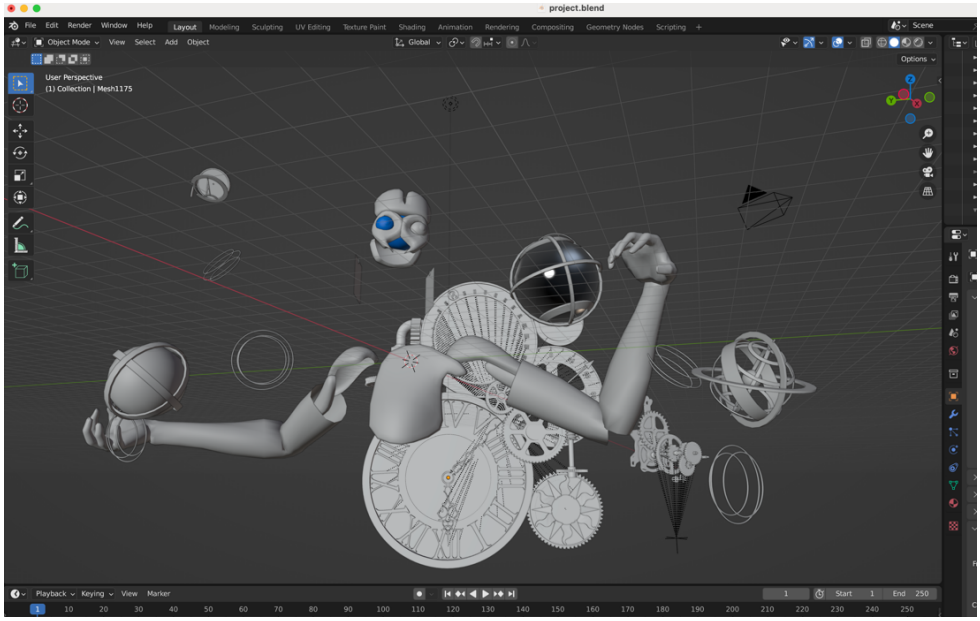
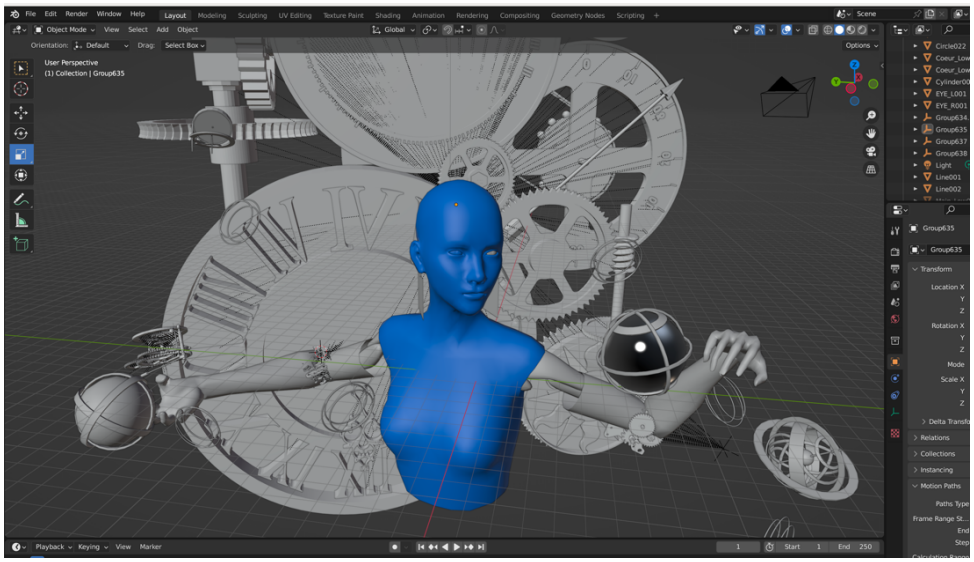
Readme

Designer： YeZhang（22008621）& Yaqi Han（22014011）

Setting up the scene

We discussed the project topic and we decided to make the relationship between machine and human as the topic. We intended to create an interactive project using a combination of Arduino and UE5. We first decided to model in Blender to build the scene. The main body of the model shows the state of human being eroded by the machine.





We had a problem importing the model into UE usage. We tried obj and fbx formats directly exported by Blender, and UE showed an import error. We finally downloaded the Datasmith plugin for UE5 to import the model. But plugins importing fbx format will cause the model parts to be split apart. The model has a lot of difficulty in changing the material and rendering.

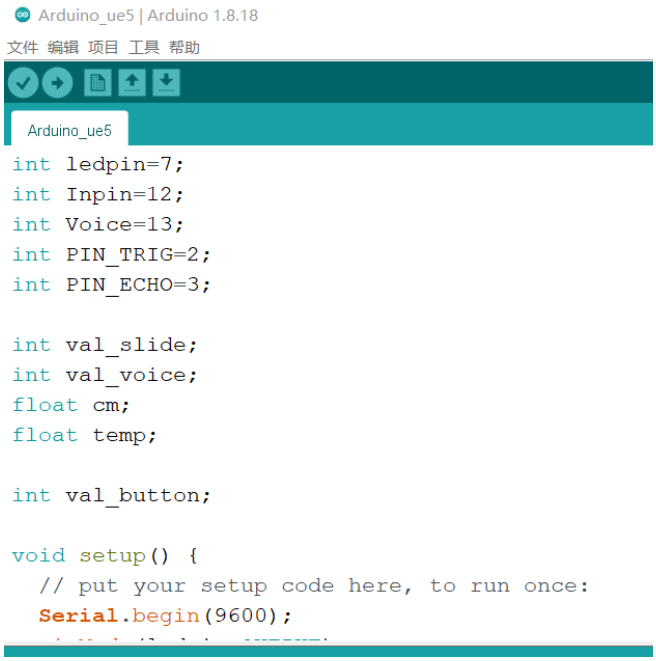
Understanding and learning to use the UE5 SerialCOM plug-in

In the class, the teacher introduced the UE5 can use SerialCOM plug-in to link Arduino, so we this assignment to the platform as a piggyback platform, I first carried out the study of this plug-in, which uses several platforms to learn:

https://github.com/videofeedback/Unreal_Engine_SerialCOM_Plugin

https://www.youtube.com/watch?v=EuMarb76dG0

Arduino connection and related code writing

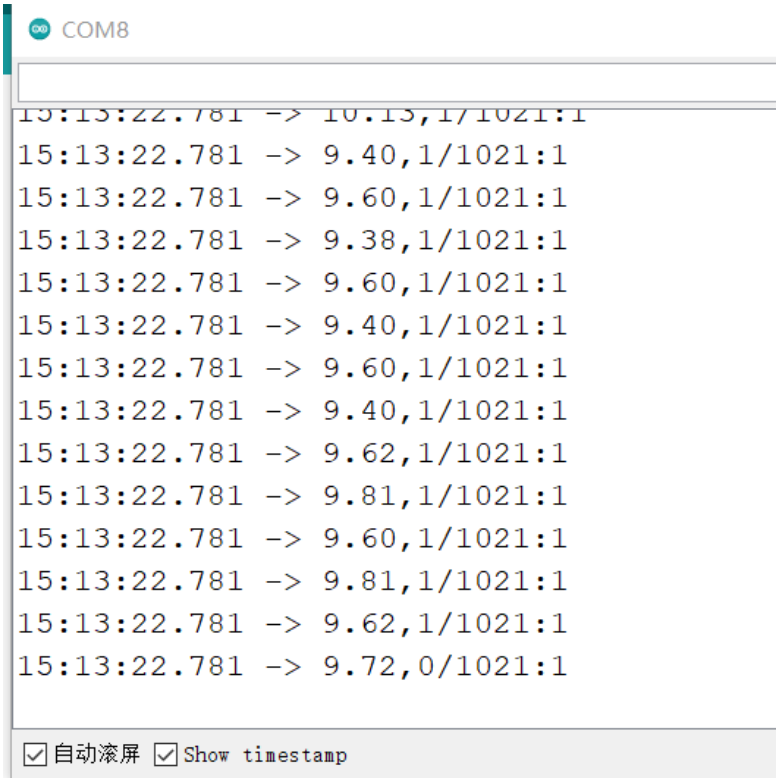


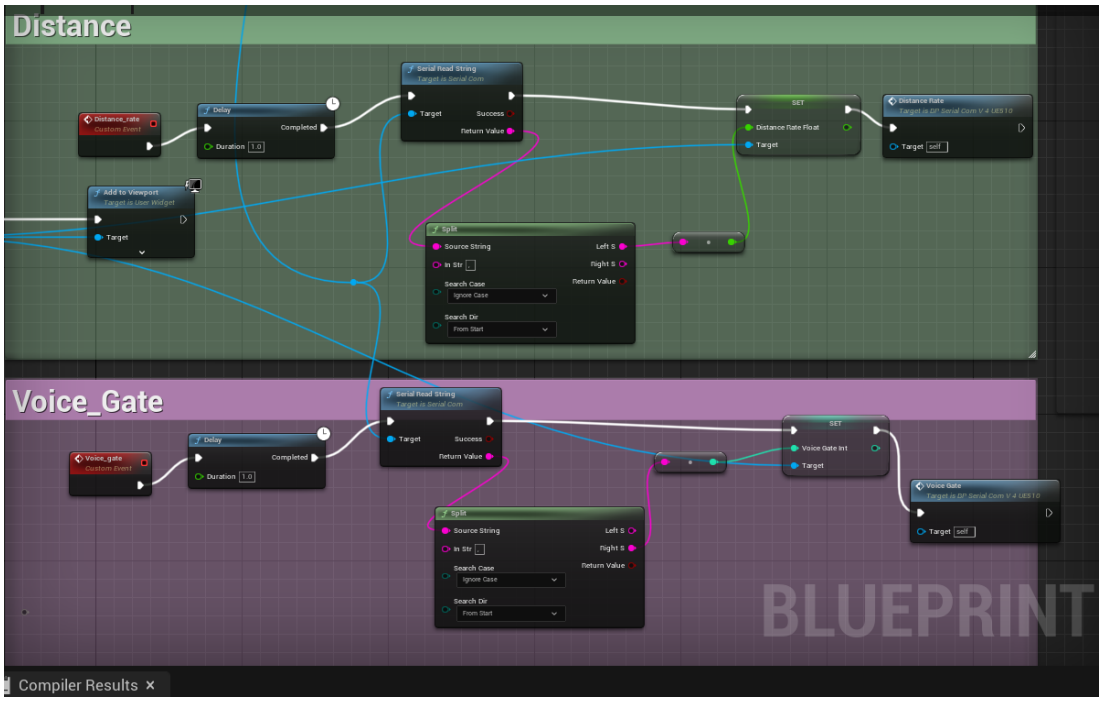
```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(ledpin,OUTPUT);
  pinMode(Inpin,INPUT);
  pinMode(Voice,INPUT);


  pinMode(PIN_TRIG, OUTPUT);
  pinMode(PIN_ECHO, INPUT);

}

void loop() {
  // put your main code here, to run repeatedly:
```

```
void loop() {
  // put your main code here, to run repeatedly:
val_button=digitalRead(Inpin);
val_voice=digitalRead(Voice);
val_slide=analogRead(A0);



  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(2);
  digitalWrite(PIN_TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);
 temp = float(pulseIn(PIN_ECHO, HIGH));
  cm = (temp * 17 )/1000;


  // Serial.print("Echo = ");
  // Serial.println(temp);
  // Serial.print(",  Distance = ");

  Serial.print(cm);
  Serial.print(",");
  //delay(100);

  Serial.print(val_voice);
  Serial.print("/");
  Serial.print(val_slide);
  Serial.print(":");
  Serial.println(val_button);
```

We found that Arduino can only transfer one data when transferring data. So we output all the data in Arduino in the form of one line of data with interval symbols

connected in between. When reading in UE5, we use the function split to split the data in the transferred line and finally read the data.

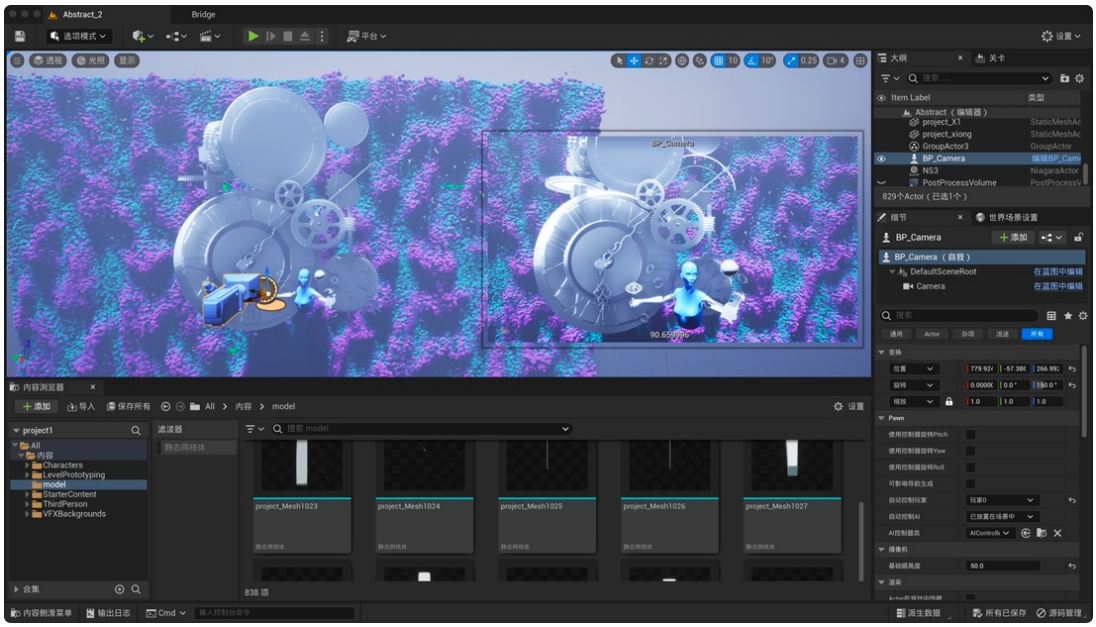




But we do not know the type of data transferred in, so we debug the data input in, because the Arduino data will have the problem of instability, we carry out the

Delay statement to optimize the operation.

