A TALE OF TWO TOMBS

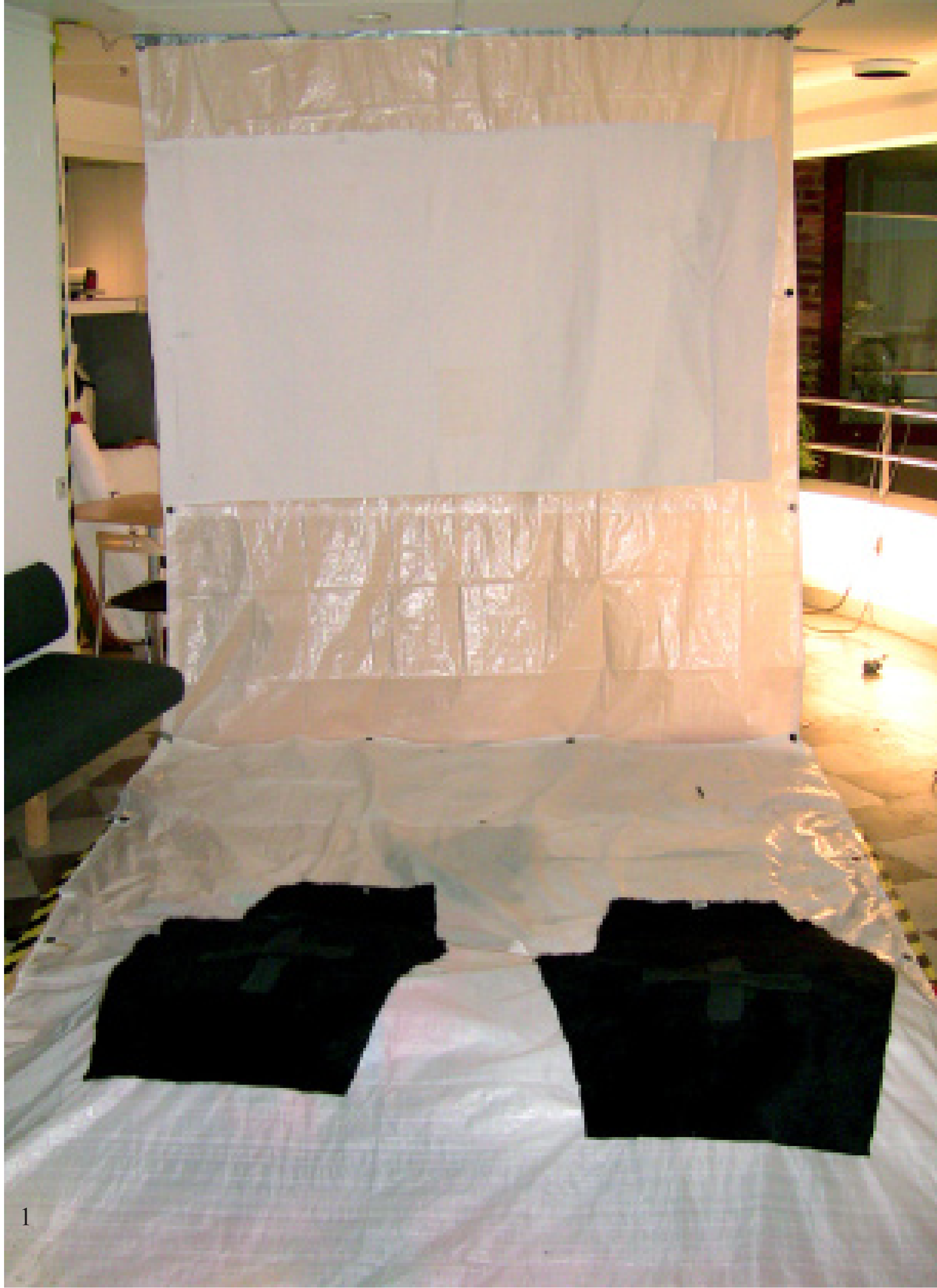
Experience prototyping report
Fall 2008 Institute of Design

Jose Ledon, Ruedee Sarawutpaiboon, Sotiris Kotronias

Index:

(Taking care of last details)

(*Installation without the video projection*)



Abstract:

The report is about the project of an interactive installation that was made for the Experience prototype course in Umea Institute of Design by Jose Ledon, Rueedee Sarawutpaiboon and Sotiris Kotronias. The two main challenges for the project were firstly to follow the brief that was given and secondly to fit in the environment of the Halloween Party that was organized that period by the students in the school and was supposed to host the installation.

Due to the brief it was necessary to focus on the magic aspect of wireless and mobile connectivity. But building wireless projects, within a frame of very tight time and budget,can be demanding. At some point it is necessary to compromise and simplify your ideas or even fake the wireless functions.

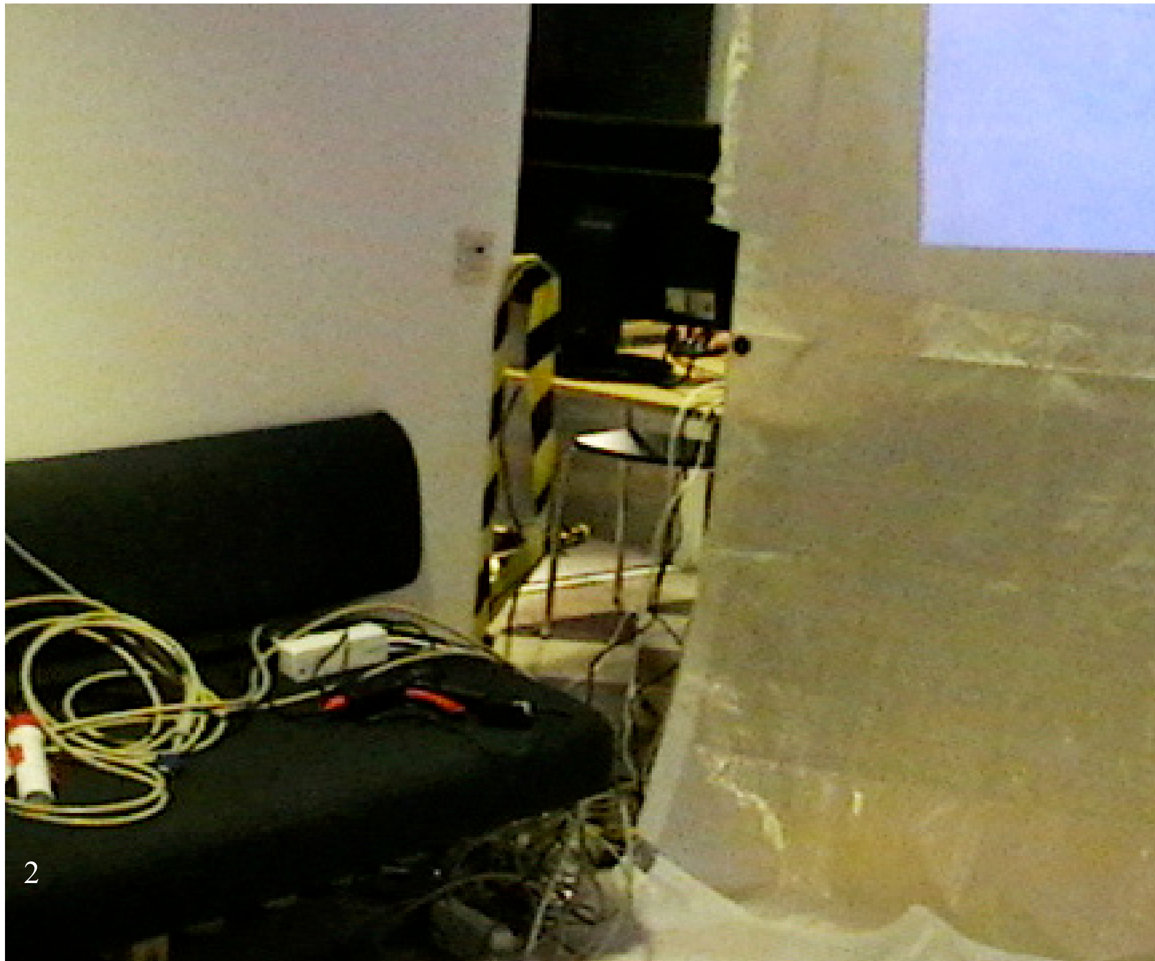
It was also necessary to talk with the organizers of the party in order to figure out realize where the installation will fit better in the space that would host the party.

