

steli

Laura Bordin + Lorenzo Cercelletta + Valentina Venza

# experience



**Steli is a collaborative musical instrument, that should make cooperate people in different spaces.**

**It can be played by everyone: amateurs, musicians or even the wind**

**The system suggests how to play in armony with the accompaniment and viceversa and returns the whole “composition” in real time in both places.**

context

Its features make it perfect for each couple of places that are one in front of the other not too distant, not too close.

For instance the two sides of the Canale della Giudecca, in Venice.

It works all day long, but has its perfect moment in the dark, when the suggestions will be more visible.

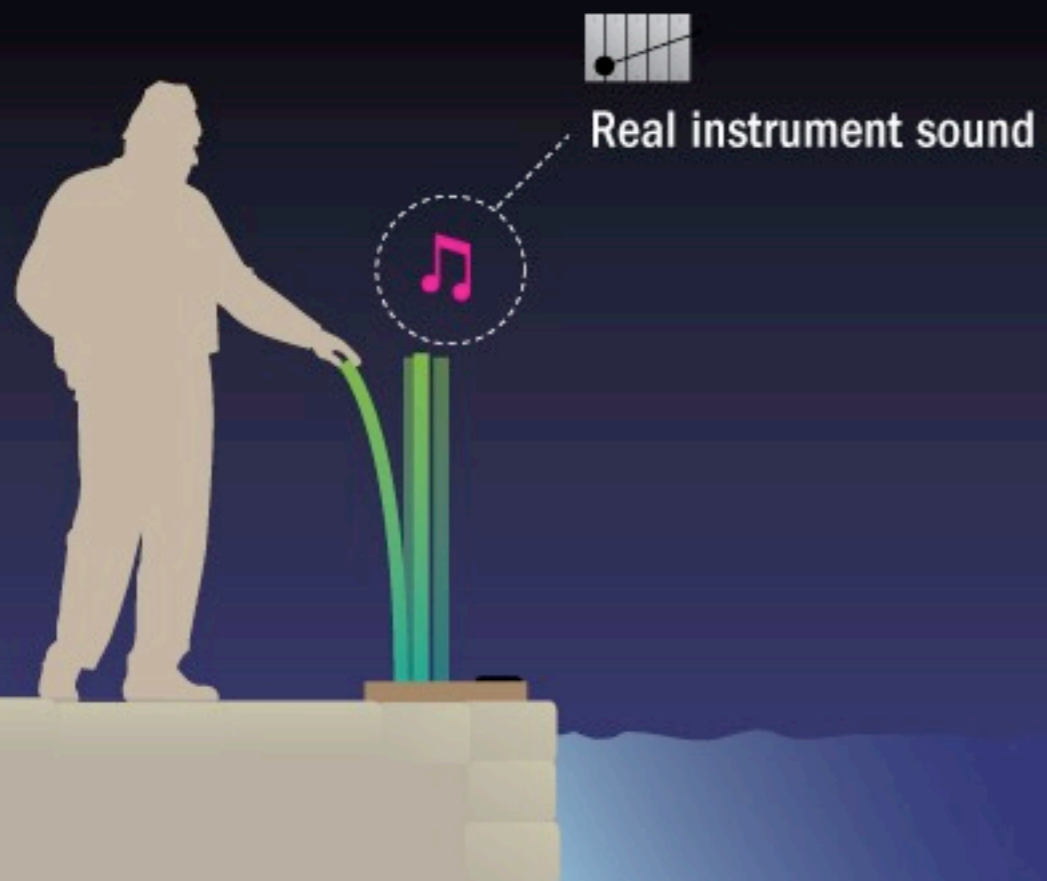
# interaction



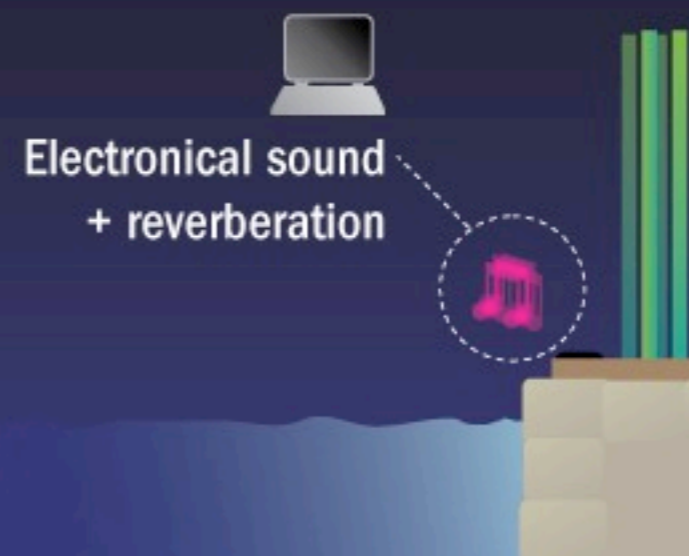
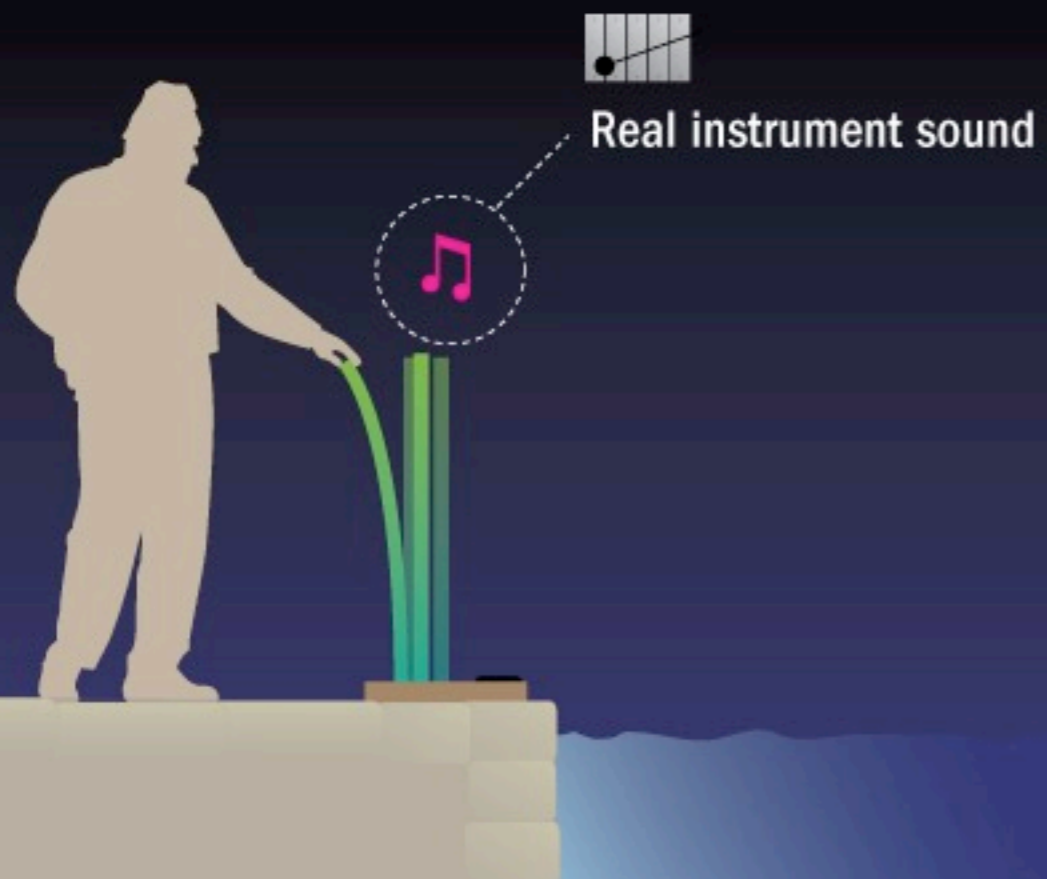
# interaction



