Context - Arts/Technology Collaborations

I argue that arts/technology collaboration is a powerful framework for enhancing ideas in both arenas.

Kinetic Sculptures: Creating Programmable Art

Erik Brunvand
University of Utah

Kinetic Sculpture

- Contains moving parts
- Motion, sound, or light
- Often controlled by microcontrollers
- Motors, actuators, transducers...
- Often reactive to environment

Naum Gabo

Russian - 1890-1977

Kinetic Construction
(Standing Wave)
1919-1920

Serpente Rosso, 2013
Marcel Duchamp
French (naturalized US) 1887-1968

Rotary Glass Plates
1920

Built with the help of Man Ray

Marcel Duchamp
French (naturalized US) 1887-1968

Rotary Demisphere
(Precision Optics)
1925

Jean Tinguely
Swiss - 1925 - 1991

Metamatic - 1959

Jean Tinguely
Swiss - 1925 - 1991
Jim Campbell’s Algorithm

Alicia Eggert
Wonder, 2011
Drawing Machine - 2013
Showed at SIGGRAPH 2013

Robert Twomey
US - b. 1979
Drawing Machine - 2013
Showed at SIGGRAPH 2013
David Bowen

US - b. 1975

Telepresent Wind (2009)
Showed at SIGGRAPH 2011


Daniel Rozen

US - b. 1961

rAndom International

London-based collective

Audience (2008)
Physical Computing Essentials

- Get some input from the environment
  - Light, motion, heat, etc.
- Cause something to happen
  - Make something move!

Arduino Microcontroller
Physical Computing Essentials

- Get some input from the environment
- Cause something to happen

- Force a +5v or 0v value on a Digital output pin
- Get some input from the environment
- Read a voltage on an Analog input pin
- Cause something to happen
Arduino Programming Environment

- www.arduino.cc
- Simple open source IDE
- Arduino code is really C/C++
- avr-gcc is the back end

Physical Computing Essentials

- pinMode(pinNumber, mode); // declare a pin INPUT or OUTPUT
- digitalWrite(pinNumber, value); // force a pin HIGH/LOW
- delay(milliseconds); // delay processing (spin wait)

Physical Computing Essentials

- Each of the digital pins can be set to one of two values
  - High and Low (logic 1 (+5v) and logic 0 (0v))
  - digitalWrite(<pin-number>, <value>);
- digitalWrite(13, HIGH);
digitalWrite(13, 1);
- digitalWrite(13, LOW);
digitalWrite(13, 0);

Arduino Programming

- Two required functions
  - void setup(){...} // Runs once at startup
  - void loop(){...} // Loops forever after setup()
- Standard(ish) C/C++ data types
  - Boolean (1 bit)
  - char (signed 8 bits), byte (unsigned 8 bits)
  - int (16 bits), long (32 bits)
  - float (32 bits), double (32 bits)
Example: Blink

```cpp
int led = 13;

void setup() { // make pin 13 an output
  pinMode(led, OUTPUT);
}

void loop() { // turn pin on and off
  digitalWrite(led, HIGH);
  delay(1000); // delay argument is in ms
  digitalWrite(led, LOW);
  delay(1000);
}
```

What's Blinking?

- Built-in LED connected to pin 13

Upload Blink to Arduino

- Load the Blink program from Examples -> Basics -> Blink
- Connect your Arduino with the USB cable

Upload Blink to Arduino

- Make sure you select the correct board
- Tools -> Board -> Uno
- Make sure you select the correct serial port
- Not the bluetooth ports...
Upload Blink to Arduino

- Click on the upload button
- Watch for blinky lights during upload

What's Blinking?

Built-in LED connected to pin 13

Big Deal?

If you can blink an LED you can control the world!

Turning a pin on and off can control all sorts of external devices...

Hobby Servos
Arduino-Controlled Motion

Controlling a Servo

• Pulse Width Modulation (PWM)

Pulse Width Modulation

Controlling a Servo

Servo Control

• PWM freq is 50 Hz (i.e. every 20 milliseconds)
• Pulse width ranges from 1 to 2 milliseconds
  • 1 millisecond = full anti-clockwise position
  • 2 milliseconds = full clockwise position
Controlling a Servo

Luckily you don’t really need to know any of this!

