TecnoCampus
Escola Superior Politècnica

Centre adscrit a la

upf. Universitat Pompeu Fabra Barcelona

Grau en Mitjans Audiovisuals

Dreams: espectacle de *videomapping* i dansa

Annexos

BERTA FERNÁNDEZ BECERRA
TUTOR/A: MARCO ANTONIO RODRÍGUEZ FERNÁNDEZ
CURS 2022-23

TecnoCampus

Índex

Índex de figures

1. Més investigacions del son

Aquest apartat mostra més investigacions que s'han realitzat sobre el somni. En concret, TED Talks de psicòlegs, neuròlegs i gent amb renom que ha fet diferents estudis sobre la son.

1.1. *TED Talks* sobre el son i el somni

El canal TED, no només publica xerrades, sinó també disposa de vídeos informatius, com *The surprising health benefits of dreaming*, de *Sleeping with Science* (TED, 2021). Es tracta d'un vídeo d'una durada aproximada de dos minuts que parla dels beneficis de somiar. En concret, menciona que la creativitat dels somnis ens anima a resoldre els nostres propis trencaclosques, que resulta en la resolució dels problemes del següent dia. Aquest mateix concepte també s'aplica en la part emocional, on el fet de somniar ajuda a descarregar la motxilla de problemes que tenim a sobre, i a sentir-nos millor; el que ell anomena teràpia nocturna. Se'n recalca la importància de fixar-nos en què hem somiat. Explica que somiar sobre un fet que ens afecta pot ajudar-nos a resoldre'l, però que somiem amb un fet, però no amb un fet en si, cal col·locar les peces del trencaclosques i sentir-nos millor (TED, 2021).

Una altra ponent és Amy Adkins, on al seu TED-Ed anomenat *Why do we dream?* (TED-Ed, 2015) on exposa que els somnis ajuden a la persona a sentir-se realitzada, i que estan plens de simbolismes. Robert Sickgold a la seva ponència *Why do we dream* (TEDx, 2021), aporta un altre punt de vista: el cervell té un mecanisme que identifica, explora i avalua associacions inesperades. És a dir, que crea un món on vol que reaccionis en la història que no és una memòria del cervell, sinó que és un món fora de les associacions establertes en el món real.

La TEDx Talks de Chongtul Rinpoche amb el nom de *Lucid dreams as bridge between realities* (TEDx Talks, 2014), destaca que el somni prové de l'experiència de la nostra vida anterior o futura, i diferència entre tipus de somnis. El primer és el *karmic dream,* que és el somni de cada nit. El segon és l'anomenat *wisdom dream*, que és quan somiem durant el dia i quan somiem desperts. Finalment, el *lucid dream*, el somni lúcid que per ell és el somni sobre la vida on sentim que l'hem viscut o fet en la nostra vida passada, però no recordem quan. Segons Rinpoche, aquesta és la millor forma de controlar el cervell i és una medicina on cal mirar dins de la nostra pròpia experiència.

Sobre els somnis lúcids també en parla Tim Post a la seva ponència amb el nom *Lucid Dreams* (Tedx Talks, 2013), on descriu aquest tipus de somnis com que se sent com si "haguessis estat absorbit per la imaginació", i apunta que és la forma de millorar el desenvolupament psicològic i tenir facilitats en trobar solucions creatives als problemes.

A diferència dels anteriors ponents, Matthew Walker aporta un nou punt de vista amb *Sleep is your superpower* (TED, 2019), explicant els beneficis o perjudicis de dormir aportant dades científiques i gràfiques que fan reflexionar a qualsevol oient. En la següent imatge, es mostren els efectes de dormir 6 hores durant una setmana.

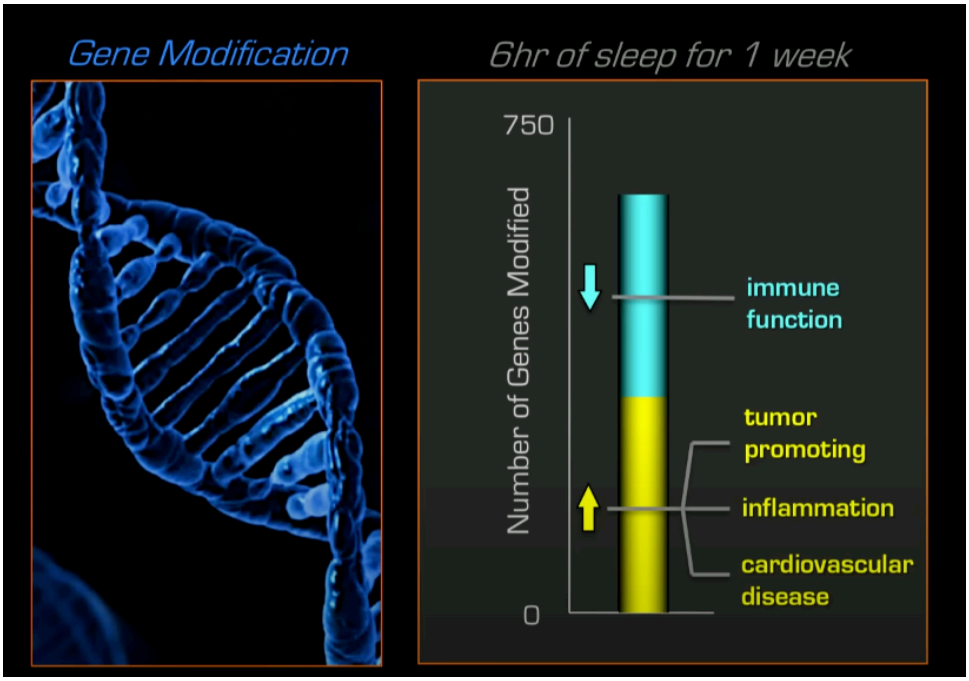

Figura 1.. Prejudicis de no dormir del TED de Matthew Walker. Font: TED, 2019.

Walker afirma que dormir 4 hores o menys o "dormir poc", augmenta aproximadament un 70% que es desenvolupi un càncer. Dormir 6 hores en comptes de dormir-ne 8, implica tenir gens de l'ADN modificat, que els gens encarregats de la funció immunitària disminueixin, i que augmentin les possibilitats de patir tumors, inflamacions i problemes cardiovasculars (TED, 2019). A més a més, l'impacte de dormir una hora més o una hora menys l'exemplifica amb el canvi d'hora estacional. Quan es canvia l'hora, en funció de si és estiu o hivern, i de si s'augmenta o disminueix 1 hora del rellotge, augmenten un 25% els atacs de cor el següent dia, o disminueixen un 25% els atacs (TED, 2019).

Per poder combatre els símptomes d'insomni, Walker proposa evitar el cafè i l'alcohol, no fer migdiades si ens costa dormir a la nit, tenir una regularitat en l'horari de dormir i de llevar-se, i apunta també que és més fàcil adormir-se amb una habitació més aviat freda que calenta, perquè pel cos és més fàcil entrar en el cicle del son en estat més fred (TED, 2019). D'altra banda, també emfatitza en el fet que, si no t'adorms quan estàs al llit, has de sortir del llit i anar a fer quelcom diferent a un altre lloc, perquè si no el cervell està associant que al llit s'està despert. El cervell ha d'entendre que el llit és el lloc de dormir, i per això recomana no "estar donant voltes" als llençols. Un exemple per reflexionar sobre aquest parer, és que "si no seus a la taula per esperar a tenir gana, per què esperes al llit per adormir-te" (TED, 2019).

2. Wayne McGregor, exemple de fusió de dansa i tecnologia

Recentment, el món de la dansa (majoritàriament companyies de dansa contemporània) han començat a treballar i utilitzar el *mapping* reactiu, on els ballarins interactuen amb el contingut (Jewel, 2015). Wayne McGregor és un referent en aquest àmbit on es fusiona la dansa i la tecnologia per crear nous espectacles.

Ell és un coreògraf i director britànic amb nombrosos premis i conegut internacionalment per les seves actuacions amb innovació que han revolucionat l'era moderna. Els seus treballs fusionen les formes artístiques, la tecnologia i les ciències, desencadenant en un estil visual distintiu que explota les possibilitats del cos i articulacions (McGregor, s.d).

Va néixer a la dècada del 1970, on John Travolta era un ídol per les seves actuacions a *Grease* i *Fiebre del sábado Noche*. Per McGregor, l'actor fou el clar model masculí perquè ell comencés a ballar. Els seus pares li feren costat i donaren suport, i pogué anar a un estudi de dansa local amb un professor progressista que el deixà que fos ell mateix i que inventés els seus propis balls. Allà, creà una sala de ball per donar classes de ball i de balls llatinoamericans. El fet de poder expressar la seva pròpia veu, l'impulsà a convertir-se en coreògraf. (TED, 2012).