The brief:

Cut the wires
The magic in wireless and mobile interaction

The main goal is to focus on the magic aspect of wireless and mobile connectivity. To Exchange information over an invisible and intangible medium, result in a different experiences and expectations for users. By removing or hiding the wires, users can interact, control and be connected with various systems while on the move or in the new situations/ contexts.

In general, building wireless projects is demanding and usually forces you to simplify your ideas/concepts and/or fake the wireless functions to successfully build prototypes with the time, technical issue and monetary constrains in place.



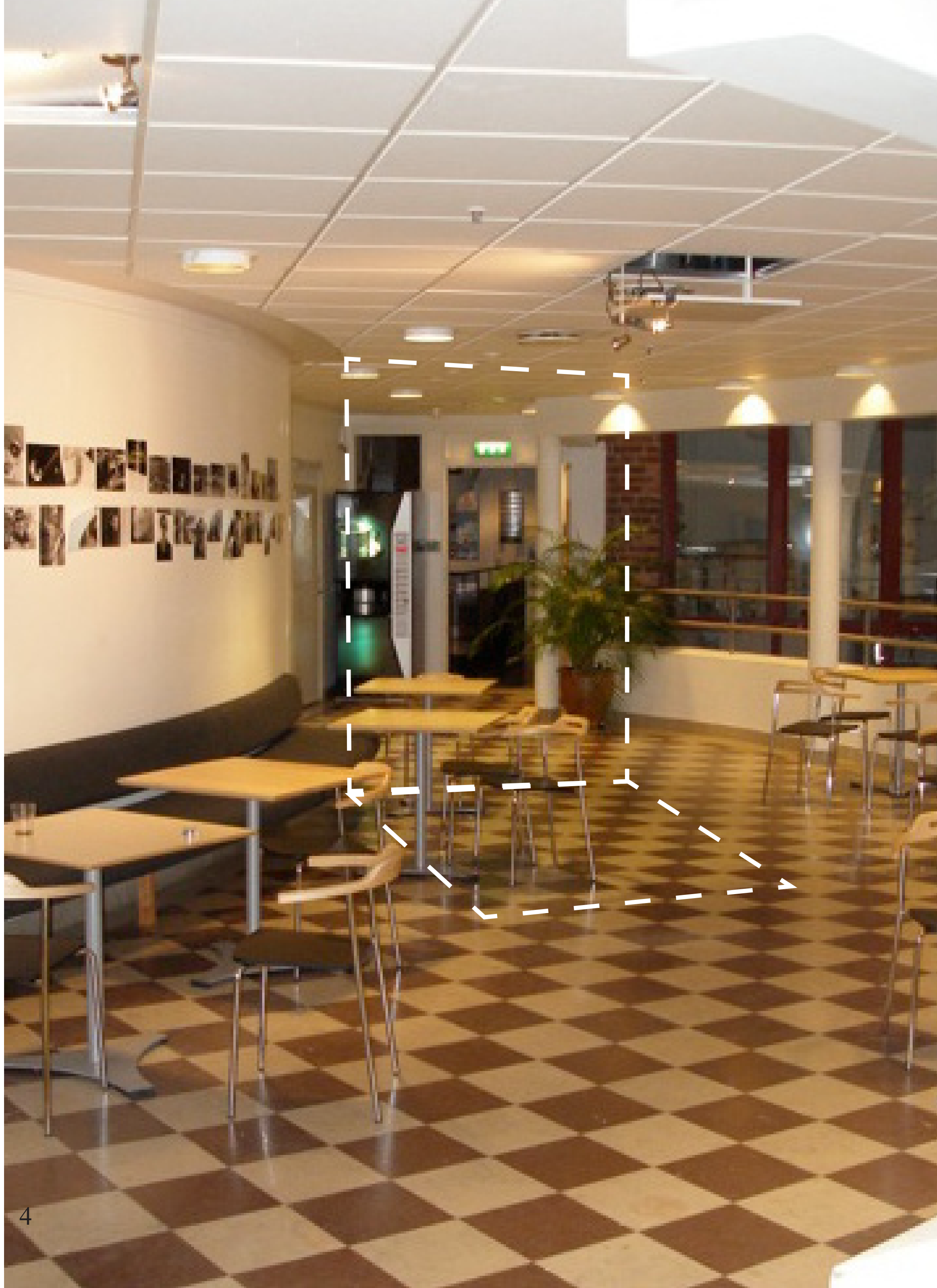
(Shaping the space)

Goal/Ideation:

Initially the idea was to create a lounge area within the party space that would be able to provide the user with an additional experience through an interactive installation.

The installation was supposed to form two coffins were the people were able to lie on. Two cameras would capture their images while they lie on the coffins and a video projection of themselves will take place on a big screen in front of them. The video of the projection proceeded through special software so it creates an effect of a ghost image. In that way we tried to fake a feeling of “life after death” which was the main aim of this installation experience. The idea for the installation was inspired by the Halloween party theme.