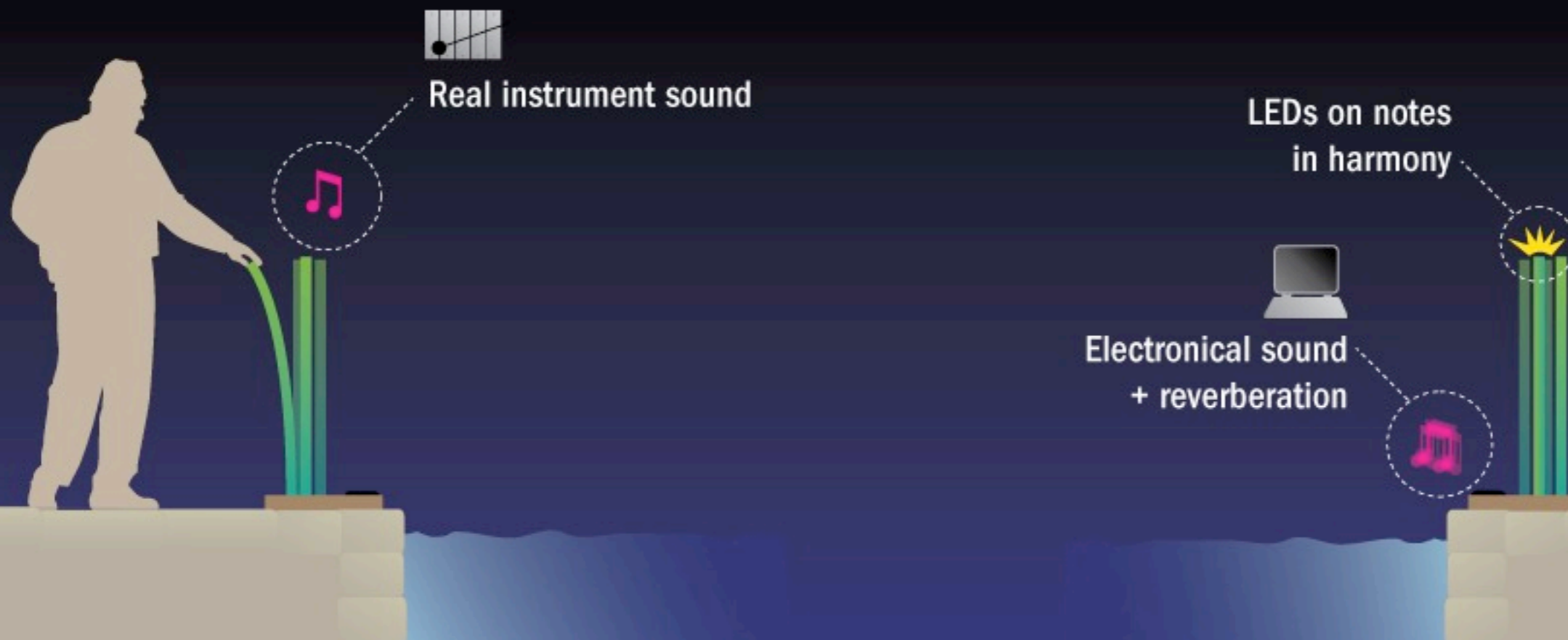
# interaction



# interaction

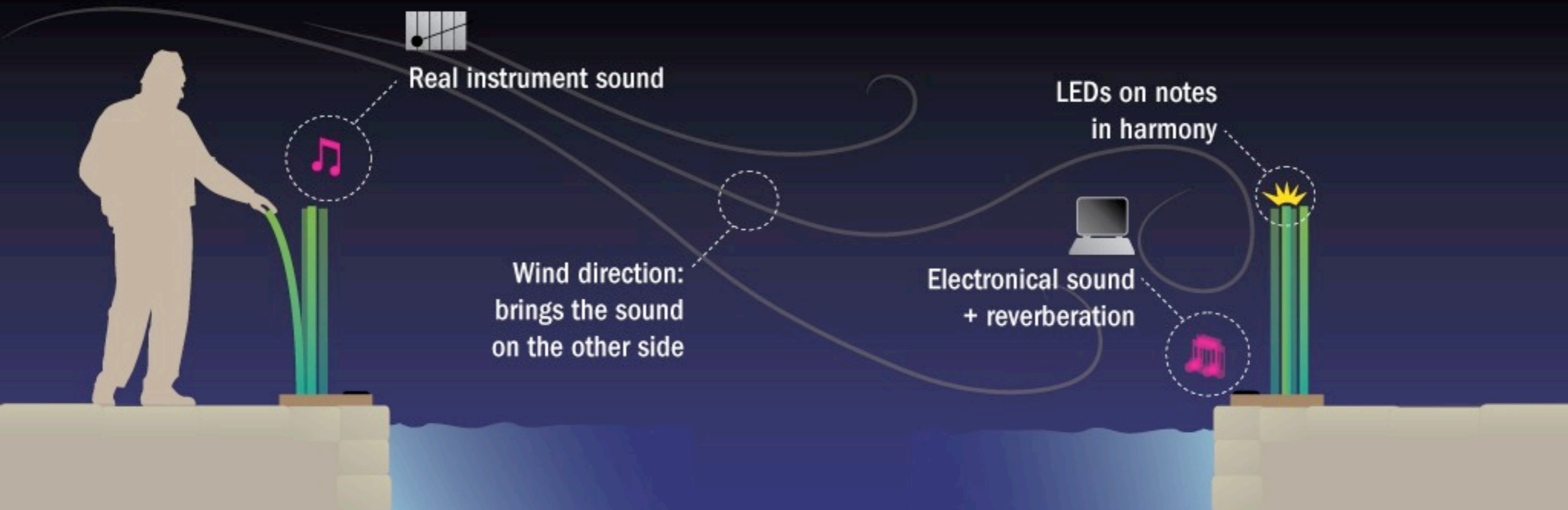


# interaction

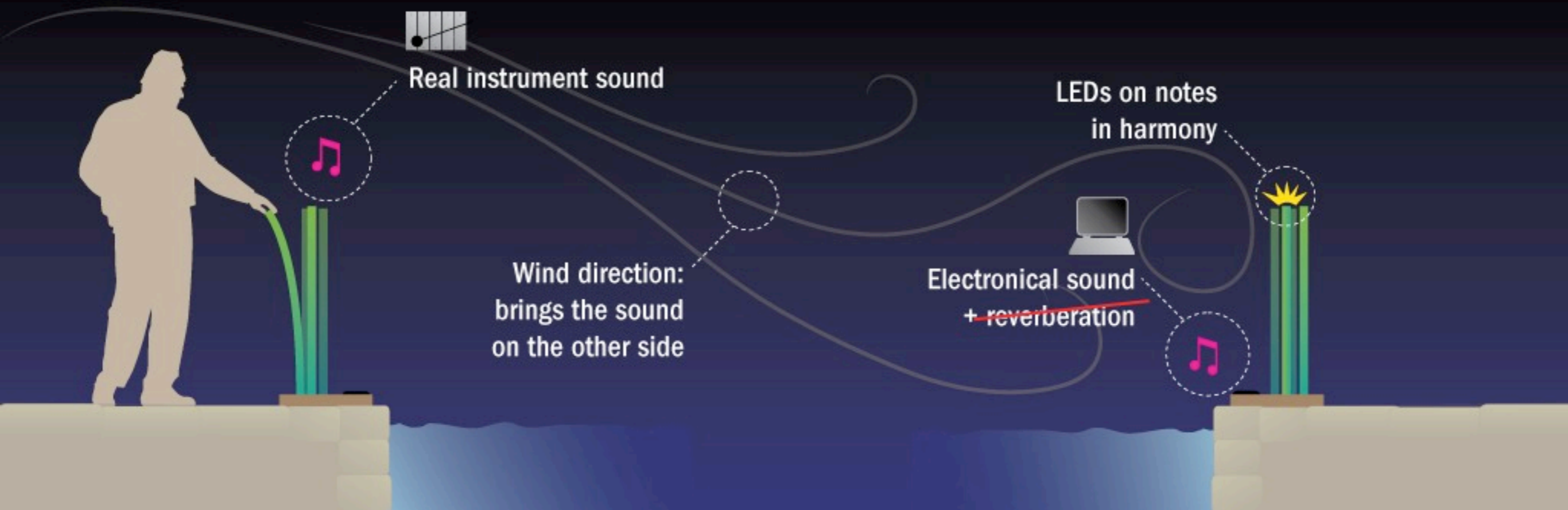




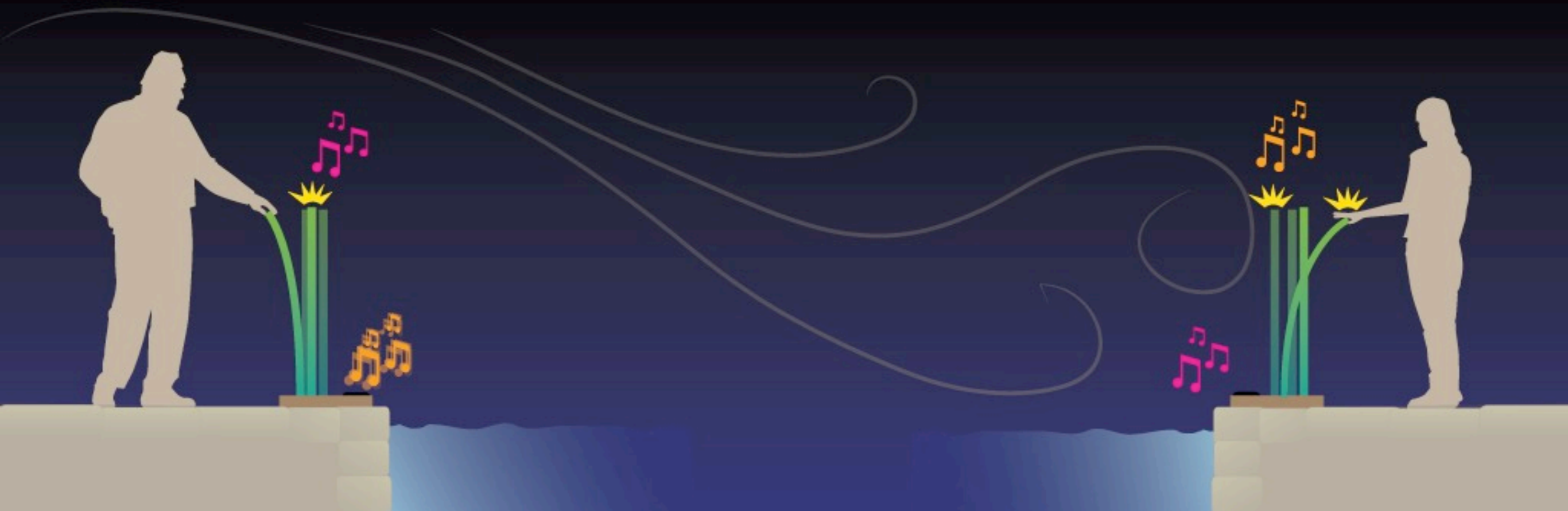
# interaction



# interaction



# interaction



# interaction

The installation can be played by people or just by the wind, at day or at night.





**design**

Our aim is to give the affordance for the right gesture.

We thought about grass, that makes you want to pass your fingers through it.

We built structures that not represent literally grass blades, but that could evoke them.