L'any 1992, fundà *Random Dance*, que posteriorment s'anomenà *Studio Wayne McGregor*. El març de 2017 l'estudi es traslladà a on nou espai al *Here East*, al *Queen Elizabeth Olympic Park*, on actualment ha dut a terme més de 30 treballs per la companyia, i continua investigant i creant peces noves (McGregor, s.d).

Des de 2006, McGregor és un Coreògraf Resident a *The Royal Ballet*, el primer coreògraf de dansa contemporània convidat, on ha realitzat més de vint treballs com *Chroma* (2006) o *Woolf Works* (2015). A més a més, no només té una alta demanda al teatre, sinó que també per l'òpera, el cinema, els vídeos musicals, la moda i la televisió (McGregor, s.d).

Durant el 2012, realitzà una *TED Talk* per explicar el seu pensament i la seva percepció de la dansa. En aquesta exposició, exposà que la seva motivació és trobar una forma de comunicar idees a l'audiència a través del cos, ajudar-los a pensar diferent. Per ell,

"coreografiar és un procés col·laboratiu amb altres persones" i és important el "pensament corpori". Per entendre una mica més com fa els seus treballs, a partir d'uns exemples, ensenyà els tres tipus de formes de treballar.

El primer l'anomenà "cos amb cos", on els ballarins tenen la memòria per realitzar els passos a partir de la informació que dona McGregor. Per exemplificar-ho, els va fer imaginar la lletra "T", i els digué que podien caminar al voltant d'ella, descriure-la amb les mans el cap els braços... A partir dels moviments, captaren aspectes del moviment i ho transformaren en una frase. El segon tipus fou un duet entre els dos ballarins on s'havia de pensar en objectes arquitectònics, línies, formes..., i crear frases que després s'organitzarien. El darrer mètode fou que ells mateixos decidiren la coreografia (TED, 2012).

A partir d'aquestes formes de treballar, només cal afegir la tecnologia per acabar els seus projectes que cada dia guanyen més renom en el món artístic.

3. Documents de producció

3.1. Esquemes de planta

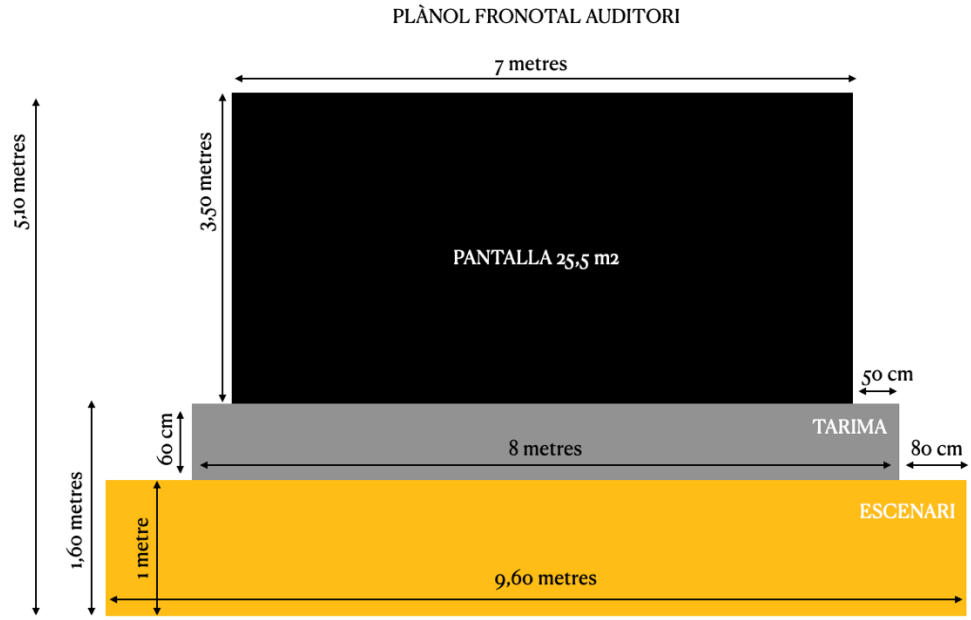

Figura 3.1.. Plànol frontal auditori. Font: elaboració pròpia.

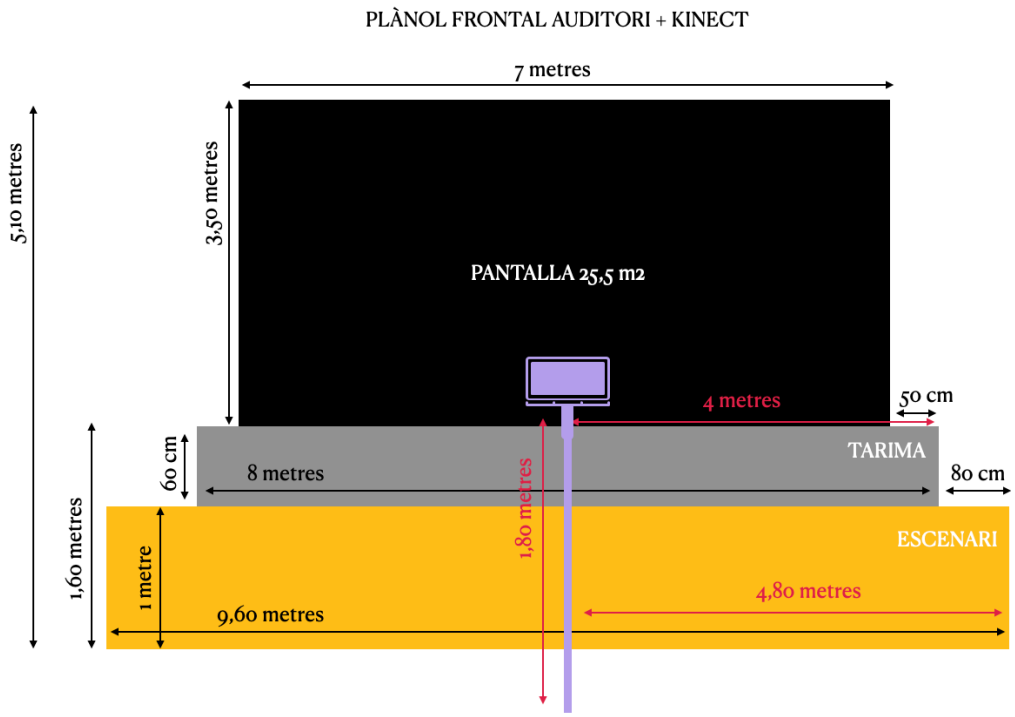

Figura 3.2.. Plànol frontal auditori + Kinect. Font: elaboració pròpia.

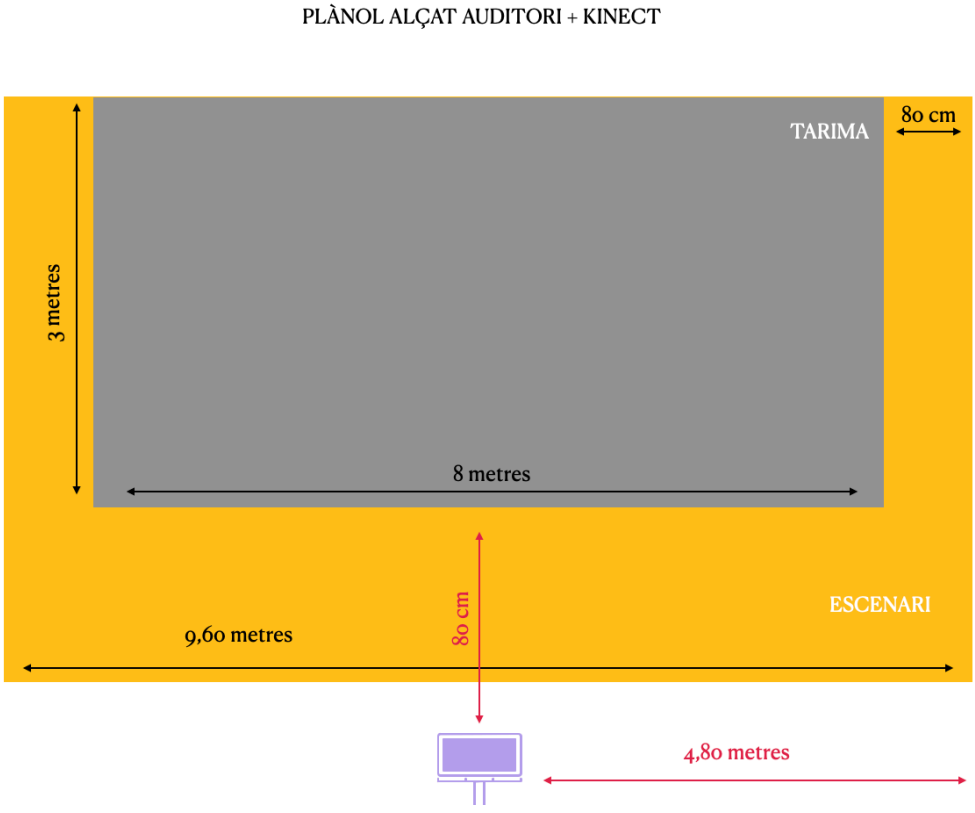

Figura 3.3.. Plànol alçat auditori + kinect. Font: elaboració pròpia.