(*Space before the installation*)



Practical issues:

1. Finding the right place to set up the installation was one of the first issues of the project. The location was chosen after taking under consideration where the lounge area of the party was. Another aspect to take under consideration was that the installation had to service as also a boundary in order to isolate an area of the building that was not supposed to be reached from the attendants of the party.

2. The low budget for setting up the installation was another issue. It became necessary to use as few materials as possible and that costs to the experience of the users mainly because of the cheap feeling of the fabrics and the plastic cover of the whole installation. Another disadvantage of the low budget was the poor quality of the screen. The result was that the resolution of the video projection was not as good as we expected.

3. Finally it was necessary to take under consideration that the installation would have been in a party area. Thus for it should have been safe and endure to any kind of accident or unpredicted behavior of the party attendants.

(*Placing the sensors and hiding all the wires for safety and wireless illusion*)

Process & Method:

Shaping the space:

The installation consists of an area on the floor in lounge that includes the two coffins where people can lay on, of a vertical screen for the projection, a projector, two cameras and the sensors that will trigger the interaction between the installation and the people. The area of the installation was defined by two white plastic sheets. One of the sheets was vertical and worked as a screen for the video projection. Some fabrics and another plastic sheet were enough to create the floor layer where the two coffins were created on. The two coffins were made by two bean bags glued on the fabrics and covered with a black fabric.

Technical issues:

We had to use two censors, one under each of the bean bags, so when people sit on the bean bags; it would have been possible to trigger the interaction we were looking for. For this purpose a pressure sensor and a hit sensor were used. It was necessary to test if they would react or if they would give the right output because of the bean bags that would cover them. The big surface of the bean bags might not apply the right pressure on the sensors and especially on the pressure sensor because of its small active surface.

Another reason that might interrupt the sensors output is that the bean bags changes their shape and position a lot while there're used by people. We ended up binding and gluing the bags under the fabrics of the installation. So even when they are used they don't transform or move as much as usual. We succeeded to make the sensor work the way we wished. During the tests we got good feedback from the sensors with different volumes while people were sitting on them. The different volumes weren't so precise but that was not necessary for the use of that installation.

(Our attempt to appear wireless,
We ended up hide all the wires and hardwares on the ceiling)

After the sensors were placed under the bean bags, their cables were covered with tape and hidden through the installation. We used long wires so they can reach all the way up the walls and ceiling in the area of the installation where all the hardware was hidden in order to be far from the reach of the people.

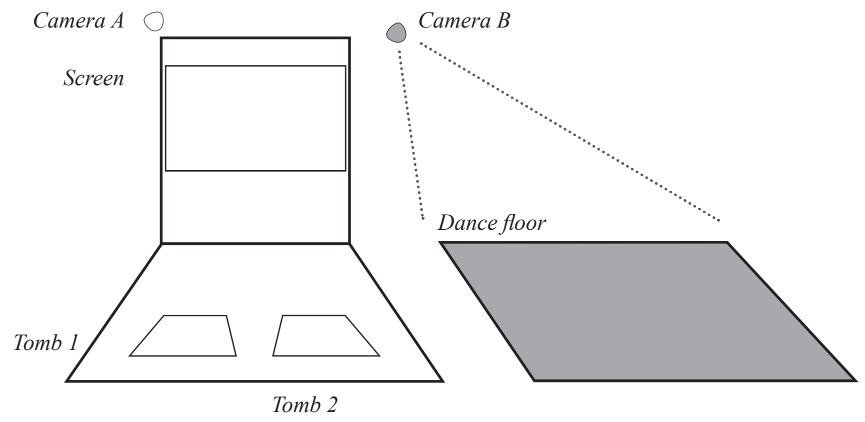
The sensors were connected to an Arduino board. Using the Analog Input from the examples of the Arduino website (www.arduino.cc) we adjusted the example in order to receive two inputs from the sensors and process the signal in order to further interact with the installation.

The installation was further included a mac mini, two isight cameras and a projector. The Arduino board, the two cameras and the projector were connected to the mac mini which was doing all the software support of the installation.

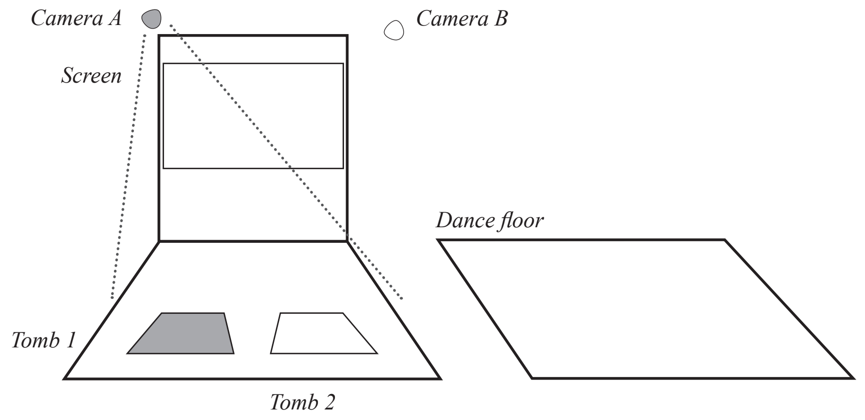
One of the cameras was capturing an image from the dancing floor of the party. The other camera was viewing the area of the two coffins where people suppose to lay. When nobody was laying on the installation the projector was projecting the video from the camera which was looking to the dance floor. When somebody sits on one of the coffins the sensor was triggering the use of the second camera. Then the video projection was from the area of the two coffins. Finally when a second person uses the second coffin, then the sensor from that coffin was activating the software that was distorting the video projection in order to produce the effect of "life after death" which was the main idea of the whole installation. *(See following illustration)*

After that, to achieve the wireless goal and to protect all the hardware parts. We hide all of them on the ceiling and taped all the wires in line under the sofa along the lounge area wall.