# sounds



Musical instrument



Place A: melody  
Glockenspiel



Musical instrument



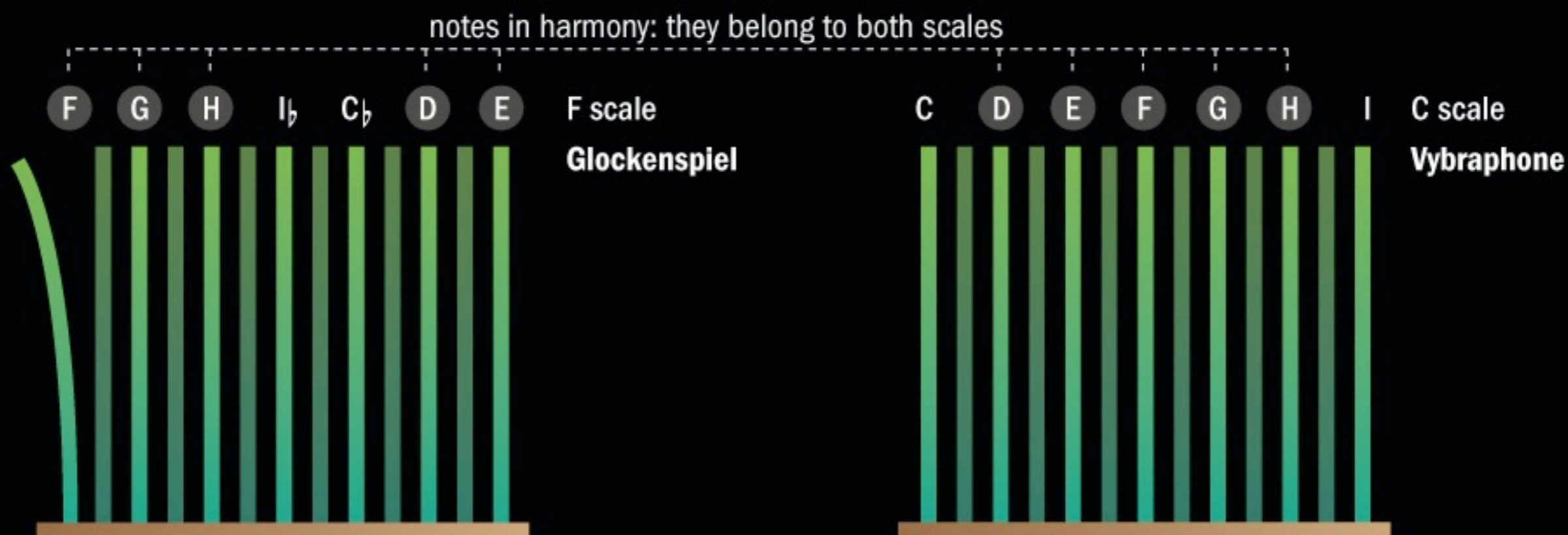
Place B: accompaniment  
Vybraphone



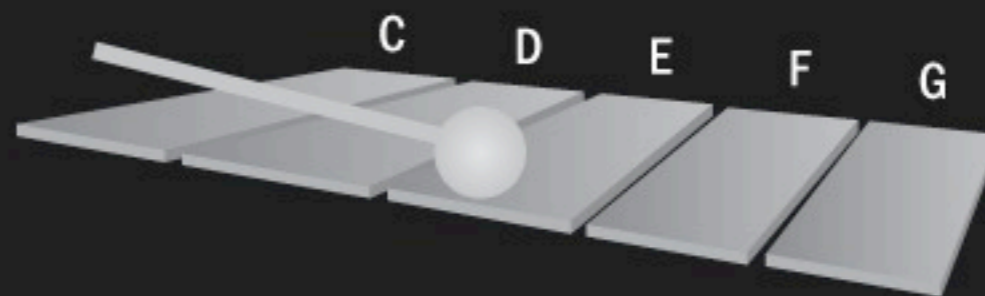
# sounds

Each blade plays one note.

The two instruments play two scales which are not in perfect harmony.