3.2. Esquema connexions

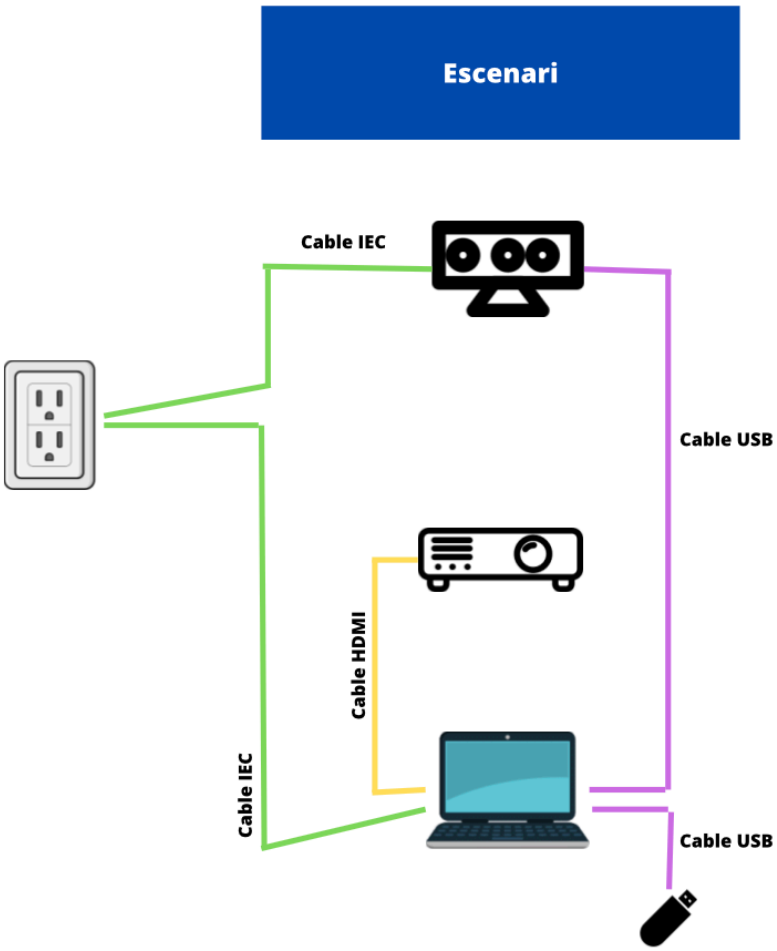

Figura 3.4.. Esquema connexions. Font: elaboració pròpia.

3.3. Resum espectacle

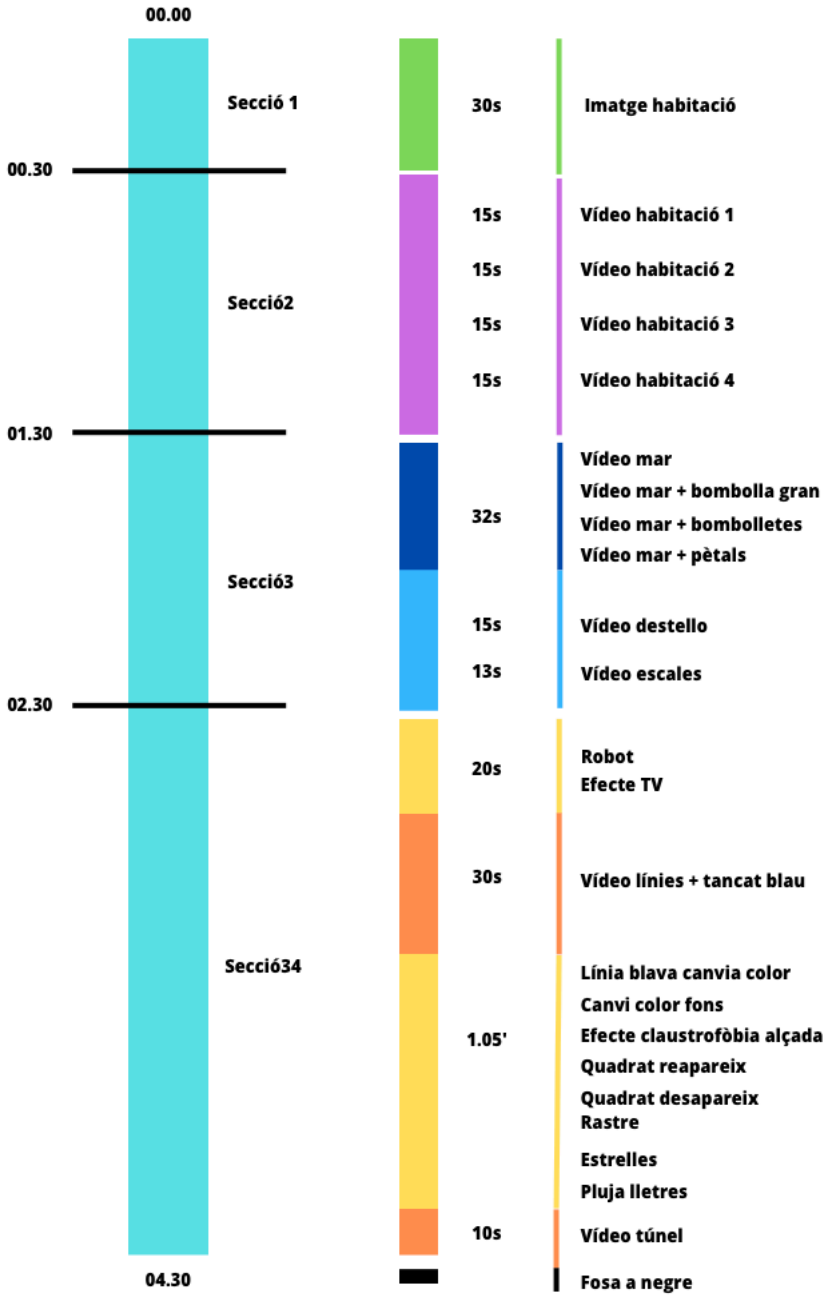

Figura 3.5.. *Timeline* resum espectacle. Font: elaboració pròpia.

3.4. Esbós pentinat i vestuari

En relació amb el pentinat de la ballarina, s'han dibuixat 3 versions diferents que es poden utilitzar per l'espectacle.

Figura 3.6.. Esbós pentinats. Font: elaboració pròpia.

El vestuari anirà en funció de la quantitat de diners que es vulgui invertir. A continuació es mostren 3 opcions diferents. Qualsevol pot tenir bons resultats però un mallot de màniga llarga és el que a l'autora li sembla millor opció.

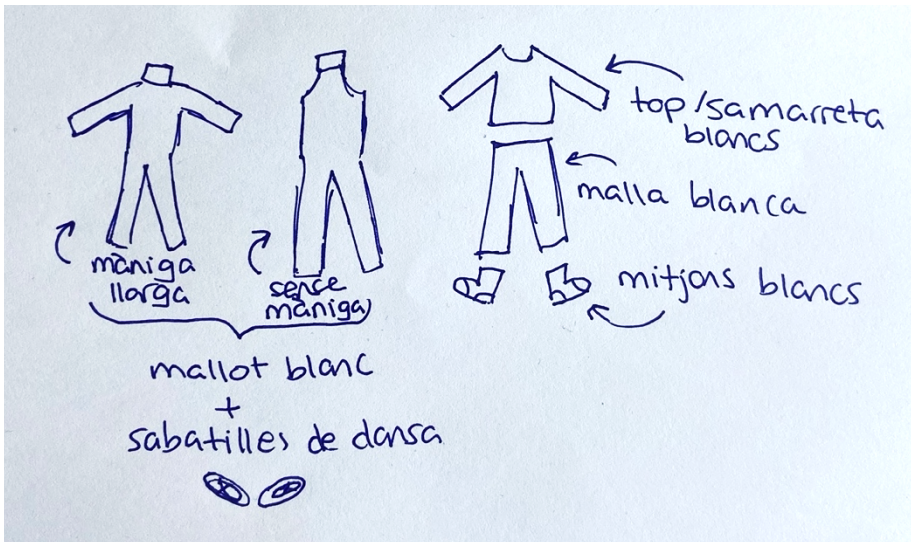

Figura 3.7.. Esbós vestuaris. Font: elaboració pròpia.