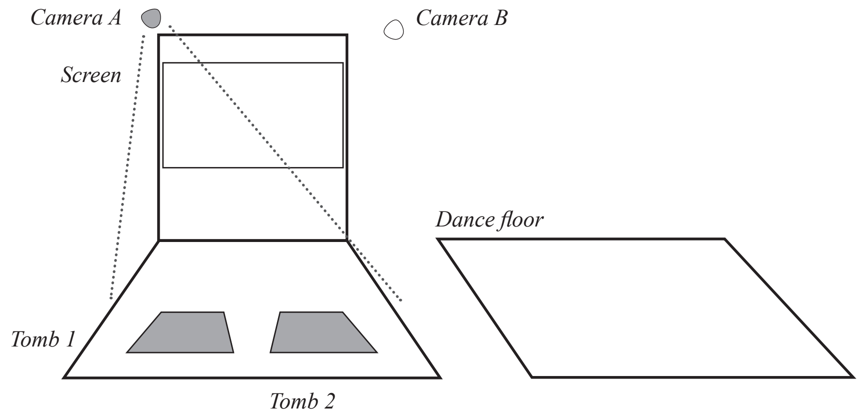
(Arduino board)



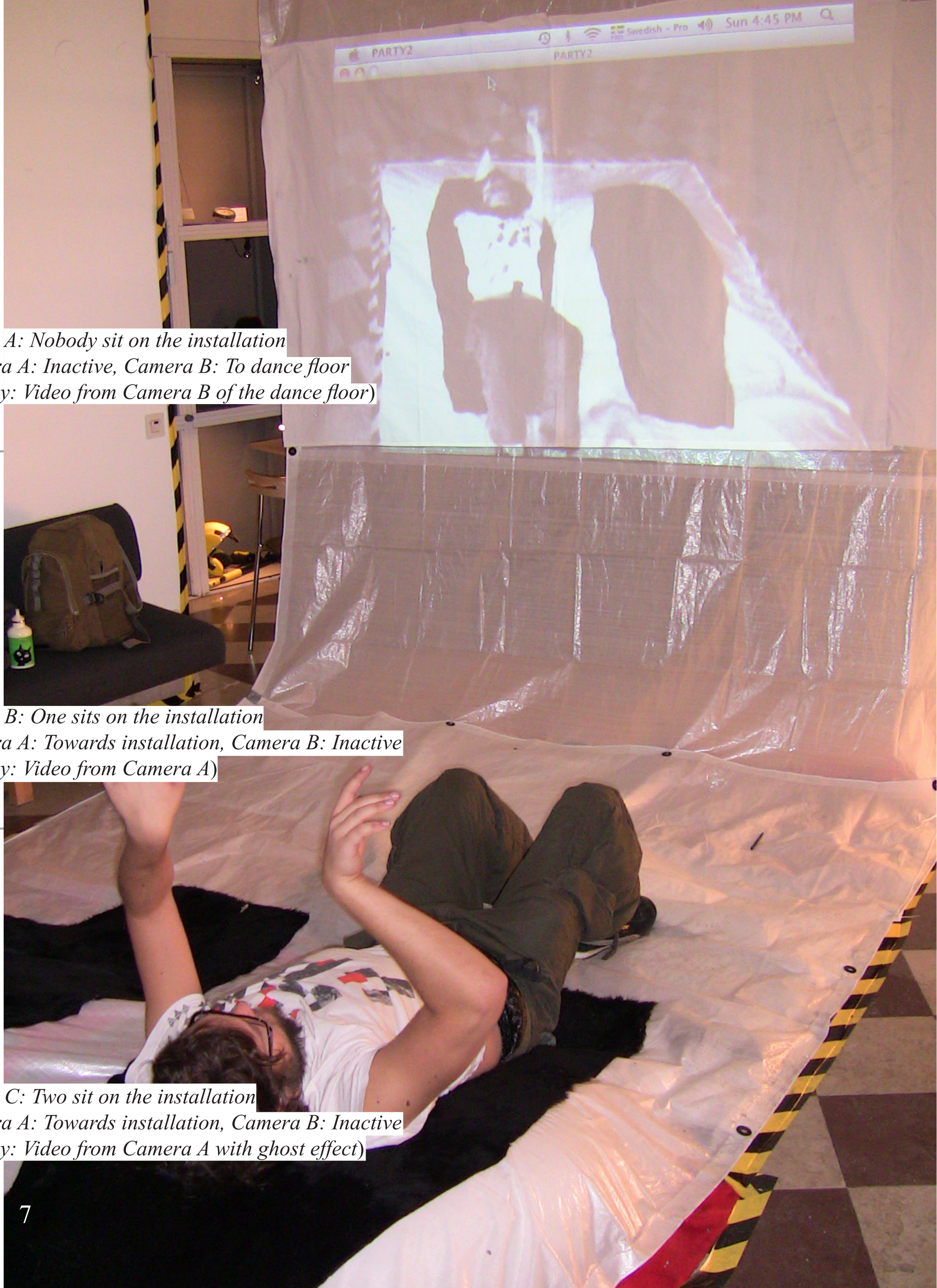
(Mode A: Nobody sit on the installation
Camera A: Inactive, Camera B: To dance floor
Display: Video from Camera B of the dance floor)