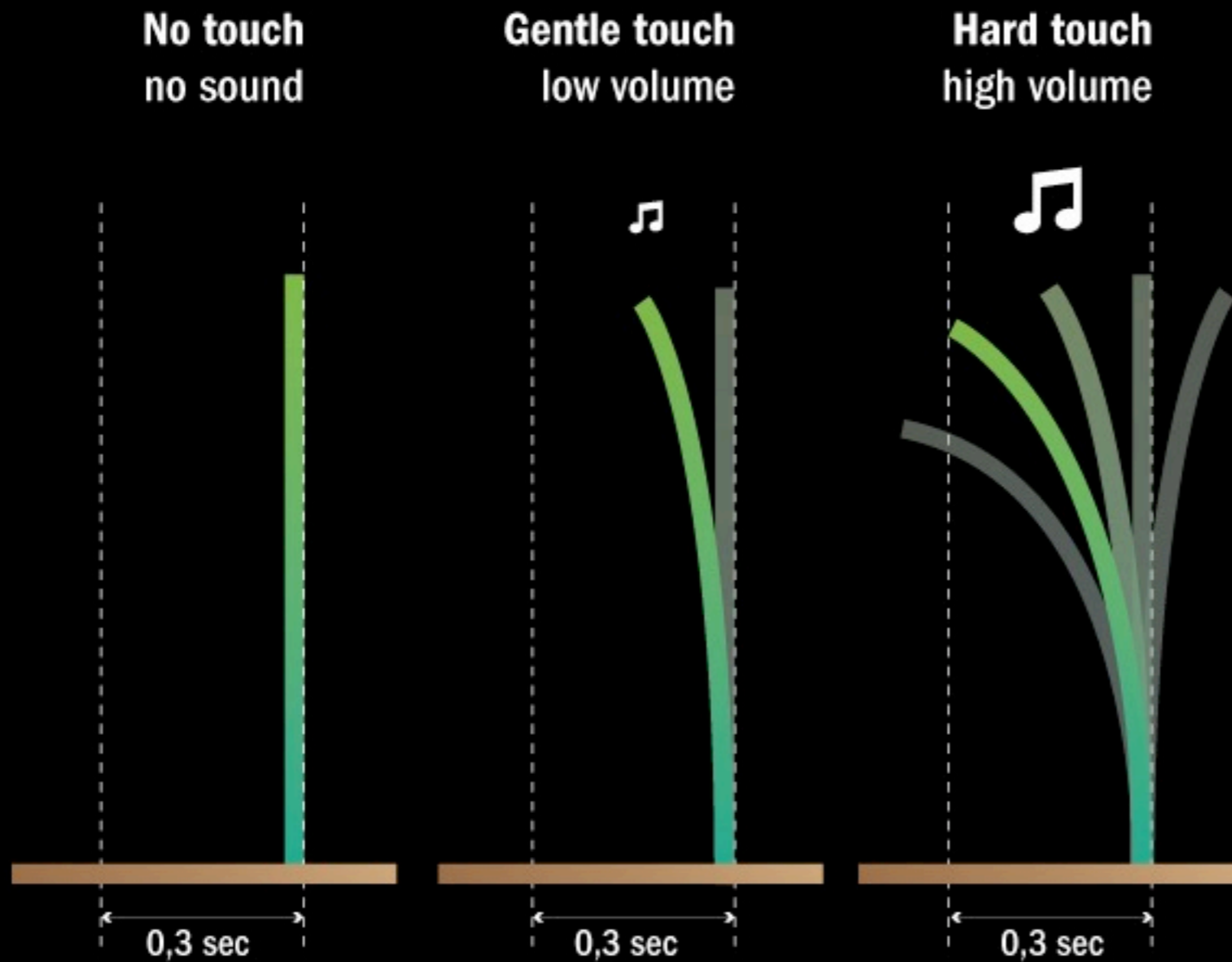


*Just like in percussive instruments:  
one plate plays one note.*

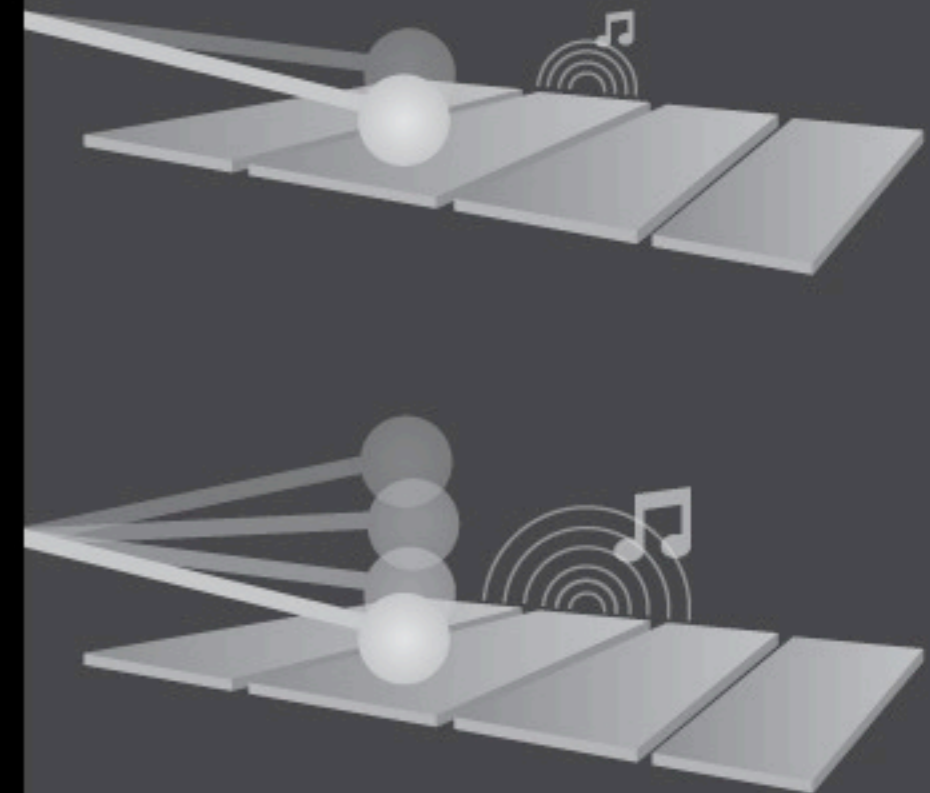


# sounds

Volume is determined by bending / time