3.5. *Riders* tècnics

Exemple de *Rider* tècnic de la prova de projecció

Fitxa tècnica		
Descripció	Proves de projecció del TFG "Dreams: espectacle de *videomapping* i dansa"	
Data	12 juny	
Localització	Aula Xnergic de la universitat TecnoCampus Mataró	
Horari	9:30h a 13:30h	

Logística		
Equip	Directora (Berta Fernández)	
Horari	9:30-10:30	recollida de material a SERMAT
	10:00-11:00	muntatge
	11:30-12:30	proves de projecció i calibratge
	12:30-13:00	desmuntar
	13:00-13:30	retorn del material a SERMAT

Material	
Càmera	
	IPhone 11
Mapping	
	Sensor Kinect
	Ordinador HP
	Pen Drive
	Projector Sony VPL-DW241

Figura 3.8.. *Rider* tècnic prova projecció. Font: elaboració pròpia.

Exemple de *Rider* tècnic directe

Fitxa tècnica	
Descripció	Proves de projecció del TFG "Dreams: espectacle de *videomapping* i dansa"
Data	28 i 29 juny
Localització	Auditori del TecnoCampus Mataró
Horari	9:30h a 14:30h

Logística 28 juny		
Equip	Directora (Berta Fernández)	
Horari	9:30-10:30	recollida de material a SERMAT
	10:00-11:00	muntatge
	11:30-13:30	proves de projecció i calibratge
	13:30-14:00	desmuntar
	14:00-14:30	retorn del material a SERMAT

Material	
Càmera	
	Canon EOS
	Trípode de càmera
Mapping	
	Sensor Kinect
	Ordinador HP
	Pen Drive
	Projector auditori
	Schuko de 15 metres (4 unitats)
Il·luminació	
	Llums auditori
	Focus de retall (si és possible)
	Carta d'enfoc

Logística 29 juny		
Equip	Directora, tècnic llums, operador de *mapping* i càmera	
Horari	9:30-10:30	recollida de material a SERMAT
	10:00-11:00	muntatge
	11:30-13:30	proves de projecció + directe
	13:30-14:00	desmuntar
	14:00-14:30	retorn del material a SERMAT

Citacions d'equip 29 juny		
Equip	Directora, tècnic llums, operador de *mapping* i càmera	
Horari	9:00	Directora i operador de *mapping*
	9:30	Tècnic de llums
	10:30	Càmera

Figura 3.9.. *Rider* tècnic directe. Font: elaboració pròpia.

4. Prova de projecció

4.1. Seccions

Secció 1

Figura 4.1. Fotogrames secció 1. Font: elaboració pròpia.

Secció 2

<u>Habitació 1</u>

Figura 4.2. Fotograma habitació 1. Font: elaboració pròpia.

Habitació 2

Figura 4.3. Fotograma habitació 2. Font: elaboració pròpia.

Habitació 3

Figura 4.4. Fotogrames habitació 3. Font: elaboració pròpia.

Habitació 4

Figura 4.5. Fotogrames habitació 4. Font: elaboració pròpia.

Secció 3

Vídeo mar

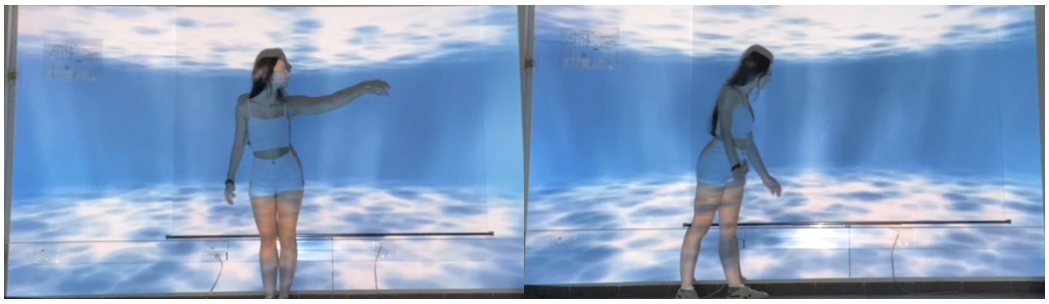

Figura 4.6. Fotogrames vídeo mar. Font: elaboració pròpia.

Map1: vídeo mar + bombolla gran

Figura 4.7. Fotogrames vídeo mar + bombolla gran. Font: elaboració pròpia.

Map2: vídeo mar + bombolletes

Figura 4.8. Fotogrames vídeo mar + bombolletes. Font: elaboració pròpia.

Map3: vídeo mar + pètals

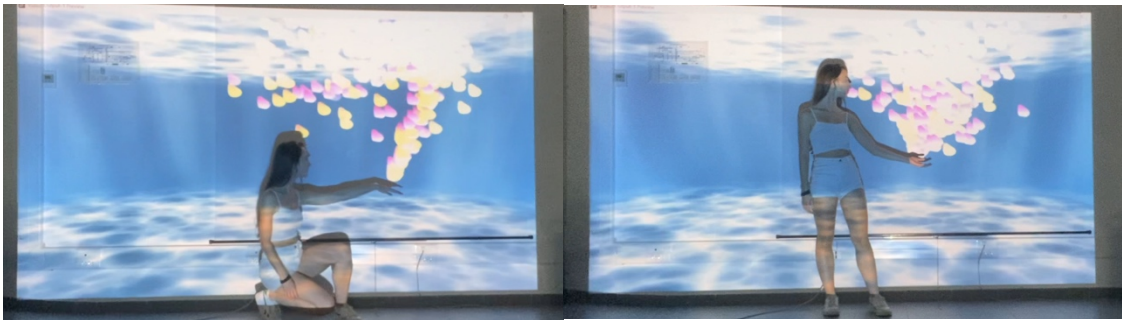

Figura 4.9. Fotogrames vídeo mar + pètals. Font: elaboració pròpia.

Vídeo feix llum

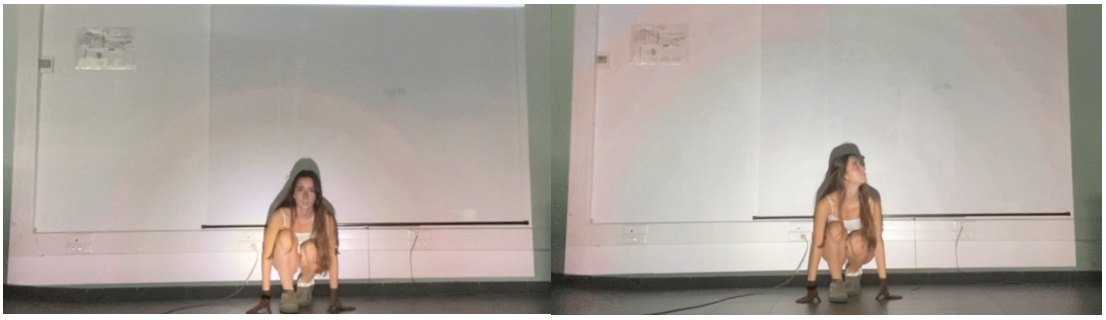

Figura 4.10.. Fotogrames vídeo feix llum Font: elaboració pròpia.

Vídeo escales

Figura 4.11.. Fotogrames vídeo escales. Font: elaboració pròpia.