Optimization model:

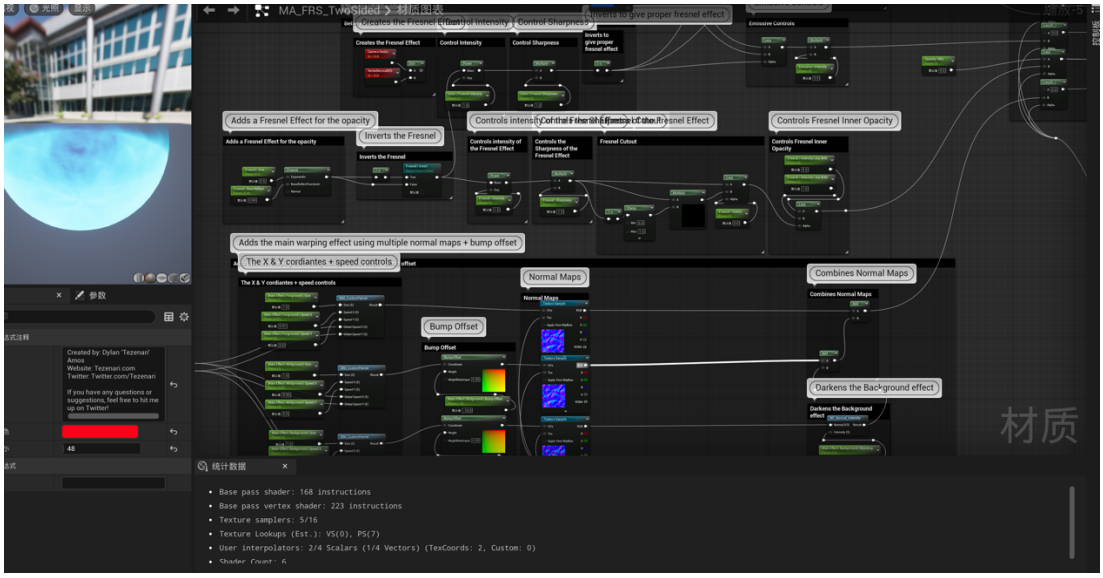


We found that the UE5 command to box multiple models during model adjustment did not have a way to select the small parts inside the model. Also, because the center of the model was grouped, the axis position was shifted, which led to misaligned motion when adding components and blueprints to some models. We also found that in UE5, when the model was converted from a static mesh to a blueprint actor, it lost its materials and original positions. We had to adjust their positions and materials repeatedly.

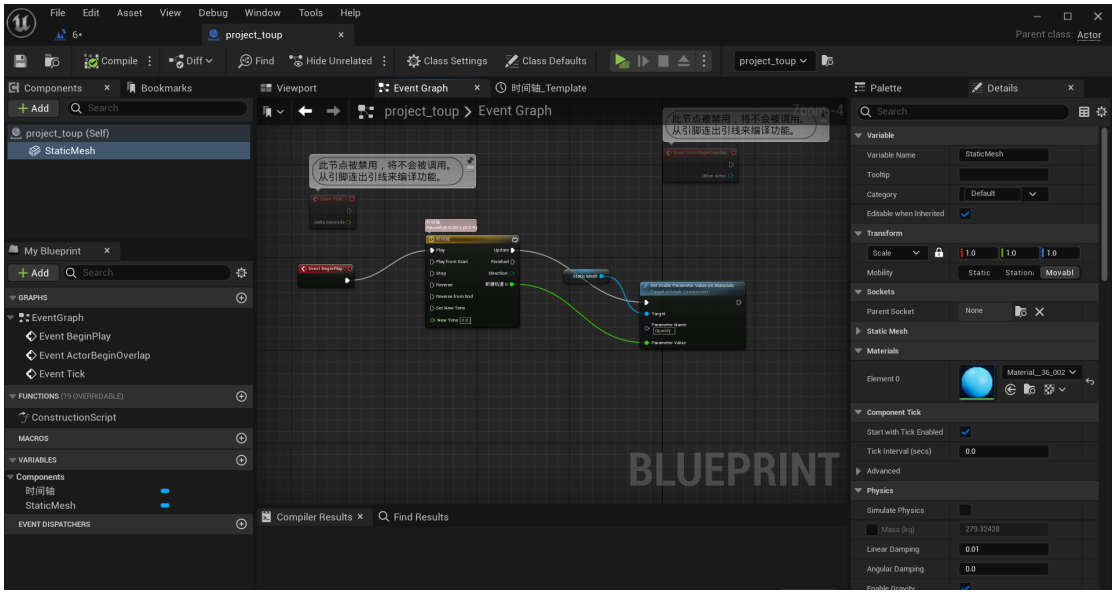
Model rendering:

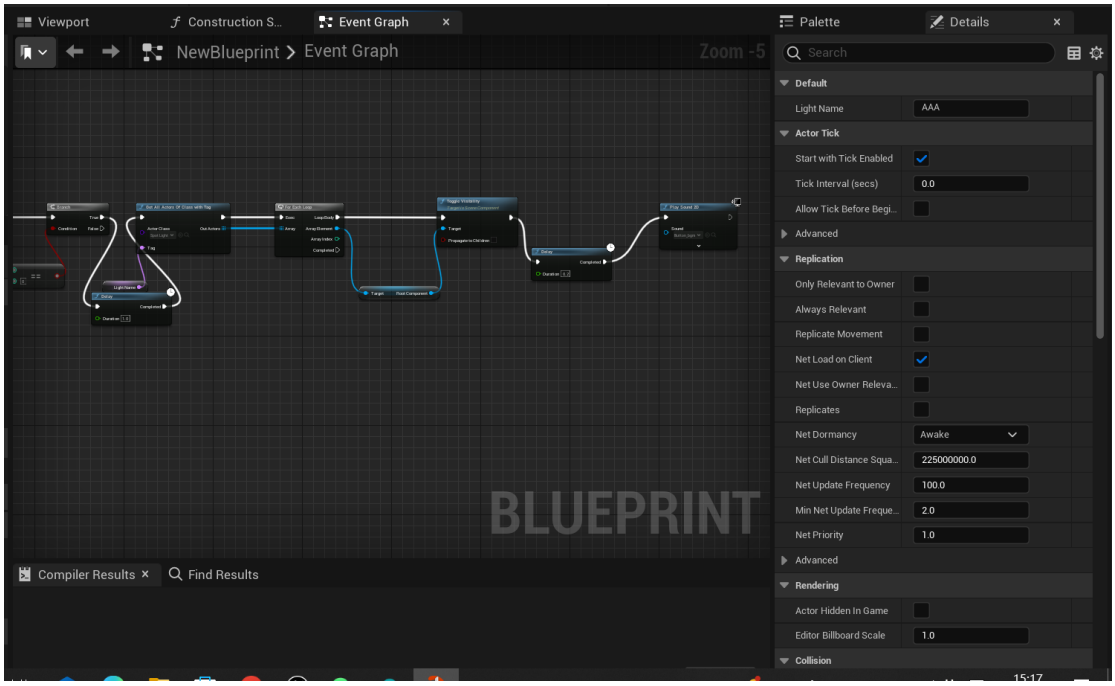
To give the model a more mechanical feel we connected a lot of dynamic blueprint materials, using different texture maps and materials. One of the references is the flow material from Twitter: Twitter.com/Tezenari to achieve the

