries of our bodily space in which motion happens.

Kinexperience immerses the public right into Laban's dance framework by wrapping a full human-sized virtual icosahedron around the person's body. The icosahedron reacts to the motion of hands and feet triggering visual effects like trace lighting or particles explosion, giving the dancer feedback and insight into its own spontaneous practice.

Galerie d'images

Ce sont toutes les images (environ 10) présentes sur le site web actuel. C'est possible de les récupérer une par une, n'hésitez pas à nous demander un dossier compressé avec les images si nécessaire.

Mots clés

kinésphère / numérique / danse / corps / motion capture

Crédits / Collaborateurs

Théoriz Studio
Melting Space
LaboNRV
Subsistances
Jeanf
Benoit Arbelot
Alex Bourgeois

Auteurs
Artiste chorégraphe : Ariane Cassimiro
Artiste numérique : Alex Andrix

Matériaux / machines / logiciels employés / technologies
3D creation software -> Unity (version 2018.0.2f)
Body motion detection -> Kinect v2
Particle system -> KAPPS (Theoriz Studio)
Dancefloor experimentation -> LABLAB
VR headset -> HTC Vive (project version 2)

Fichiers et liens associés
alexandrix.com/kinexperience/