Secció 4

Mapping 1: Robot

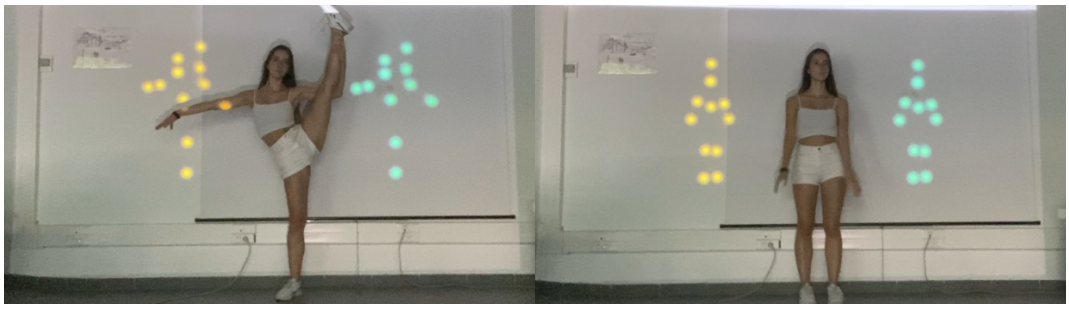

Figura 4.12.. Fotogrames *mapping* robot. Font: elaboració pròpia.

Mapping 2:Efecte TV

Figura 4.13.. Fotogrames *mapping* efecte TV. Font: elaboració pròpia.

Vídeo línies claustrofòbia a blau

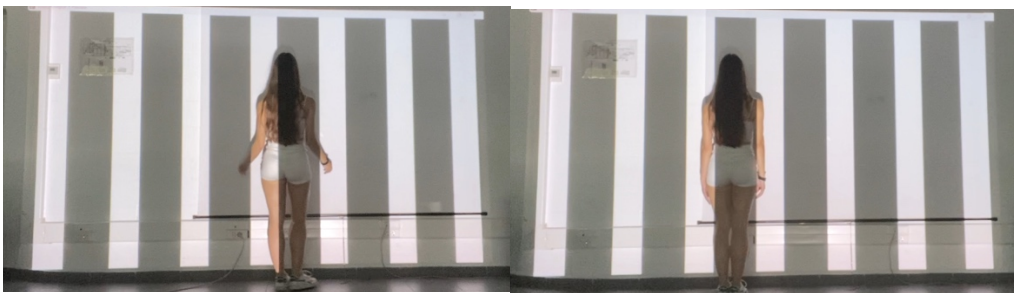

Figura 4.14.. Fotogrames vídeo línies. Font: elaboració pròpia.

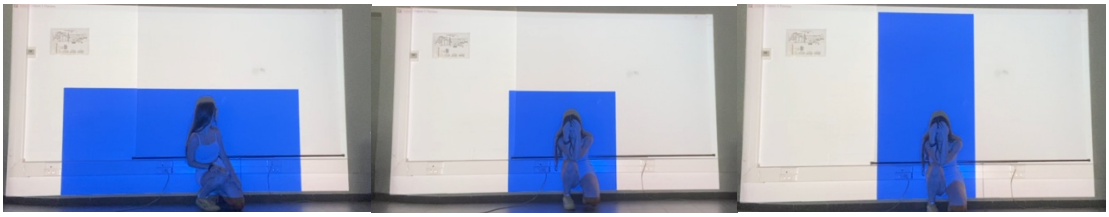

Figura 4.14.. Fotogrames vídeo claustrofòbia. Font: elaboració pròpia.

Mapping 3:Línia blava canvia posició

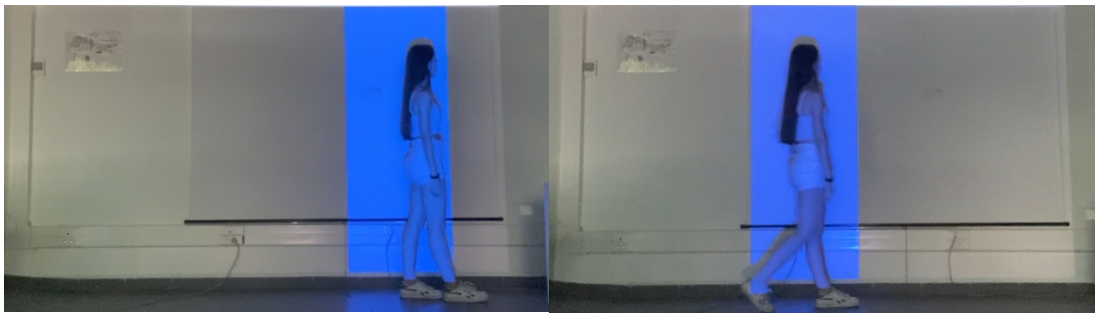

Figura 4.15.. Fotogrames *mapping* línia blava. Font: elaboració pròpia.

Mapping 4: Canvi color fons

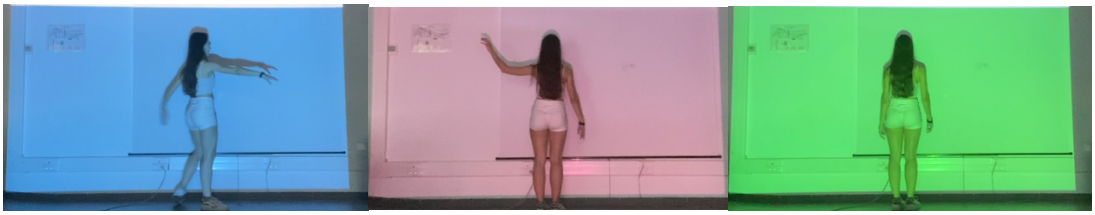

Figura 4.16.. Fotogrames *mapping* canvi color fons. Font: elaboració pròpia.

Mapping 5: Claustrofòbia

Figura 4.17.. Fotograma *mapping* claustrofòbia. Font: elaboració pròpia.

Mapping 6: Quadrat reapareix

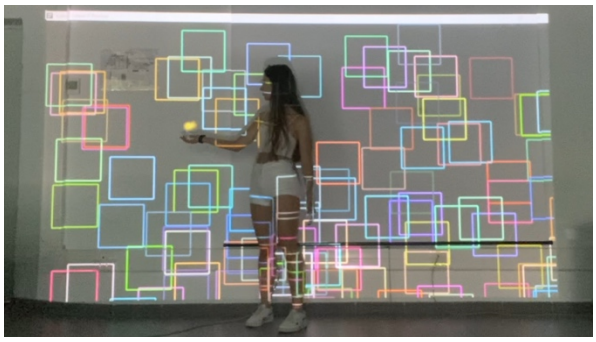

Figura 4.18.. Fotograma *mapping* quadrat reapareix. Font: elaboració pròpia.

Mapping 7: Quadrat desapareix

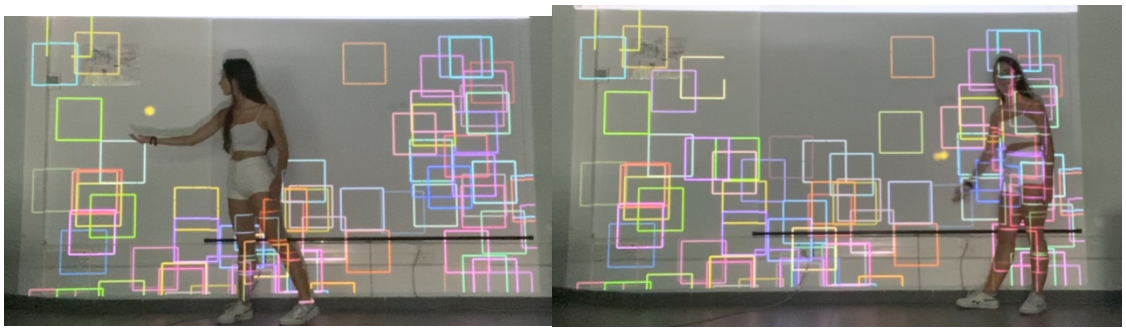

Figura 4.19.. Fotogrames *mapping* quadrat desapareix. Font: elaboració pròpia.

Mapping 8: Rastre

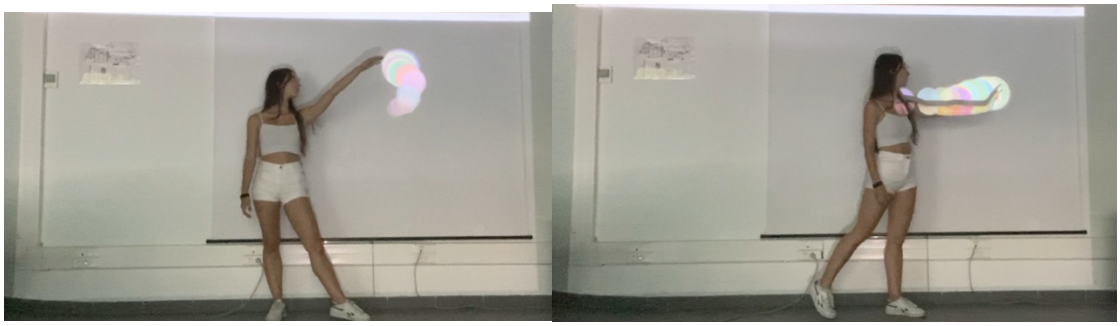

Figura 4.20.. Fotogrames *mapping* rastre. Font: elaboració pròpia.