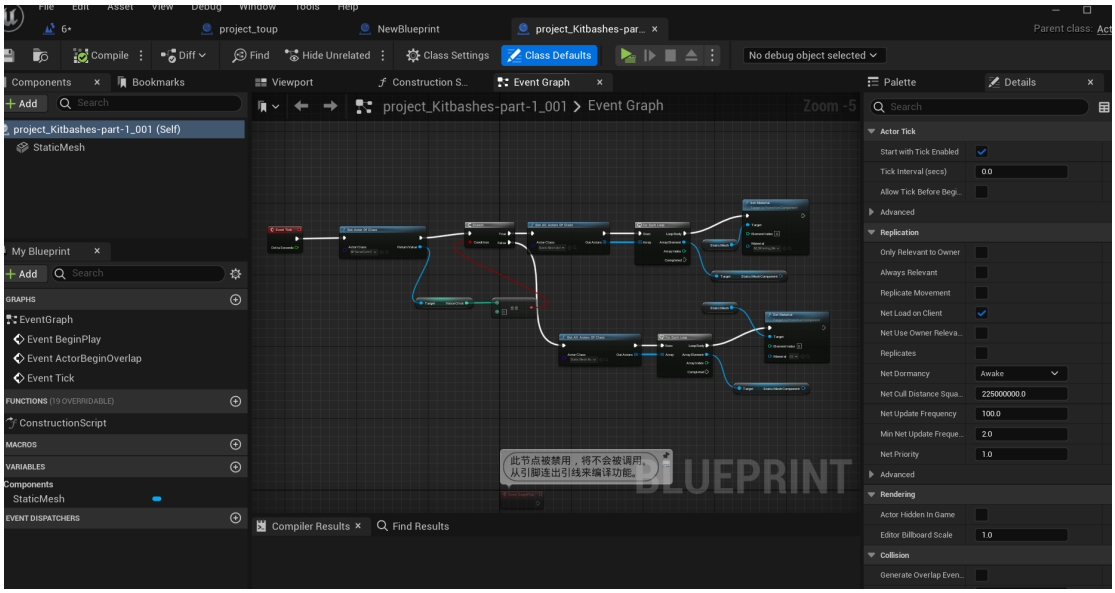
flow material effect.



We use the timeline to connect to the material, and use click events and blueprints to connect to the sensor values to achieve changes in the material and transparency of the object.

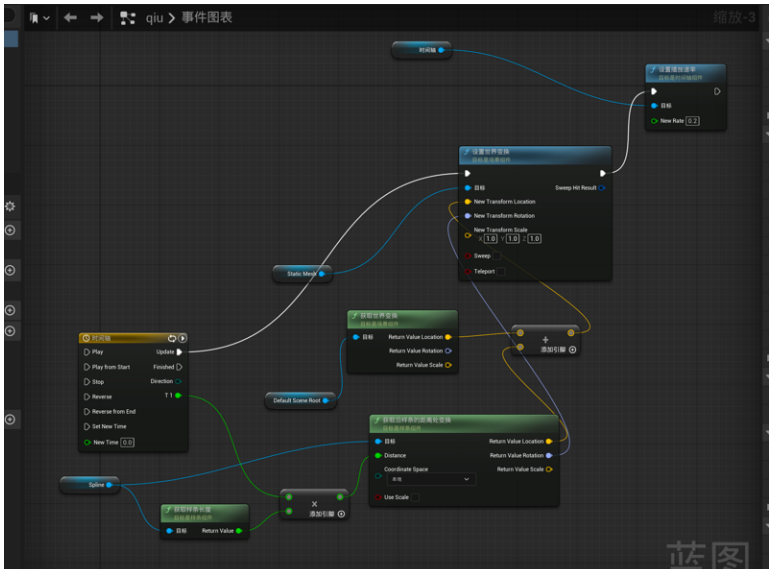


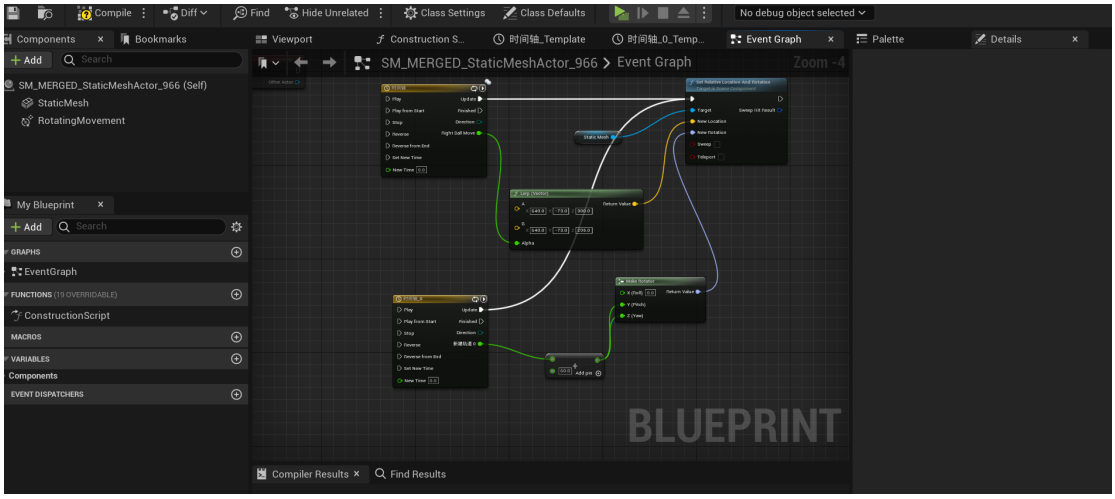




Object hover：

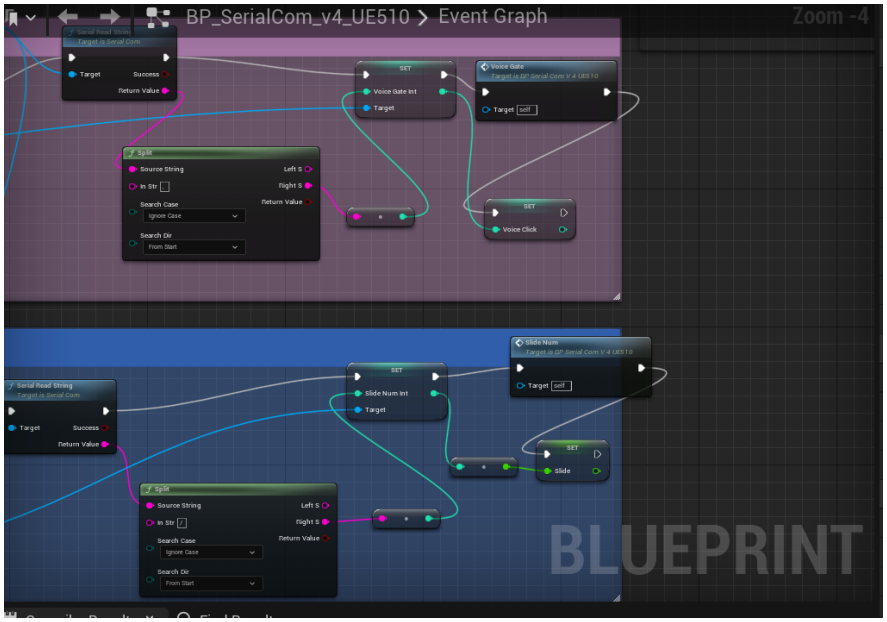
We tried two ways to connect the object hover and finally chose to use the timeline control as a method to control the hover of the object.