There’s built-in Arduino code for driving servos!

Servo Object (Class Instance)

```cpp
#include <Servo.h>  // include servo library
Servo servo1;    // create servo object
void setup() {
  servo1.attach(9);  // attach to pin 9
}
void loop() {
  servo1.write(67);  // move to 67 degrees
  delay(100);       // give it time to move
}
```

Servo Functions (C++ Class)

- **Servo myServo;**  // creates an instance of Servo class named “myServo”
- **myServo.attach(pin);**  // attach myServo to a digital output pin
  - doesn’t need to be PWM pin - can be anything from 0-13
  - Servo library can control up to 12 servos on our boards
  - a side effect is that it disables the PWM on pins 9 and 10
- **myServo.write(pos);**  // moves myServo – pos ranges from 0-179
- **myServo.read();**  // returns current position of myServo (0-179)
Connecting a Servo

Power is always in the middle.
GND is the darker of the two on the edge.
Control is the lighter of the two on the edge.

Load and Run Sweep
Light Sensor

Light-sensitive resistors
Also called photocells or CdS Sensors

Voltage Divider

\[ V_{out} = \frac{R_2}{R_1 + R_2} V_{dd} \]

• Vout is proportional to the ratio of R1 and R2

• The changing voltage at OUT can be sensed by the ADC of Arduino
• `analogRead(pinNumber);`
• This senses the voltage (0v to 5v) on the pin and returns a digital value from 0 to 1023

Analog Digital
0-5v 0-1023
10 bit resolution
Light Sensor Connection

Calibrate Analog Voltage

Use the Analog Voltage

#include <Servo.h>

Servo myservo; // create servo object
int potpin = 0; // analog pin / analog voltage
int val; // variable to hold analog value

void setup()
{
  myservo.attach(9); // attach myservo to pin 9
}

void loop()
{
  val = analogRead(potpin); // reads analog value (0-1023)
  val = map(val, 0, 1023, 0, 179); // scale to use with servo (0-179)
  val = constrain(val, 0, 179); // constrain to 0-179
  myservo.write(val); // set servo position
  delay(15); // wait for servo
Use the Analog Voltage

```c
#include <Servo.h>

Servo myServo; // create servo object
int potPin = 0; // analog pin / analog voltage
int val; // variable to hold analog value

void setup()
{
  myServo.attach(9); // attach myServo to pin 9
}

void loop()
{
  val = analogRead(potPin); // read analog value (0-1023)
  val = map(val, 0, 1023, 0, 179); // scale to use with servo (0-179)
  val = constrain(val, 0, 179); // constrain to 0-179
  myServo.write(val); // set servo position
  delay(15); // wait for servo
}
```

Servo/CdS Light Meter

Go Make Something!

- You have the basic tools you need
- You can make something move
- You can respond to light
- Use your imagination and the resources of the Studio
  - Printers
  - Laser cutters
  - Cardboard, foam core, paper, etc.

Extra Material
Potentiometers (Knobs)

- Variable resistors with a knob

Potentiometers (Knobs)

- Use them just like a CdS light sensor

Voltage Regulation

- Take a higher voltage (e.g. 9v) and reduce it to a regulated lower voltage (e.g. 5v)
  - Extra voltage is converted to heat!
  - Provides up to 1.5A of current with an appropriate heat sink
  - Will drive lots of servos!

- Cap values not critical…

Schematic:

- 7805
- IN
- 5 volts (ground)
- OUT
- +5 volts DC out
- The capacitors are optional, but make the voltage extra smooth.
Voltage Regulation

• Take a higher voltage (e.g. 9v) and reduce it to a regulated lower voltage (e.g. 5v)
  • Extra voltage is converted to heat!
  • Provides up to 1.5A of current with an appropriate heat sink
  • Will drive lots of servos!
  • Cap values not critical…

Include picture of regulator on breadboard

Linkages: Pantograph

Include picture

Linkages: Four-Bar

Include pictures of linkages

Linkages: Slider-Crank, Rack & Pinion

Include pictures of linkages