Mapping 9: Estrelles

Figura 4.21.. Fotograma *mapping* estrelles. Font: elaboració pròpia.

Mapping 10: Pluja de lletres

Figura 4.22.. Fotogrames *mapping* pluja de lletres. Font: elaboració pròpia.

Vídeo túnel

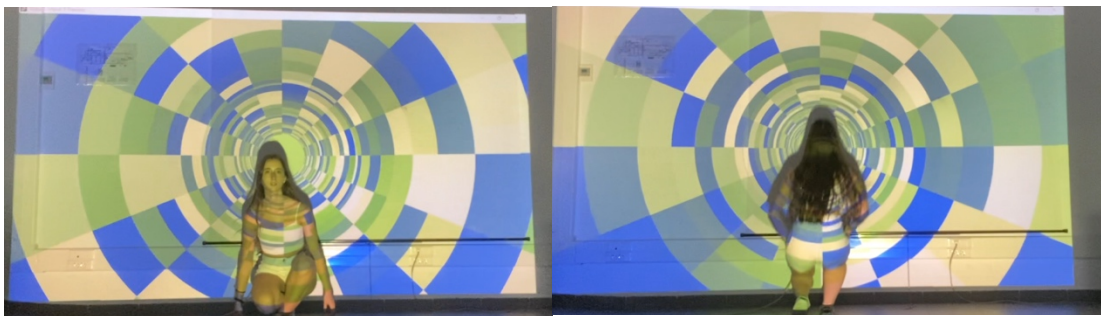

Figura 4.23.. Fotogrames vídeo túnel. Font: elaboració pròpia.