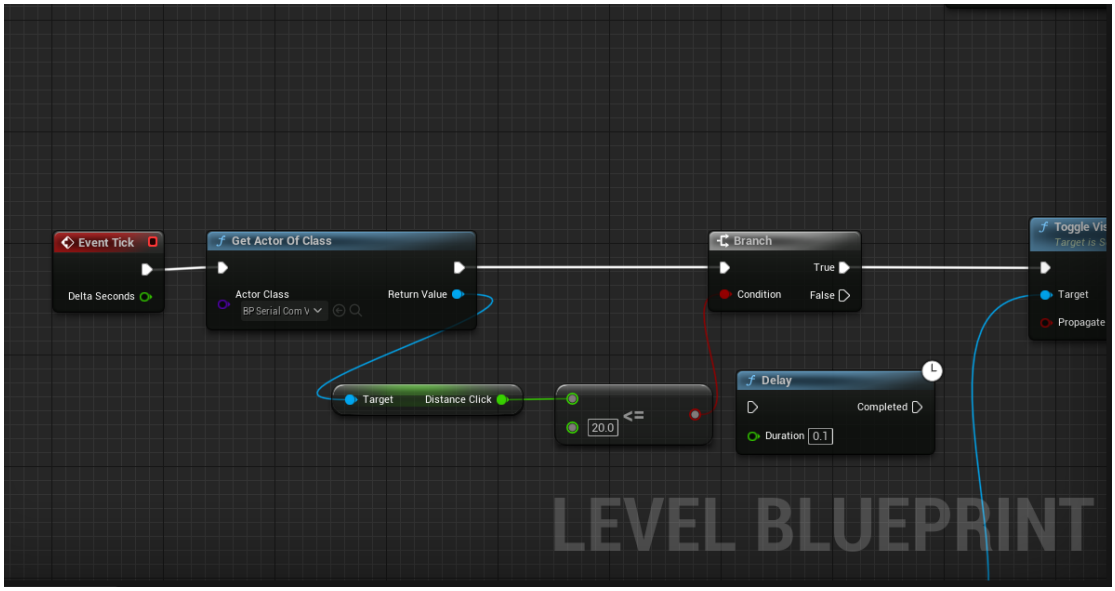


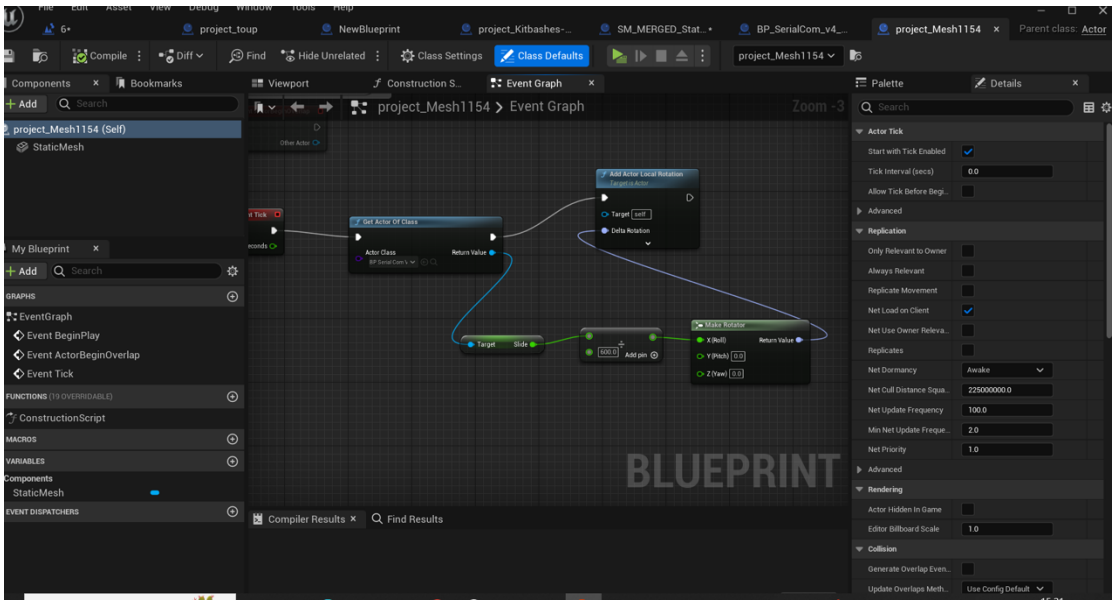


Data using:

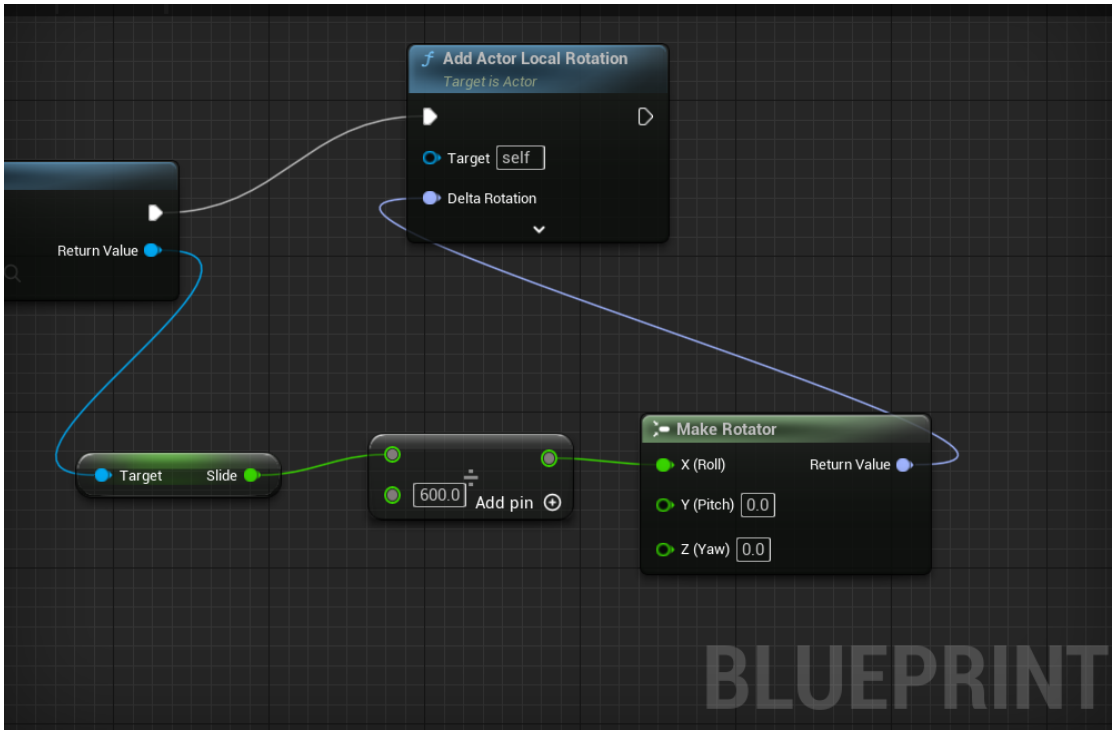
For the data transferred in from Arduino, we didn't know how to use it in other blueprints at the beginning, and the tutorials we searched online were all about connecting the level blueprints. But before referencing, you must define the assignment in the Bp_Serialcom blueprint before it can be referenced.


BP_SerialCom_v4_UE510 > Event Graph
BLUEPRINT

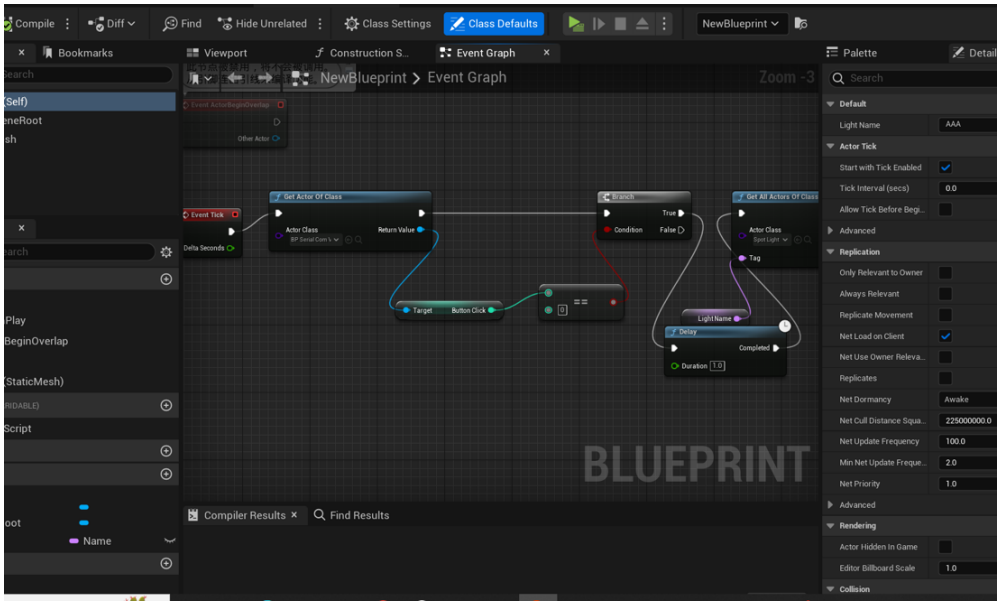

Event Tick
Delta Seconds
Get Actor Of Class
Actor Class
Return Value
Branch
True
Condition
False
Target
Distance Click
20.0
Delay
Completed
Duration
0.1
LEVEL BLUEPRINT