*Just like in percussive instruments...*

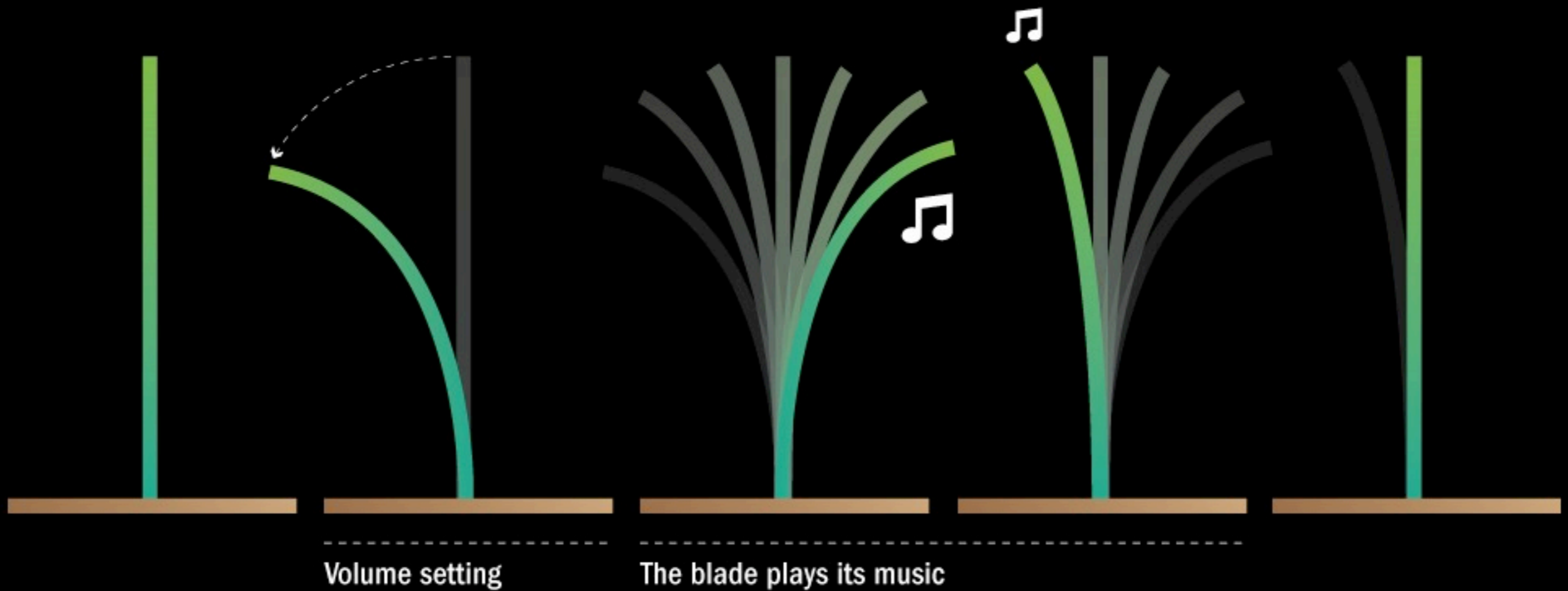


**The harder the gesture, the higher the volume**

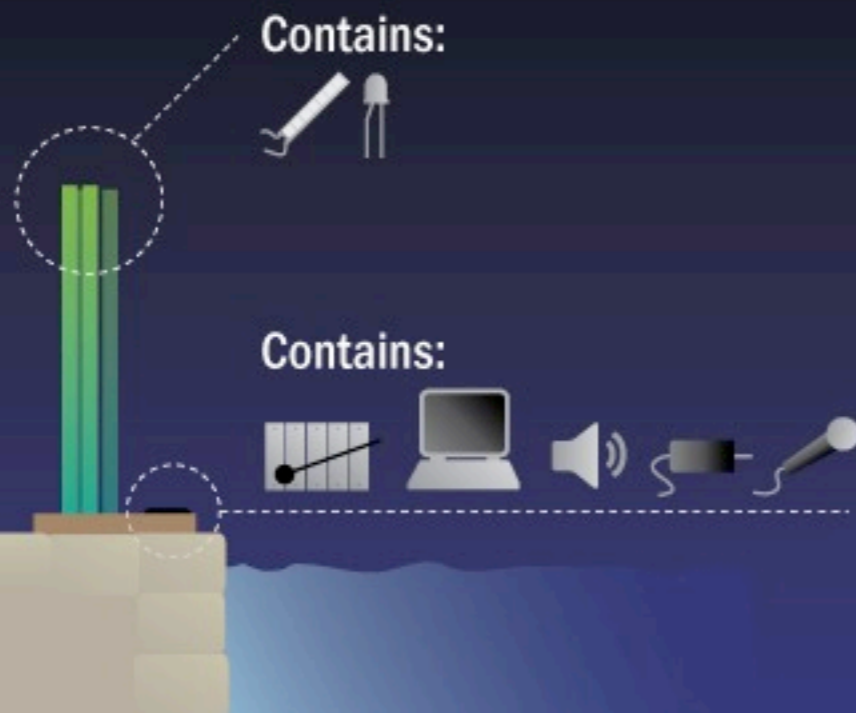