5. Codi de *Processing*

```
//GENERAL

import spout.*;

import kinect4WinSDK.Kinect;

import kinect4WinSDK.SkeletonData;

import processing.sound.*;



Spout spout;

SoundFile audio1, audio2, audio3, audio4, audio5, audio6, audio7, audio8, audio9, audio10;



int estado=0;

Kinect kinect;

ArrayList <SkeletonData> bodies;



//map1

PImage bombolla;

//map2

PImage bombolla1;

int npartTotal = 50000;

int npartPerFrame = -1;
```

```
float speed = 1;

float gravity = -0.005;

float partSize = 15;


int partLifetime;

PVector positions[];

PVector velocities[];

int lifetimes[];


int fcount, lastm;

float frate;

int fint = 3;

//map3

ParticleSystem ps;

//mapping1

PImage rodona;

PImage rodona2;

//mapping2

int temps = 0;

int temps2 = 0;

//mapping3
```

```
float lineaX = 0; // Posició lini

//mapping4

int tiempo;

float u ;

float g ;

float b ;

float a ;

//mapping5


//mapping6

int[] x = new int[200];

int[] y = new int[200];

color[] squareColors = new color[200];

PImage groc;

//mapping7

PImage punt;


//mapping8

ArrayList<Particle2> particles;

//mapping9

ArrayList<Particle3> particles3 = new ArrayList<Particle3>();
```

```
//mapping10

float[] j = new float[60];

float[] k = new float[60];

float[] v = new float[60];

char[] l = new char[60];


void setup() {

  //GENERAL

  size(708, 400, P3D);

  spout = new Spout(this);

  spout.setSenderName("Berta");

  kinect = new Kinect(this);

  smooth();

  bodies = new ArrayList<SkeletonData>();


  //map1

  bombolla=loadImage("bombolla_be.png");

  //map2

  bombolla1 = loadImage("bombolla_be.png");

  partLifetime = npartTotal / npartPerFrame;

  initPositions();
```

```
initVelocities();

initLifetimes();

hint(DISABLE_DEPTH_MASK);

//map3

PImage img = loadImage("petal.png");

PImage img2 = loadImage("petal_groc.png");

ps = new ParticleSystem(0, new PVector(width/2, height-60), img, img2);

//mapping1

rodona=loadImage("punt_groc.png");

rodona2=loadImage("punt_blau.png");

audio1 = new SoundFile(this, "audio1.mp3");

audio1.loop();

//mapping2

int temps = 0;

int temps2 = 0;

audio2 = new SoundFile(this, "audio2.mp3");

audio2.loop();

//mapping3

audio3 = new SoundFile(this, "audio3.mp3");

audio3.loop();

//mapping4
```

```
tiempo=millis();
audio4 = new SoundFile(this, "audio4.mp3");
audio4.loop();
//mapping5
audio5 = new SoundFile(this, "audio5.mp3");
audio5.loop();
//mapping6
for (int j = 0; j < 100; j++) {
  x[j] = int(random(0, 708));
  y[j] = int(random(0, 400));
  squareColors[j] = color(random(255), random(255), random(255));
}
groc=loadImage("punt_groc.png");
audio6 = new SoundFile(this, "audio6.mp3");
audio6.loop();
//mapping7
punt=loadImage("punt_groc.png");
audio7 = new SoundFile(this, "audio7.mp3");
audio7.loop();
//mapping8
particles = new ArrayList<Particle2>();
```

```
  audio8 = new SoundFile(this, "audio8.mp3");
  audio8.loop();
  //mapping9
  audio9 = new SoundFile(this, "audio9.mp3");
  audio9.loop();
  smooth();
  for (int l = 0; l < 100; l++) {
    particles3.add(new Particle3(random(width), random(height), random(10, 30)));
  }
  //mapping10
  audio10 = new SoundFile(this, "audio10.mp3");
  audio10.loop();
  for (int m = 0; m < 60; m++) {
    j[m] = random(0, 708);
    k[m] = 0;
    v[m] = random(1, 5);
    int x= 65 + int(random(25));
    l[m] = char(x);
  }
}
```

```
void draw() {
 background (0);
 if (estado==1) {
  map1(); //1bombolla_gran
 } else  if (estado==2) {
  map2(); //2sketch_3bombolla
 } else
  if (estado==3) {
   map3(); //3petalsmov
  } else
   if (estado==4) {
    mapping1(); //robot
   } else
    if (estado==5) {
     mapping2(); //efectetv
    } else
     if (estado==6) {
      mapping3(); //linia_canvi_color_posicio
     } else
      if (estado==7) {
       mapping4(); //canvi_fons_color
```

```
        } else
         if (estado==8) {
          mapping5(); //claustrofobic
         } else
          if (estado==9) {
           mapping6(); //tocarQuadrats_reapareixen
          } else
           if (estado==10) {
            mapping7(); //quadrat_desapareix
           } else
            if (estado==11) {
             mapping8(); //8a_rastre
            } else
             if (estado==12) {


              mapping9(); //8estrellitas
             } else
              if (estado==13) {
               mapping10(); //lletres_cauen
              }
spout.sendTexture();
```

```
}



void keyReleased() {
  if (key=='q') {
    estado=1;
  }
  if (key=='w') {
    estado=2;
  }
  if (key=='e') {
    estado=3;
  }
  if (key=='r') {
    estado=4;
  }
  if (key=='t') {
    estado=5;
  }
  if (key=='y') {
    estado=6;
```

```
}
if (key=='u') {
  estado=7;
}
if (key=='i') {
  estado=8;
}

if (key=='o') {
  estado=9;
}

if (key=='p') {
  estado=10;
}


if (key=='a') {
  estado=11;
}
```

```
  if (key=='s') {
    estado=12;
  }
  if (key=='d') {
    estado=13;
  }
}

//SECCIÓ 3
void map1() { //1bombolla_gran

  for (int i=0; i<bodies.size (); i++)
  {
    // PINTEM L'ELEMENT
    SkeletonData s = bodies.get(i);
    float x= s.skeletonPositions[0].x; // El número indica quina extremitat es vol dibuixar
    float y= s.skeletonPositions[0].y;
    image(bombolla, x*width-150, 150, 250, 250);
  }
}
void map2() { //2sketch_3bombolla
```

```
for (int i=0; i<bodies.size (); i++)
{

  // PINTEM L'ELEMENT
  SkeletonData s = bodies.get(i);
  float r= s.skeletonPositions[10].x; // El número indica quina extremitat es vol dibuixar
  float o= s.skeletonPositions[10].y;

  for (int n = 0; n < npartTotal; n++) {
    lifetimes[n]++;
    if (lifetimes[n] == partLifetime) {
      lifetimes[n] = 0;
    }

    if (0 <= lifetimes[n]) {
      float opacity = 1.0 - float(lifetimes[n]) / partLifetime;

      if (lifetimes[n] == 0) {
        // Re-spawn dead particle
        positions[n].x =  r*width-30;
```

```
        positions[n].y = o*height+50;


        float angle = random(0, TWO_PI);
        float u = random(0.5 * speed, 0.5 * speed);
        velocities[n].x = u * cos(angle);
        velocities[n].y = u * sin(angle);
      } else {
        positions[n].x += velocities[n].x;
        positions[n].y += velocities[n].y;


        velocities[n].y += gravity;
      }
      drawParticle(positions[n], opacity);
    }
  }
  spout.sendTexture();
}


fcount += 1;
int m = millis();
if (m - lastm > 1000 * fint) {
```

```
    frate = float(fcount) / fint;
    fcount = 0;
    lastm = m;
    println("fps: " + frate);
  }
}



void map3() { //3petalsmov
  ps.run();
  for (int j=0; j<bodies.size (); j++)
  {

    // PINTEM L'ELEMENT
    SkeletonData s = bodies.get(j);
    float r= s.skeletonPositions[10].x; // El número indica quina extremitat es vol dibuixar
    float o= s.skeletonPositions[10].y;

    //PVector wind = new PVector(10, 0);
    //ps.applyForce(wind);
```

```
    ps.origin.x=r*width-30;
    ps.origin.y=o*height+50;
  }
  for (int i = 0; i < 2; i++) {
    ps.addParticle();
  }
}

//////VIDEO destello
//////VIDEO llum secuencia
//////textura?


//SECCIÓ 4
void mapping1() { //robot
  for (int i=0; i<bodies.size (); i++)
  {
    audio1.play();
    SkeletonData s = bodies.get(i);
    float x= s.skeletonPositions[3].x; // El número indica quina extremitat es vol dibuixar
    float y= s.skeletonPositions[3].y;
```

```
image(rodona, x*350, y*350, 30, 30);

image(rodona2, x*350+300, y*350, 30, 30);


float a= s.skeletonPositions[10].x; // El número indica quina extremitat es vol dibuixar

float b= s.skeletonPositions[10].y;


image(rodona, a*350, b*350, 30, 30);

image(rodona2, a*350+300, b*350, 30, 30);


float c= s.skeletonPositions[6].x; // El número indica quina extremitat es vol dibuixar

float d= s.skeletonPositions[6].y;

image(rodona, c*350, d*350, 30, 30);

image(rodona2, c*350+300, d*350, 30, 30);


float e= s.skeletonPositions[9].x; // El número indica quina extremitat es vol dibuixar

float f= s.skeletonPositions[9].y;

image(rodona, e*350, f*350, 30, 30);

image(rodona2, e*350+300, f*350, 30, 30);


float g= s.skeletonPositions[5].x; // El número indica quina extremitat es vol dibuixar

float h= s.skeletonPositions[5].y;
```

```
image(rodona, g*350, h*350, 30, 30);

image(rodona2, g*350+300, h*350, 30, 30);


float v= s.skeletonPositions[2].x; // El número indica quina extremitat es vol dibuixar

float j= s.skeletonPositions[2].y;

image(rodona, v*350, j*350, 30, 30);

image(rodona2, v*350+300, j*350, 30, 30);


float z= s.skeletonPositions[2].x; // El número indica quina extremitat es vol dibuixar

float w= s.skeletonPositions[2].y;

image(rodona, z*350, w*350, 30, 30);

image(rodona2, z*350+300, w*350, 30, 30);


float k= s.skeletonPositions[0].x; // El número indica quina extremitat es vol dibuixar

float l= s.skeletonPositions[0].y;

image(rodona, k*350, l*350, 30, 30);


image(rodona2, k*350+300, l*350, 30, 30);


float m= s.skeletonPositions[13].x; // El número indica quina extremitat es vol dibuixar

float n= s.skeletonPositions[13].y;
```

```
image(rodona, m*350, n*350, 30, 30);


image(rodona2, m*350+300, n*350, 30, 30);


float o= s.skeletonPositions[17].x; // El número indica quina extremitat es vol dibuixar
float p= s.skeletonPositions[17].y;
image(rodona, o*350, p*350, 30, 30);


image(rodona2, o*350+300, p*350, 30, 30);


float q= s.skeletonPositions[18].x; // El número indica quina extremitat es vol dibuixar
float r= s.skeletonPositions[18].y;
image(rodona, q*350, r*350, 30, 30);


image(rodona2, q*350+300, r*350, 30, 30);


float u= s.skeletonPositions[14].x; // El número indica quina extremitat es vol dibuixar
float t= s.skeletonPositions[14].y;
image(rodona, u*350, t*350, 30, 30);


image(rodona2, u*350+300, t*350, 30, 30);
```

```
  }
}
void mapping2() { //1efecteTV
  noStroke();
  audio2.play();
  if (temps < 180) {
    temps++;
    fill(255);
  } else {
    temps2++;
  }

  rect(0, 0, temps2, 708);
  rect(708 - temps2, 0, temps2, 708);
  rect(0, 0, 708, temps);
  rect(0, 708 - temps, 708, temps);
}
```

```
//VIDEO LINIES CLAUSTRoO

void mapping3() { //linia_canvi_color_posicio

  for (int i=0; i<bodies.size (); i++)

  {

    audio3.play();

    // PINTEM L'ELEMENT

    SkeletonData s = bodies.get(i);

    float posiciox= s.skeletonPositions[1].x; // El número indica quina extremitat es vol dibuixar

    float posicioy= s.skeletonPositions[1].y;



    color inici = color(10, 0, 0);

    color fin = color(0, 0, 255);



    // color en funció x

    float t = map(lineaX, 0, width, 0, 1);

    color actual = lerpColor(inici, fin, t);



    strokeWeight(150);

    stroke(actual);
```

```
    line(lineaX, 0, lineaX, height);


    lineaX = posiciox*width-50;


    // no superi limits
    if (lineaX < 0 || lineaX > width) {
      lineaX = constrain(lineaX, 0, width);
    }
  }
}
void mapping4() { //canvi_fons_color
  background(u, g, b, a);
  audio4.play();
  if (millis()-tiempo>1500) {
    u = random(0, 255);
    g = random(0, 255);
    b = random(0, 255);
    a =  random(0, 255);


    tiempo=millis();
  }
```

```
}


void mapping5() { //claustrofobic
  noStroke();
  audio5.play();
  for (int i=0; i<bodies.size (); i++)
  {

    // PINTEM L'ELEMENT
    SkeletonData s = bodies.get(i);
    float capx= s.skeletonPositions[3].x; // El número indica quina extremitat es vol dibuixar
    float capy= s.skeletonPositions[3].y;
    // posició
    float mouseYPos = capy*height+100;
    float rectY = height - mouseYPos;
    fill(255);
    rect(0, -rectY, width, mouseYPos);
    mouseYPos += 2; // moviment rect
  }
}
```

```
void mapping6() { //tocarQuadrats_reapareixen
  for (int j = 0; j < 200; j++) {
    noFill();
    audio6.play();
    stroke(squareColors[j]);
    rect(x[j], y[j], 60, 60);
    for (int i=0; i<bodies.size (); i++)
    {

      // PINTEM L'ELEMENT
      SkeletonData s = bodies.get(i);
      float max= s.skeletonPositions[7].x; // El número indica quina extremitat es vol dibuixar
      float may= s.skeletonPositions[7].y;

      image(groc, max*width-100, may*height, 20, 20);

      float d = dist(max*width-100, may*height, x[j] + 30, y[j] + 30);
      if (d < 30) {
        x[j] = int(random(0, 708));
        y[j] = int(random(0, 400));
      }
```

```
    }
  }
}

void mapping7() { //quadrat_desapareix
  for (int j = 0; j < 200; j++) {
    noFill();
    audio7.play();
    stroke(squareColors[j]);
    rect(x[j], y[j], 60, 60);
    for (int i=0; i<bodies.size (); i++)
    {

      // PINTEM L'ELEMENT
      SkeletonData s = bodies.get(i);
      float max= s.skeletonPositions[7].x; // El número indica quina extremitat es vol dibuixar
      float may= s.skeletonPositions[7].y;
      image(punt, max*width-100, may*height, 20, 20);
      float d = dist(max*width-100, may*height, x[j] + 30, y[j] + 30);
      if (d < 30) {
        squareColors[j]=0;
```

```
    }
   }
  }
}

void mapping8() { //8a_rastre
  for (int i=0; i<bodies.size (); i++)
  {
   audio8.play();
   // PINTEM L'ELEMENT
   SkeletonData s = bodies.get(i);
   float ratolix= s.skeletonPositions[11].x; // El número indica quina extremitat es vol dibuixar
   float ratoliy= s.skeletonPositions[11].y;
   particles.add(new Particle2(ratolix*width, ratoliy*height));
   for (int k = particles.size() - 1; k >= 0; k--) {
    Particle2 p = particles.get(k);
    p.update();
    p.display();
    if (p.alpha <= 0) {
     particles.remove(k);
```

```
      }
    }
  }
}
void mapping9() { //8estrellitas
  for (Particle3 p : particles3) {
    noStroke();
    audio9.play();
    p.update2();
    p.rodones();
  }
}

void mapping10() { //lletres_cauen
  for (int m = 0; m < 60; m++) {
    fill(255);
    audio10.play();
    //rect(x[i], y[i], 3, 6);
    text(l[m], j[m], k[m]);
    k[m] = k[m] + v[m];
    if (k[m] > 708) {
```

```
      k[m] = 0;
    }
  }
}
void drawParticle(PVector center, float opacity) {
  beginShape(QUAD);
  noStroke();
  tint(255, opacity * 255);
  texture(bombolla1);
  normal(0, 0, 1);
  vertex(center.x - partSize/1.5, center.y - partSize/1.5, 0, 0);
  vertex(center.x + partSize/1.5, center.y - partSize/1.5, bombolla1.width, 0);
  vertex(center.x + partSize/1.5, center.y + partSize/1.5, bombolla1.width,
bombolla1.height);
  vertex(center.x - partSize/1.5, center.y + partSize/1.5, 0, bombolla1.height);
  endShape();
}

void initPositions() {
  positions = new PVector[npartTotal];
  for (int n = 0; n < positions.length; n++) {
```

```
    positions[n] = new PVector();
  }
}

void initVelocities() {
  velocities = new PVector[npartTotal];
  for (int n = 0; n < velocities.length; n++) {
    velocities[n] = new PVector();
  }
}

void initLifetimes() {
  // Initializing particles with negative lifetimes so they are added
  // progressively into the screen during the first frames of the sketch
  lifetimes = new int[npartTotal];
  int t = -1;
  for (int n = 0; n < lifetimes.length; n++) {
    if (n % npartPerFrame == 0) {
      t++;
    }
    lifetimes[n] = -t;
```

```
  }
}

void appearEvent(SkeletonData _s) {
  if (_s.trackingState == Kinect.NUI_SKELETON_NOT_TRACKED)
  {
    return;
  }
  synchronized(bodies) {
    bodies.add(_s);
  }
}

void disappearEvent(SkeletonData _s)
{
  synchronized(bodies) {
    for (int i=bodies.size ()-1; i>=0; i--)
    {
      if (_s.dwTrackingID == bodies.get(i).dwTrackingID)
      {
        bodies.remove(i);
```

```
    }
   }
  }
}

void moveEvent(SkeletonData _b, SkeletonData _a)
{
 if (_a.trackingState == Kinect.NUI_SKELETON_NOT_TRACKED)
 {
  return;
 }
 synchronized(bodies) {
  for (int i=bodies.size ()-1; i>=0; i--)
  {
   if (_b.dwTrackingID == bodies.get(i).dwTrackingID)
   {
    bodies.get(i).copy(_a);
    break;
   }
  }
 }
```

```
}
```