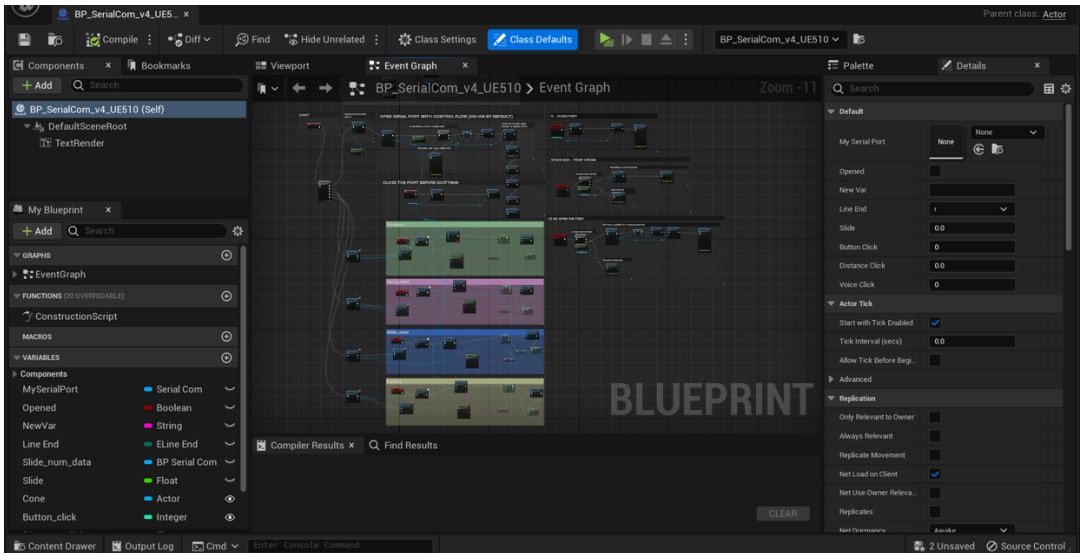

project_Mesh1154 > Event Graph
BLUEPRINT

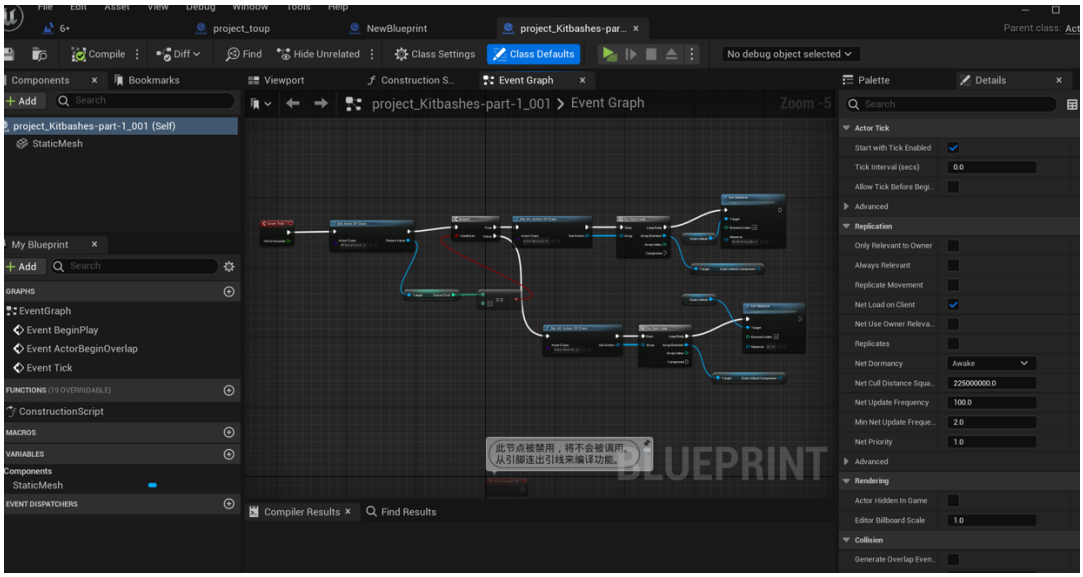
The angle of the clock cannot be controlled: we originally wanted to set the angle of rotation of the clock using the angle of rotation, but we encountered problems in the use of the data, we did not have a way to solve the accuracy of the data, we did not have a way to use the control once the rotation variable, so we ended up using a sliding rheostat to control the speed of rotation of the clock.

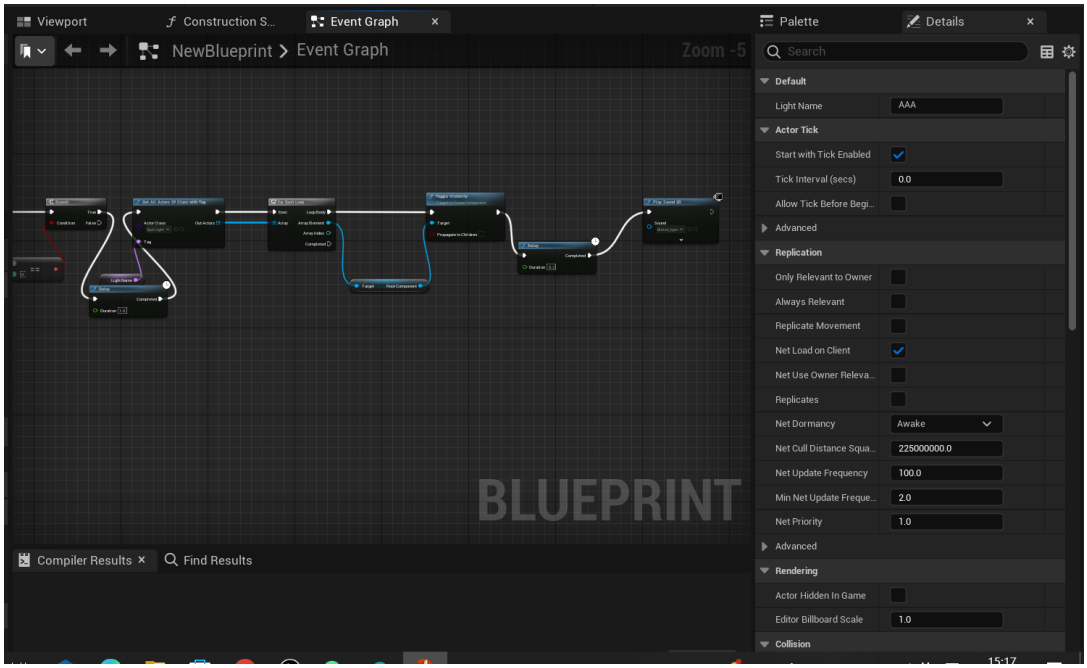


Finally we combine the tested blueprint with our asset blueprint.