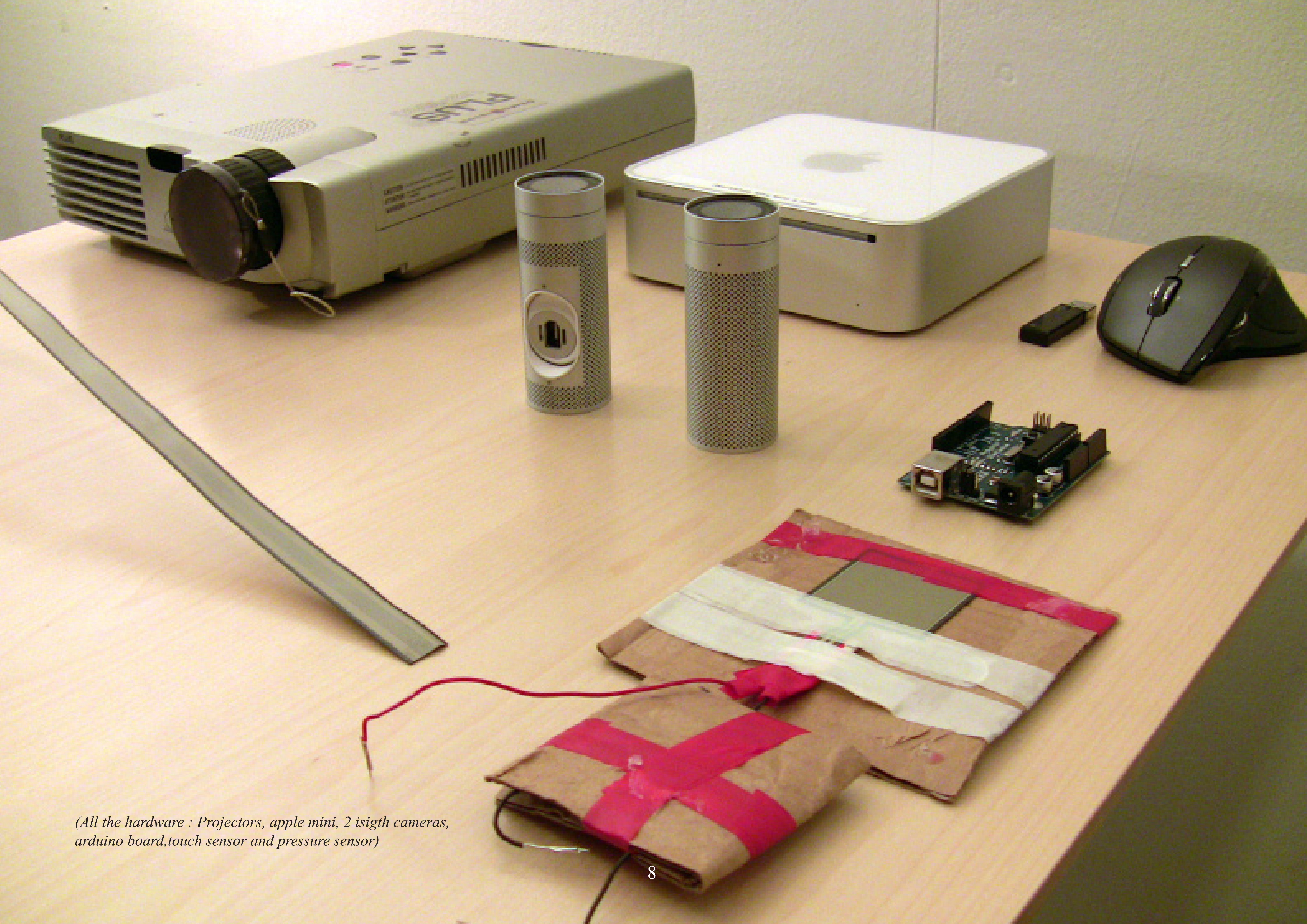


(Mode B: One sits on the installation
Camera A: Towards installation, Camera B: Inactive
Display: Video from Camera A)



(Mode C: Two sit on the installation
Camera A: Towards installation, Camera B: Inactive
Display: Video from Camera A with ghost effect)




(All the hardware : Projectors, apple mini, 2 isigth cameras, arduino board,touch sensor and pressure sensor)

Concept of the Video:

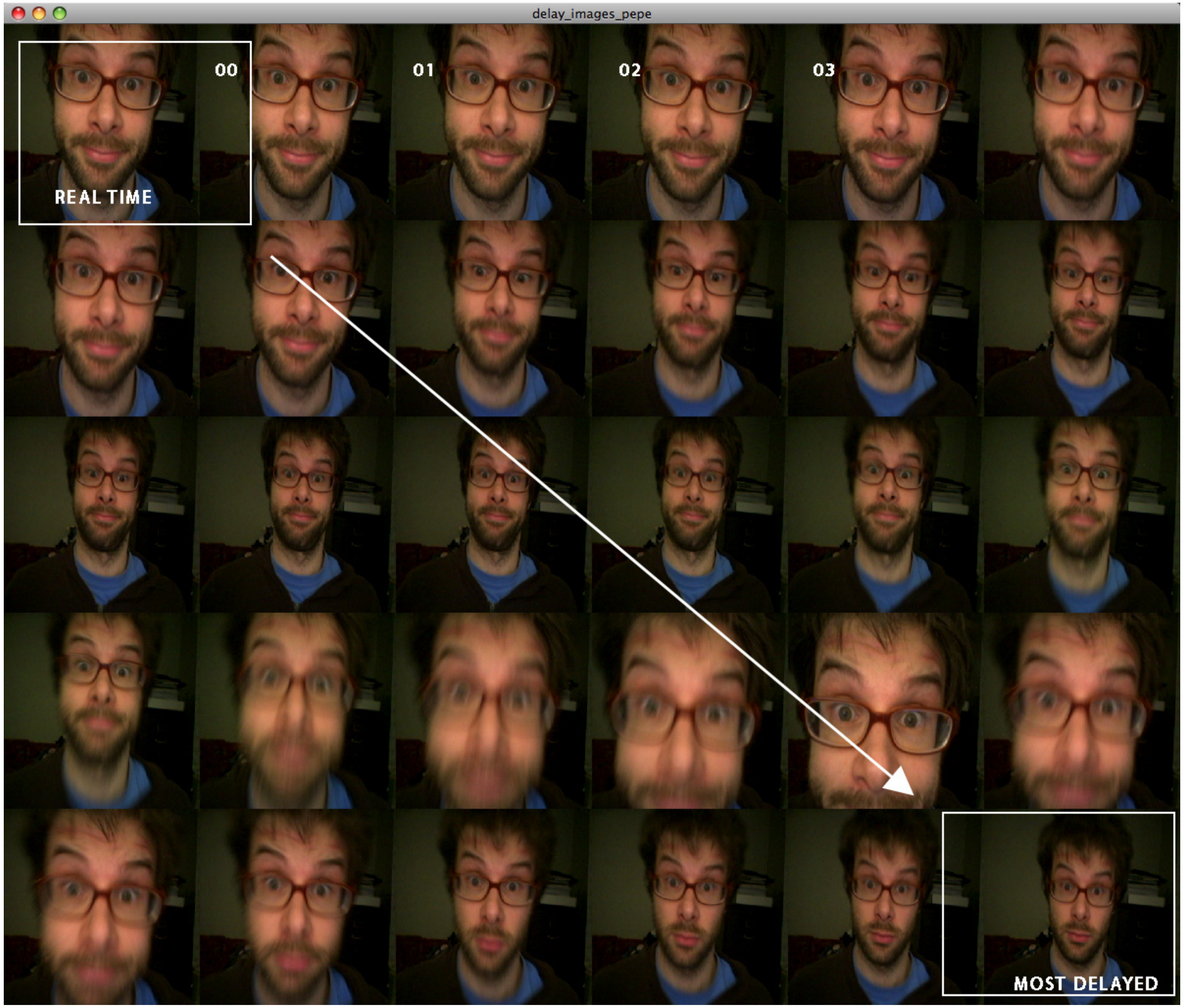
As part of the whole installation we develop a program using Processing that will capture video and broadcast it later with a delay.