Particle

```
// A simple Particle class, renders the particle as an image
class Particle {
  PVector loc;
  PVector vel;
  PVector acc;
  float lifespan;
  PImage img;

  Particle(PVector l, PImage img_, PImage img2_) {
    acc = new PVector(0, 0);
    float vx = randomGaussian()*0.3;
    float vy = randomGaussian()*0.3 - 1.0;
    vel = new PVector(vx, vy);
    loc = l.copy();
    lifespan = 1000.0;
    int r = int(random(2));
    if (r==0) img = img_;
    else img = img2_;
```

```
}

void run() {
  update();
  render();
}

// Method to apply a force vector to the Particle object
// Note we are ignoring "mass" here
void applyForce(PVector f) {
  acc.add(f);
}

// Method to update position
void update() {
  vel.add(acc);
  loc.add(vel);
  lifespan -= 2.5;
  acc.mult(0); // clear Acceleration
}
```

```
// Method to display
void render() {

  // PINTEM L'ELEMENT
  imageMode(CENTER);
  tint(255, lifespan);
  image(img, loc.x, loc.y, 20, 20);
  // Drawing a circle instead
  // fill(255,lifespan);
  // noStroke();
  // ellipse(loc.x,loc.y,img.width,img.height);
}

// Is the particle still useful?
boolean isDead() {
  if (lifespan <= 0.0) {
    return true;
  } else {
    return false;
  }
}
```

```
}
```

Particle 2

```
class Particle2 {
  float x, y;
  float diameter;
  float alpha;

  Particle2(float x, float y) {
    this.x = x;
    this.y = y;
    this.diameter = 50;
    this.alpha = 255;
  }

  void update() {
    diameter -= 0.5;
    alpha -= 5;
  }

  void display() {
```

```
    noStroke();
    float r=random(50, 255);
    float g=random(50, 255);
    float b=random(50, 255);
    fill(r, g, b, alpha);
    ellipse(x, y, diameter, diameter);
  }
}
```

Particle 3

```
class Particle3 {

  PVector position;
  float diameter;
  float growthRate;
    float w;
  float n;
  float o;

  Particle3(float x, float y, float initialDiameter) {
    position = new PVector(x, y);
```

```
    diameter = initialDiameter;
    growthRate = random(0.5, 2);
  }
  void update2() {
    diameter -= growthRate;

    if (diameter <= 0) {
      diameter = random(10, 40);
      position.x = random(width);
      position.y = random(height);
      growthRate = random(0.5, 3);
    }
  }
  void rodones() {
    w= random(50, 255);
    n= random(50, 255);
    o= random(50, 255);
    fill(w, n, o);
    ellipse(position.x, position.y, diameter, diameter);
  }
}
```

Particle System

```
// A class to describe a group of Particles

// An ArrayList is used to manage the list of Particles


class ParticleSystem {


  ArrayList<Particle> particles;    // An arraylist for all the particles

  PVector origin;                   // An origin point for where particles are birthed

  PImage img;

  PImage img2;

  ParticleSystem(int num, PVector v, PImage img_, PImage img2_) {

    particles = new ArrayList<Particle>();              // Initialize the arraylist

    origin = v.copy();                                  // Store the origin point

    img = img_;

    img2 = img2_;

    for (int i = 0; i < num; i++) {

      particles.add(new Particle(origin, img, img2));

    }

  }


  void run() {
```

```
    for (int i = particles.size()-1; i >= 0; i--) {
      Particle p = particles.get(i);
      p.run();
      if (p.isDead()) {
        particles.remove(i);
      }
    }
  }

  // Method to add a force vector to all particles currently in the system
  void applyForce(PVector dir) {
    // Enhanced loop!!!
    for (Particle p : particles) {
      p.applyForce(dir);
    }
  }

  void addParticle() {
    particles.add(new Particle(origin, img, img2));
  }
}
```

6. Bibliografia

McGregor, W. (s.d). https://waynemcgregor.com/research/choreographic-language-agent

TED. (2012, setembre 14). Wayne McGregor: El proceso creativo de un coreógrafo en tiempo real [Vídeo]. https://www.youtube.com/watch?v=KPPxXeoIzRY&t=5s

TED. (2019, juny 3). Sleep is your superpower – Matt Walker [Vídeo]. https://www.youtube.com/watch?v=YXn-eNPzlo8

TED. (2021, novembre 24). The surprising health benefits of dreaming – Sleeping with Science [Vídeo]. https://www.youtube.com/watch?v=YXn-eNPzlo8

TEDx Talks. (2013, novembre 5). Lucid Dreams – Tim Post - TEDxTwenteU[Vídeo]. https://www.youtube.com/watch?v=OK3SfNxbK3Y

TEDx Talks. (2014, setembre 22). Lucid dreams as a bridge between realities – Chongtul Rinpoche – TEDxFultonStreet [Vídeo]. https://www.youtube.com/watch?v=exjlR7izakg

TED-Ed. (2015, desembre 10). Why do we dream? - Amy Adkins [Vídeo]. https://www.youtube.com/watch?v=2W85Dwxx218