BP_SerialCom_v4_UE5...
Parent class: Actor
Compile
Diff
Find
Hide Unrelated
Class Settings
Class Defaults
BP_SerialCom_v4_UE510
Components
Bookmarks
Viewport
Event Graph
BP_SerialCom_v4_UE510 > Event Graph
Zoom -11
Palette
Details
BP_SerialCom_v4_UE510 (Self)
DefaultSceneRoot
TextRender
My Blueprint
EventGraph
ConstructionScript
MACROS
VARIABLES
MySerialPort
Opened
NewVar
Line End
Slide_num_data
Slide
Cone
Button_click
Serial Com
Boolean
String
ELine End
BP Serial Com
Float
Actor
Integer
BLUEPRINT
Compiler Results
Find Results
CLEAR
Content Drawer
Output Log
Cmd
Enter Console Command
2 Unsaved
Source Control

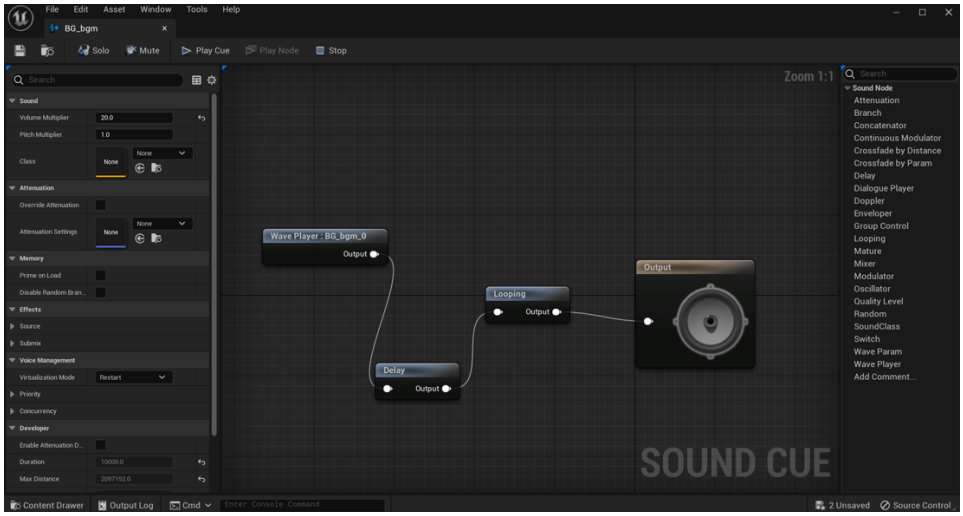

project_toup
NewBlueprint
project_Kitbashes-par...
Parent class: Act
Compile
Diff
Find
Hide Unrelated
Class Settings
Class Defaults
No debug object selected
Components
Bookmarks
Viewport
Construction S...
Event Graph
project_Kitbashes-part-1_001 > Event Graph
Zoom -5
Palette
Details
project_Kitbashes-part-1_001 (Self)
StaticMesh
My Blueprint
EventGraph
Event BeginPlay
Event ActorBeginOverlap
Event Tick
FUNCTIONS
ConstructionScript
MACROS
VARIABLES
StaticMesh
EVENT DISPATCHERS
此节点被禁用，将不会被调用。
从引脚连出引线来编译功能。
BLUEPRINT
Compiler Results
Find Results
Actor Tick
Start with Tick Enabled
Tick Interval (secs)
Allow Tick Before Begi...
Advanced
Replication
Only Relevant to Owner
Always Relevant
Replicate Movement
Net Load on Client
Net Use Owner Relevа...
Replicates
Net Dormancy
Awake
Net Cull Distance Squa...
225000000.0
Net Update Frequency
100.0
Min Net Update Freque...
2.0
Net Priority
1.0
Rendering
Actor Hidden In Game
Editor Billboard Scale
Collision
Generate Overlap Even...

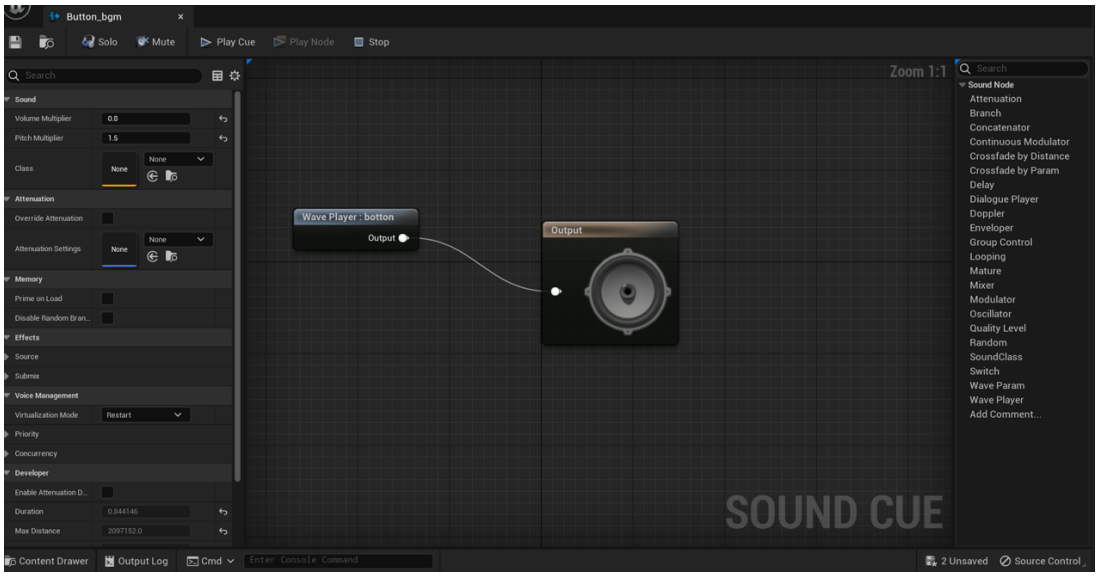

Viewport
Construction S...
Event Graph
NewBlueprint > Event Graph
Zoom -5
Palette
Details
Search
Default
Light Name
AAA
Actor Tick
Start with Tick Enabled
Tick Interval (secs)
0.0
Allow Tick Before Begi...
Advanced
Replication
Only Relevant to Owner
Always Relevant
Replicate Movement
Net Load on Client
Net Use Owner Relevа...
Replicates
Net Dormancy
Awake
Net Cull Distance Squa...
225000000.0
Net Update Frequency
100.0
Min Net Update Freque...
2.0
Net Priority
1.0
Advanced
Rendering
Actor Hidden In Game
Editor Billboard Scale
1.0
Collision
BLUEPRINT
Compiler Results
Find Results

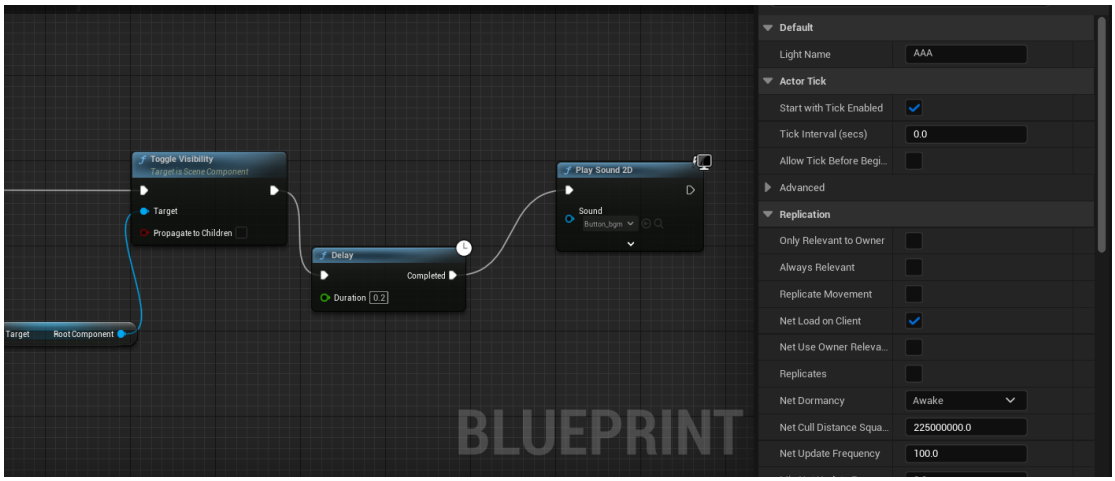
Music:

We searched for some sound sources that could be used as a combination of sensor triggers and sound effects. We used the Kontakt sound sampler in Logic pro to mix the sound effects and connect them to the blueprint. By considering the fluctuating values of distance sensors and potentiometers, we finally chose buttons and sound sensors to control both music effects.

We have a Loop design for background music which we have set up using the SoundCue component and adjusted the pitch and play speed. This component is also used in other sensors but does not use loops, when used in blueprints the music is obtained through the GetSound2D component.





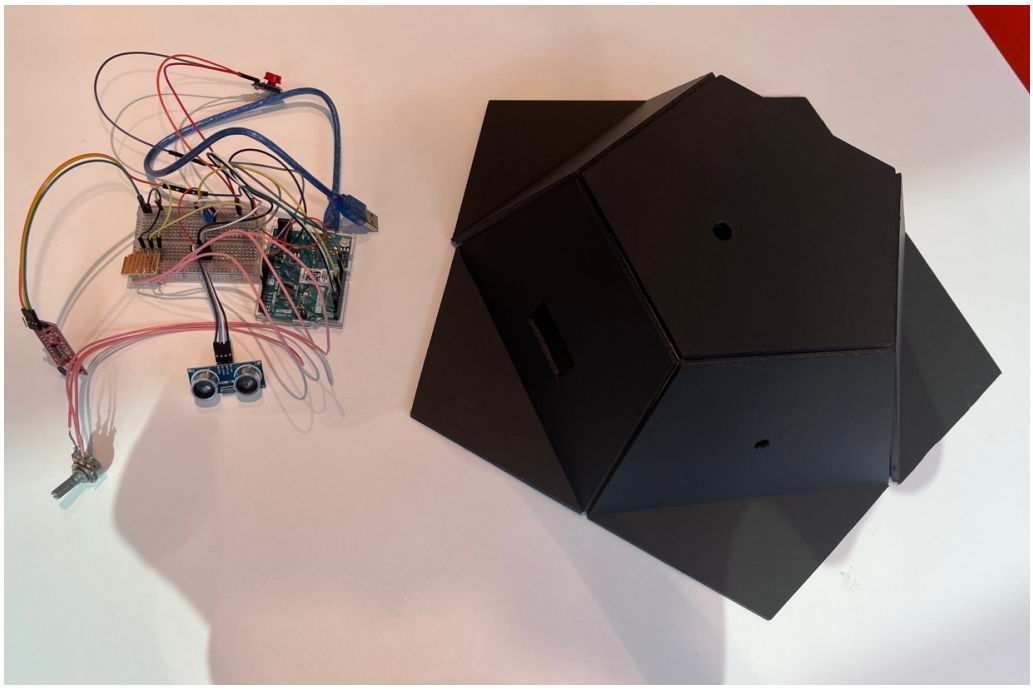


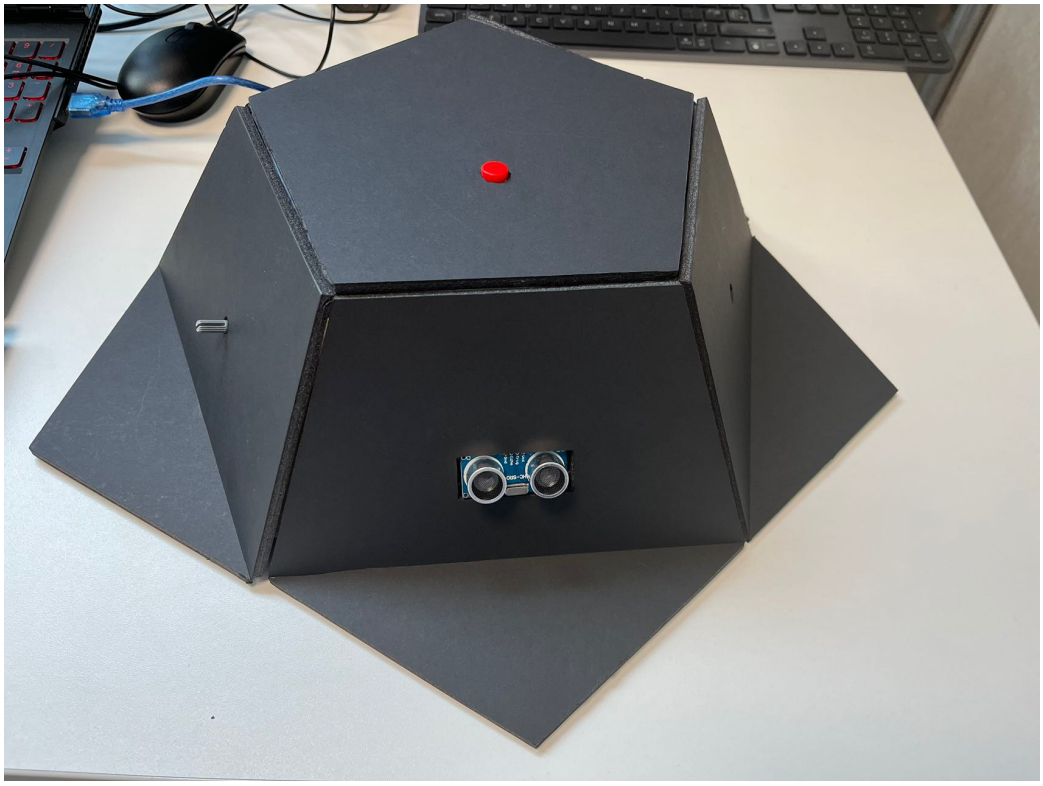
Others:

Optimizing memory: Because our model was split into very small unit models, the computer was not running smoothly enough after a large number of blueprints were connected. With the help of Mr. John, we changed the runtime port to optimize memory. At the same time, on his advice, we merged some of the models that did not need blueprints and changed the real-time rendering accuracy of the models. We were able to save the model materials and make it run more smoothly.

Design and construction:

We designed a pentagonal housing for our external control unit so that each side of the pentagon can be connected to a sensor for control.





Running effects:

We will use four sensors to control all interactive shifts.

Distance sensor: when the human body approaches overall lights starts to activate and the interaction lights to run. At the same time, a piece of music will play.

Sound sensor: when activated, shows the transition from human to machine, triggering sounds effect during the transition. The colour of the muscles and blood vessels connecting the arms will change from red to a mechanically connected state; the sound is not activated and the colour of the muscles themselves will be displayed.

A sliding rheostat controls the speed of rotation of the clock hand behind the semi-robot; as the value of the sliding rheostat gets larger, the speed of rotation gets faster, and conversely, as the sliding value gets smaller, the speed of rotation gets smaller.

Buttons, which control the illumination of the two spotlights around the body, illuminate the body when the button is pressed and trigger sound effects on the stage.