# sounds

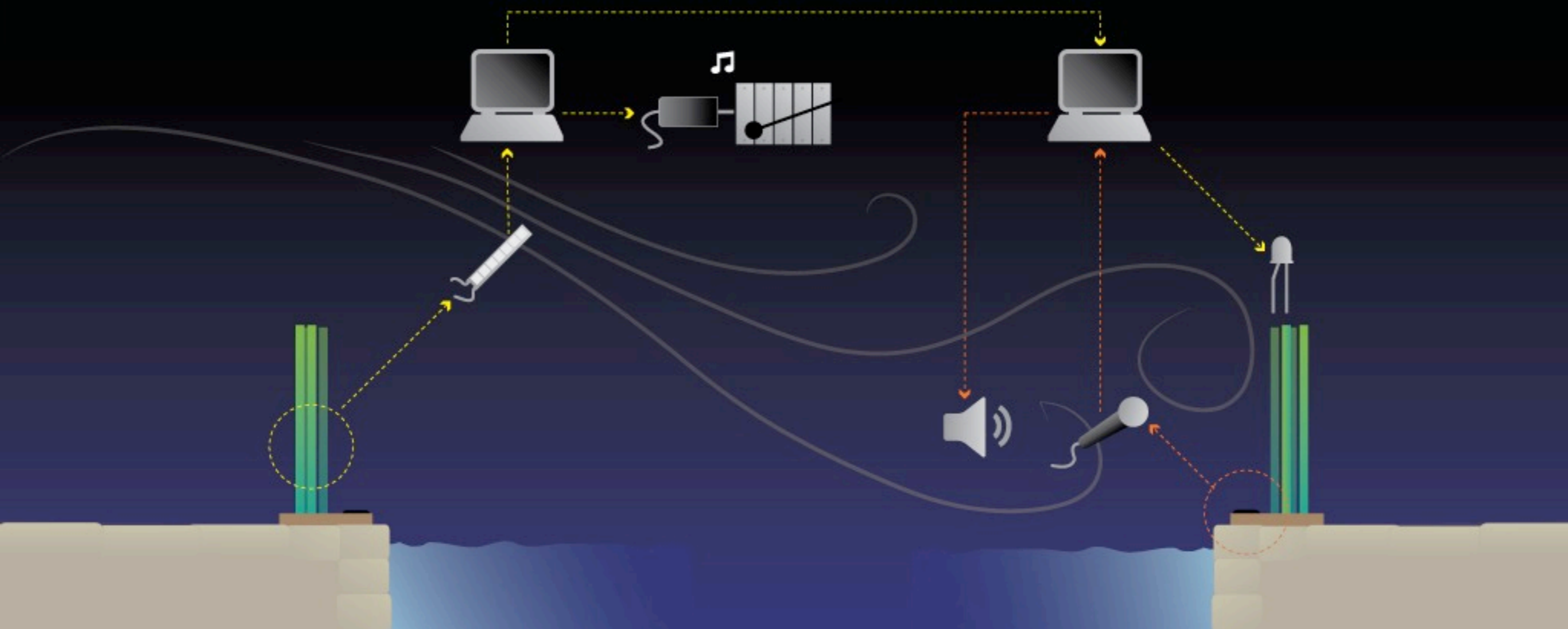
Bending the blade first sets the volume. Releasing it makes it play.  
You will hear a note at every bounce of the blade.