Halloween is a special date because all the parties are full of monsters, ghosts, aliens, etc. The general intention is to produce horror in the others, nevertheless commonly many customs became funny. For this project we developed an interactive video installation with the aim that people can interact with themselves in a “horrific” way by being able to watch their own ghost…

STEP #1:
The program works recording a frame of video, and playing it a few seconds after being recorded, a simple delay that can vary using code. (Image 1 and CODE 1 “Mosaic of Images”)

Code Sample #1:

```
//CODE 1  “Mosaic of images”
//By Camille Moussette
import processing.video.*;
PImage myImage = createImage(600, 600, RGB);
PImage[] moreImages = new PImage[64]; //datatype[]
varvar[element] = valuevar.length
String cam1 = “DV Video”;//name of your own camera
int counter = 0 ;
int delayCounter = 0;
int delayInterval = 1 ;
Capture myCapture;

void setup()
{
 size(1200, 1000);
 myCapture = new Capture(this, 200, 200,””, 24);
}

void captureEvent(Capture myCapture) {
 if (counter == 0){  // init all images to black
  println(“init”);
  for ( int k = moreImages.length - 1; k >= 0;  k--){
   moreImages[k] =myImage;
  }
 }
 if (delayCounter >=  delayInterval){  //println(“capturing”);
  for ( int k = moreImages.length - 1; k > 0;  k--){ // shift all the
  images in the array
   moreImages[k] = moreImages[k-1];
  }
 }
 moreImages[0] = myImage;  // last one to update
 delayCounter = 0;   // clear the interval counter
 myCapture.read();  // capture latest frame/image
 myImage = myCapture.get();  //myCapture1.read();
 delayCounter++;//add one unit to counter
}

void draw() {
 image(myImage, 0, 0);
 for ( int k = 1; k < 6;  k++){  // first row
  image(moreImages[k], k * 200, 0);
 }
 for ( int k = 0; k < 6; k++){ // second row
  image(moreImages[k+6], k * 200, 200);
 }
 for ( int k = 0; k < 6; k++){ // third row
  image(moreImages[k+12], k * 200, 400);
 }
 for ( int k = 0; k < 6; k++){// fourth row
  image(moreImages[k+18], k * 200, 600);
 }
 for ( int k = 0; k < 6; k++){ // fifth row
  image(moreImages[k+24], k * 200, 800);
 }  //add more rows or extend the columns indefinetly
 counter++;
}
```

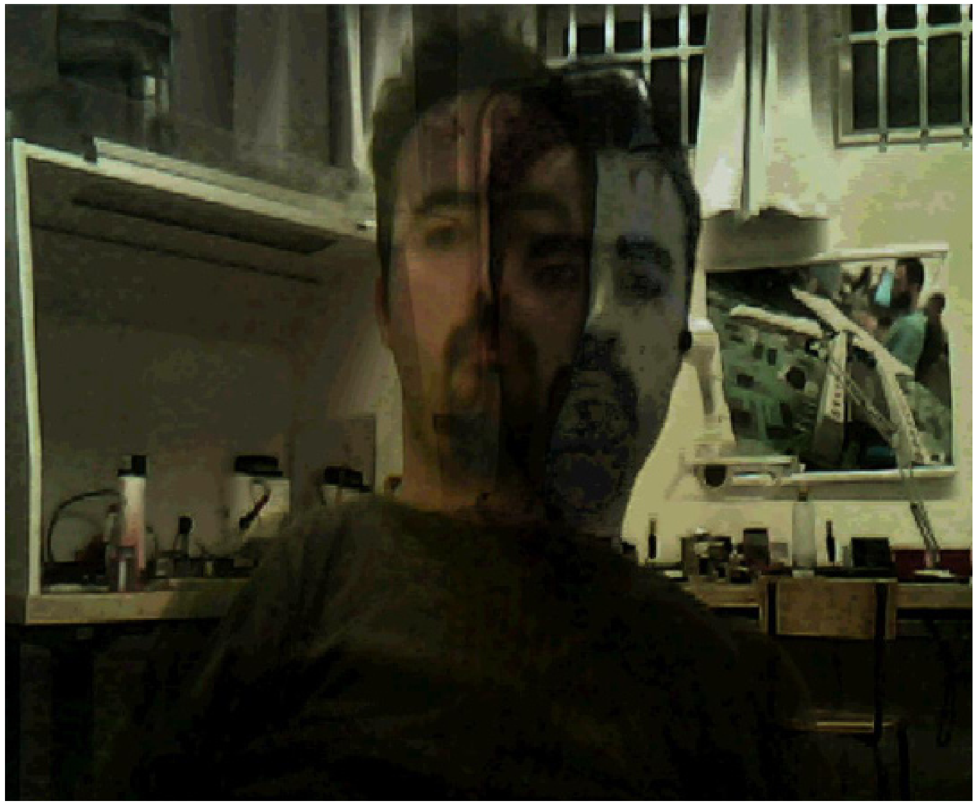


(Image 1 Delayed camera capture images see CODE 1 "Mosaic of Images")

STEP #2:
Using this idea of delay, we combined one image in real time and one delayed; playing at the same time in the same display. The result is a "ghosted" image of the real one. The background (still objects) remains the same, but the moving objects duplicates to have a "ghost" effect.

Code Sample #2:

```
//CODE 2  "Ghost"
// By Camille Moussette & Jose Ledon
import processing.video.*;
PImage myImage = createImage(640, 480, RGB);
PImage[] moreImages = new PImage[164];
int counter = 0 ;
int delayCounter = 0;
int delayInterval = 1 ;
Capture myCapture;

void setup()
{
  size(640, 480);
  frameRate(15);
  myCapture = new Capture(this, 640, 480,"", 12);
}

void captureEvent(Capture c) {
  if (counter == 0){ // init all images to black
    println("capture Event");
    for ( int k = moreImages.length - 1; k >= 0;  k--){
      moreImages[k] =myImage;
    }
  }
  if (delayCounter >=  delayInterval){ // shift all the images in the
     array
     for ( int k = moreImages.length - 1; k > 0;  k--){
      moreImages[k] = moreImages[k-1];
    }
    moreImages[0] = myImage;  // last one to update
    delayCounter = 0; // clear the interval counter
  }
  myCapture.read(); // capture latest frame/image
  myImage = myCapture.get();
  delayCounter++;
}

void draw(){
  background(0);
  PImage test = moreImages[0].get();//moreImages[0] = realtime
  image, change the number to have a delayed image
  myImage.mask(moreImages[50]);//Increse the number [50] to have
  more delayed for the ghost
  image(test, 0, 0);//if you use "myImage" insted of "test" = flickering
  counter++;
}
```


(Image 2. Two video captures combined see CODE 2 "Ghost")

STEP #3:
Since the method used to combine this two images works by masking one image into another by subtracting the whiter colors of a grayscale image into the other; the result shows a difference in colors, with the real time image in color and the ghosted image in gray scale. For this reason we chose to convert both images into gray scale using a video filter.

Code Sample #3:

```
//CODE 3 Color Fix

void draw(){
  background(0);
  PImage test = moreImages[0].get();//moreImages[0] = realtime
  image, change the number to have a delayed image
  myImage.mask(moreImages[50]);//Increse the number [50] to have
  more delayed for the ghost
  image(test, 0, 0);//if you use "myImage" insted of "test" = flickering
  filter(GRAY);//convert both images to Grayscale
  counter++;
}
```

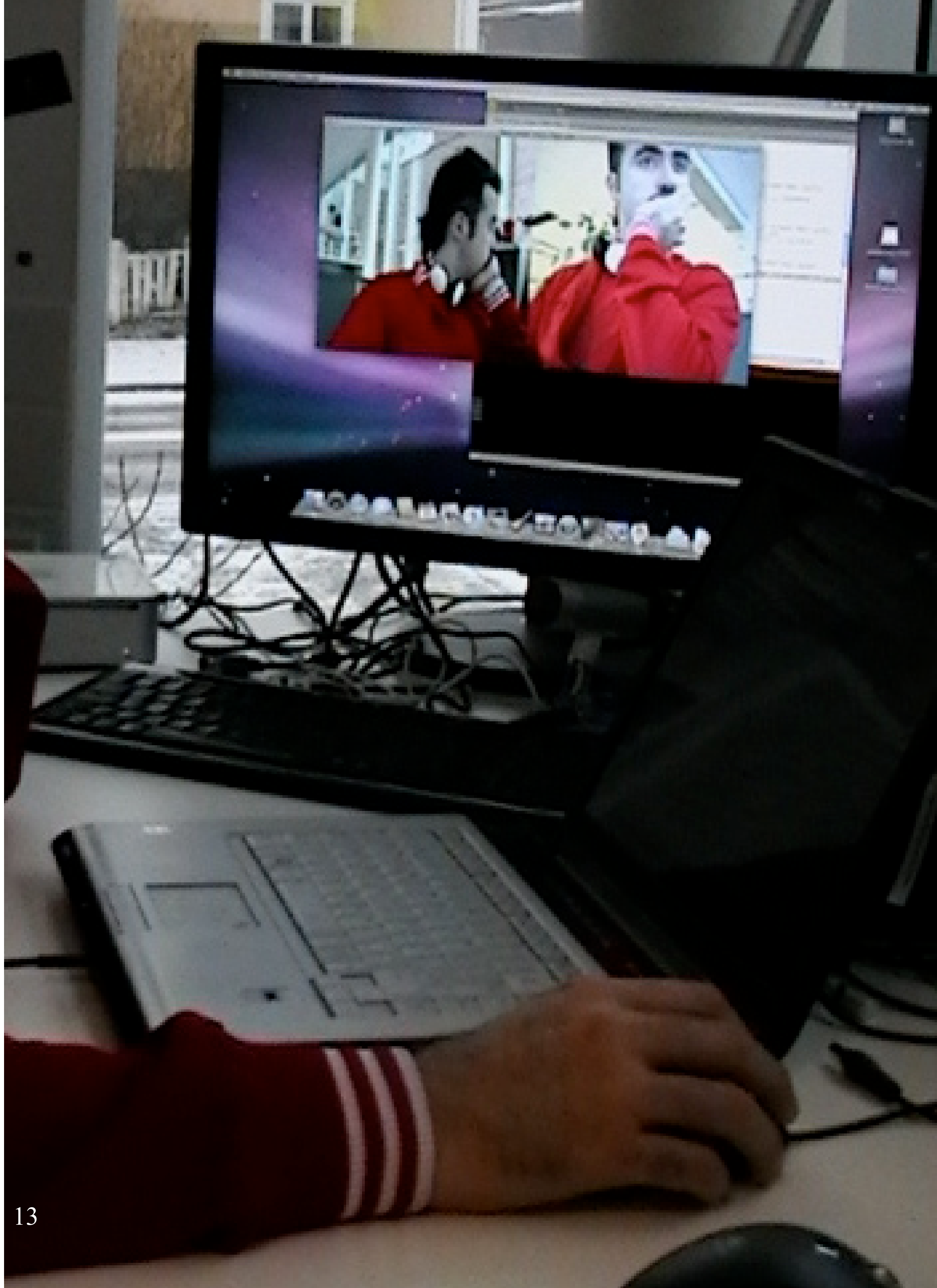
(Image 3 Two video captures in grayscale see CODE 3 "Fixed Color")

Make the sensors work:

In the installation we have two sensors that will trigger the video “ghost” but if no one is sited on the sensor, then we will use another video capture from a second camera to show a different capture. The following code shows how to setup two cameras in processing.

Code Sample #4:

```
//CODE 4 “Using Two Cameras”
import processing.video.*;
Capture mycam;
Capture mycam2;

void setup() {
  size(1280, 480, P3D);
  mycam = new Capture(this,640, 480,””,25);
  mycam.settings();
  mycam2 = new Capture(this,640, 480,””,25);
  mycam2.settings();
}

void captureEvent(Capture mycam) {
  mycam.read();
}

void draw() {
  background(0);
  image(mycam,0,0,width/2,height);
  image(mycam2,width/2,0,width/2,height);
}
```



(Working with the video effects)

How did it work during the party?

The Installation had some specific elements that were specifically setup for that day. E.g. Arduino board with 2 sensors attached and a second camera.

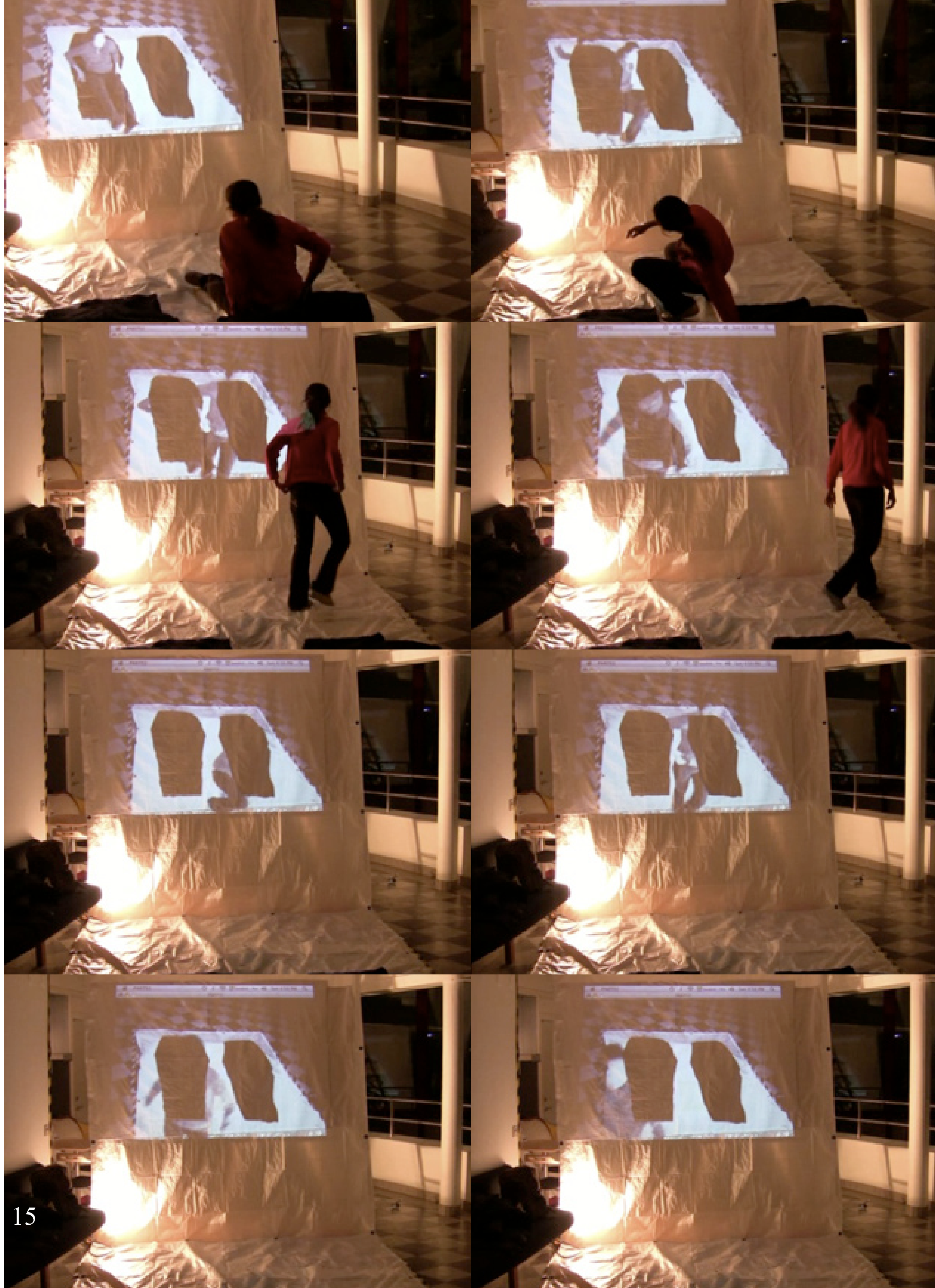
Previously we modified the code in order to make it accessible from any computer that has a webcam. But here is the original code including the Arduino board, the two sensors and the extra camera.