# telecommunication | components



# telecommunications



# coding

The image shows two overlapping code editors. The top editor is the Processing IDE, titled 'steli\_VYBRAPHONE\_server\_client | Processing 0135 Beta'. It contains a multi-line code file with comments and declarations. The bottom editor is the Arduino IDE, titled 'Arduino - C++', showing the 'arduino\_analog\_digital' sketch. It includes the Firmata library and defines several callback functions for digital I/O.

```
steli_VYBRAPHONE_server_client | Processing 0135 Beta
-----
*
* STELI / vybraphone / server + client
*
* Server code: gets values (notes and volume) FROM the glockenspiel installation
* > reproduces those sounds
* > turns on LEDs of vybraphone notes that will play in harmony
* > measures the wind and adds reverberation according to its power
*
* Client code: sends values (notes and volume) TO the glockenspiel installation
* > gets flex pins and value
* > activates solenoids that play an actual vybraphone
*
-----
// Declarations
//
// Libraries
import processing.serial.*;
import cc.arduino.*;
import krister.Ess.*;
import processing.net.*;

Envelope myEnvelope;
Arduino arduino;
Arduino arduino2;
Serial port;
Client client;
Server server;

boolean serverRunning;

// ellipses positions
int posX = 100;
int posX2 = 450;
int posY1 = 50;
int posY2 = 120;
int posY3 = 190;

-----
Arduino - C++
-----
#include <Firmata.h>

byte previousPIN[2]; // PIN means PORT for input
byte previousPORT[2];
byte analogPin;

void analogWriteCallback(byte pin, int value) {
  pinMode(pin, OUTPUT);
  analogWrite(pin, value);
}

void outputPort(byte portNumber, byte portValue) {
  // only send the data when it changes, otherwise you'll be flooding the network
  if(previousPIN[portNumber] != portValue) {
    Firmata.sendDigitalPort(portNumber, portValue);
    previousPIN[portNumber] = portValue;
    Firmata.sendDigitalPort(portNumber, portValue);
  }
}

void setPinModeCallback(byte pin, int mode) {
  if(pin > 1) { // don't touch RxTx pins (0 and 1)
    pinMode(pin, mode);
  }
}

void digitalWriteCallback(byte port, int value) {
  byte i;
  byte currentPinValue, previousPinValue;
```

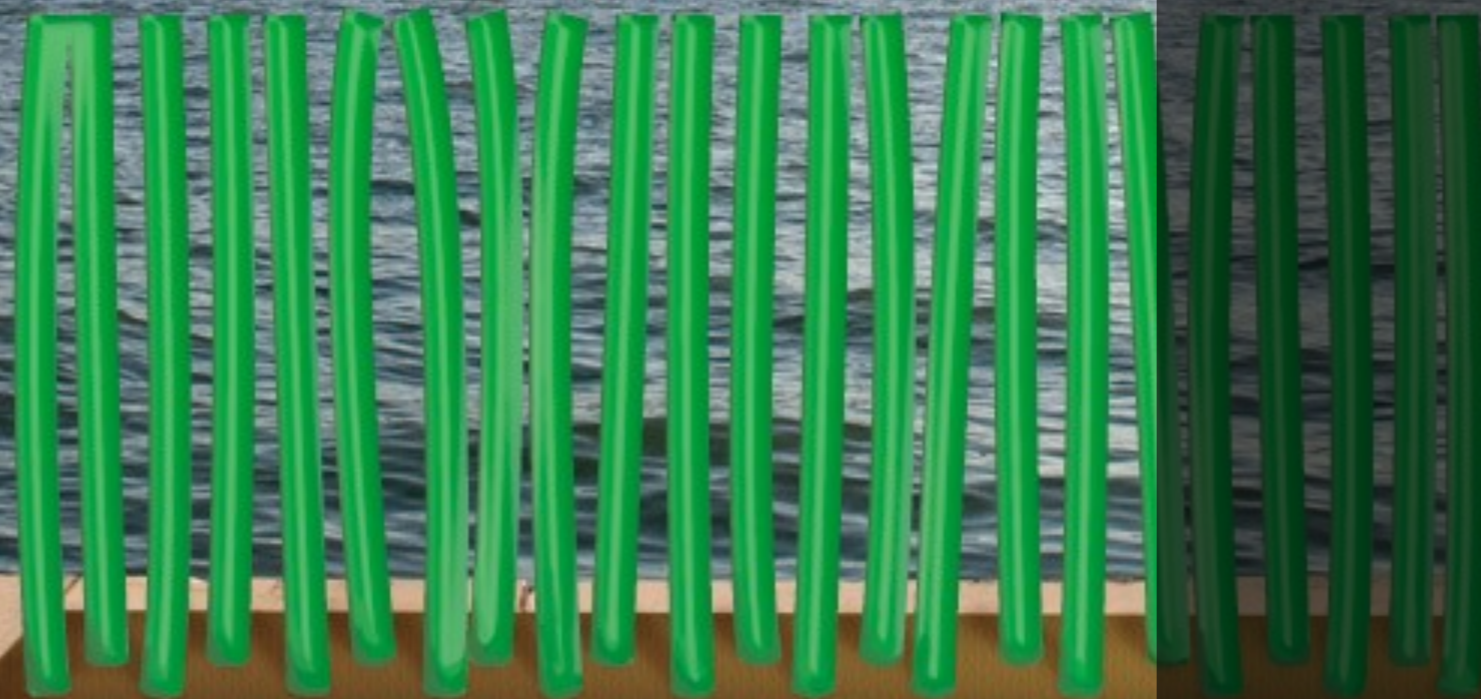
The two computers are connected to the same net and communicate via Wifi.

One is a client and one is a server: they both process and send the bending values to the other one and receive the other's values.

Everything is controlled by Processing, that communicates to Arduino thanks to the Firmata library.

# the real thing

The real thing should have all the direct sounds played by solenoids, that are regulating in intensity by the bending values.



# thank you

Special thanks to

Durrel Bishop  
Tom Hulbert

Filippo Mastinu  
Matteo Torcinovich

and all the IxD Lab.2