Notice that to use this code properly, you would need to have all the elements we mentioned before connected to your computer. Also notice having 2 cameras working together with Processing is a difficult task, for this particular Installation we used two iSight cameras working fine, but we had problems using other types of cameras.

Code Sample #5 Tale of two tombs (original):

```
//Tale of Two Tombs
//Project by Ruedee Sarawutpaiboon, Sotiris Kotronias, Camille Moussete and
//Jose Ledon
//Umea Institute of Design 2008
//For this code to run properly is necessary to have an Arduino Board with 2
//sensors connected to the PC
//Also to have 2 Cameras connected to the PC (We used 2 iSights)
//If you just want to try the ghost effect with only one camera, please refer to
//the file ghost.pde (Code Sample #2)

import processing.video.*;
import processing.serial.*;
import cc.arduino.*;
PImage myImage = createImage(800, 600, RGB);
PImage myImage1 = createImage(800, 600, RGB);
PImage[] moreImages = new PImage[64];
int counter = 0 ;
int delayCounter = 0;
int delayInterval = 1 ;
int sensorVal1;
int sensorVal2;

Arduino arduino;
Capture myCapture;//Camera with ghost effect
Capture myCapture1;//Dance floor camera
void setup() {
 size(640, 480);
 frameRate(15);
 // setup arduino
 arduino = new Arduino(this, Arduino.list()[0], 115200);
 myCapture = new Capture(this, 640, 480,"", 12);
 myCapture.settings
 myCapture1 = new Capture(this, 640, 480,"", 12);
 myCapture1.settings();
}

void captureEvent(Capture c) {
 if (myCapture.available() == true){
  // init all images to black
  if (counter == 0){
   println("capture Event");
   for ( int k = moreImages.length - 1; k >= 0;  k--){
    moreImages[k] =myImage;
   }
  }
  //println("capturing");
  if (delayCounter >=  delayInterval){
   // shift all the images in the array
   for ( int k = moreImages.length - 1; k > 0;  k--){
    moreImages[k] = moreImages[k-1];
   }
   // last one to update
   moreImages[0] = myImage;
   // clear the interval counter
   delayCounter = 0;
  }
  // capture latest frame/image
  myCapture.read();
  myImage = myCapture.get();
  delayCounter++;
 }
 else if (myCapture1.available() == true){
  myCapture1.read();
  myImage1 = myCapture1.get();
 }
}
//Mode for Ghost program running from Camera 2 (myCapture)
void modeA(){
 background(0);
 PImage test = moreImages[0].get();
//moreImages[0] = realtime image, change the number to have a delayed image
```

```
  myImage.mask(moreImages[50]);
//Increse the number [50] to have more delayed for the ghost
  image(test, 0, 0); //if you use “myImage” insted of “test” = flickering
  filter(GRAY); //convert both images to Grayscale
}

//Mode for normal video from Camera 2  (myCapture)
void modeB(){
  background(0);
  image(myImage, 0, 0);
}
//Mode for normal video from Camera 1 (myCapture1)
void modeC(){
  background(0);
  image(myImage1, 0, 0);
}

//Code for controlling the Arduino Sensors
//2 sensors pressed = Switch from Camera 1 to Camera 2 + Ghost effect
//1 sensor pressed = Switch from Camera 1 to Camera 2
//No sensor pressed = Show video capture from Camera 1
void draw(){
  sensorVal1 = arduino.analogRead(2);
  sensorVal2 = arduino.analogRead(4);
  println(sensorVal1 + “ “ + sensorVal2);
  // if both sensors activated
  // show ghost
  if((sensorVal1 > 500 && sensorVal2 > 500) || keyPressed == true){//key-
Pressed for testing with keyboard
    modeA();
  }
  // if only one sensor of the two is activated
  // display camera 2
  else if(((sensorVal1 > 500) || (sensorVal2 > 500)) || mousePressed == true)
{//mousePressed for testing with mouse
    modeB();
  }
  else{
    // if no sensor is activated
    // display camera 1
    background(0);
    modeC();
  }
  counter++;
}
```


(Photo sequence of the ghost video effect)

Conclusion:

1) The installation was meant to work during a Halloween party where the lighting conditions were inadequate, since we didn't take on count the illumination factor, it was difficult to see what was happening in the video display, reason why the whole idea lose importance.

2) Prepare to be flexible and think of everything in one whole picture. Even though, we planned for one installation but for the limited time that we had, we split our work in 2 parts, hardware and software part. Therefore we had to also made some changes along the way to suited the time and technology that we had.

3) Even thought the conditions weren't the optimum, technically speaking, we were able to have working together Arduino and two sensors (flexibility and pressure) with Processing. The most challenging task was to use Processing together with two video capture devices. To have all this equipment working in harmony was the best achievement of this project.