Report 4 – Secret Lives

DECO3200 - Human Computer Experience Design Studio
Semester 2 2012

Tom Dermott, Dale Hurwitz, Mark C Mitchell, Matt Norris, David Wallis, Hanley Weng, Adam Younis

Table of Contents

Secret Lives

Goal

The purpose of this report is to provide overall documentation and final outcomes of the installation work 'Secret Lives'.

'Secret Lives' is best described as:

> *A disparate network of computer systems casually converse. As a viewer approaches, the computers become curious, and address the viewer through different forms of communication: a robot hand; swivelling monitors; a webcam-as-eyeball, placing the viewer on-screen. The longer a viewer stays, the more agitated the computers become. Once all viewers move a distance away or turn their backs, the casual interaction between the computers resumes — it is clear that humans are not welcome in this space.*

Background

Our work has drawn inspiration from many things, ranging from pop culture to art installations that encompass our chosen theme. The notion of advanced artificial intelligence and what it will be capable of in the future, has given rise to speculation as to whether it should be allowed to develop to the point of self sufficiency. This mixture of paranoia and possibility has seen AI and robots at the forefront of much popular science fiction and related works.

Recently, in expanding upon the power of intelligently evolving and generative debate, the philosophical notion of knowledge traversal (Griffen, 2012), and the potential within the interconnectedness of ideas (Lima, 2011) has been acknowledged and explored within further evolutions of our concept.

Approach

The following is a list of modular component sections that were worked on collaboratively. In addition to portions of the project that made it to the final installation, this includes omitted components the informed the development of others.

Though each section is headed by primary persons, this collaborative effort saw all members advising on multiple sections.

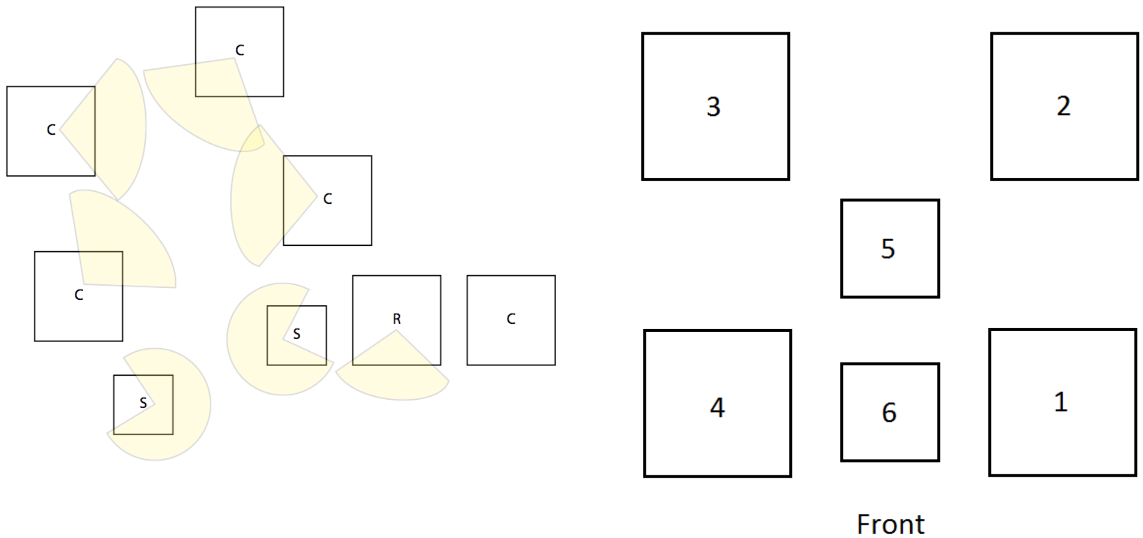
Plinths

The plinths, our solution to housing the installation, have quickly become one of the more integral aspects of The Secret Lives of Computer Systems, serving as a literal and conceptual platform for achieving the full integrity of the piece. As we've worked through the semester, experimentation with the dimensions, colour, layout and function of the plinths has led to the maturation of our concept, working as a catalyst for the discovery of constraints and opportunities.

The production process for the plinths followed a fairly structured and iterative process. We first derived the concept after considering a number of possible spaces and the implication they would have on the piece as a whole. One, for example, was the notion of placing the system on desks and creating the illusion that the installation was merely a collection of workstations. We eventually decided that, conceptually, the installation would benefit from being presented at eye-height, asserting a sense of imposition over the audience as our themes changed from curiosity to a more ominous sense of singularity.

We experimented with uses and forms for the plinths, including a compartmental setup that would allow us to essentially "jig-saw" the plinths together to create a shape that fit multiple spaces and a version that featured stands on wheels with fabric draped over them. After speculating on various factors, we built a prototype and tested it against the rest of the installation as it stood at the time. We were happy with the construction and materials, though it was eventually decided that our final list of components would be comprised mostly of full desktop computers, as opposed to one desktop and multiple smaller mechanisms. This uncertainty encouraged us to experiment with various heights for the plinths, resulting in an established hierarchy and inspiring our final layout.

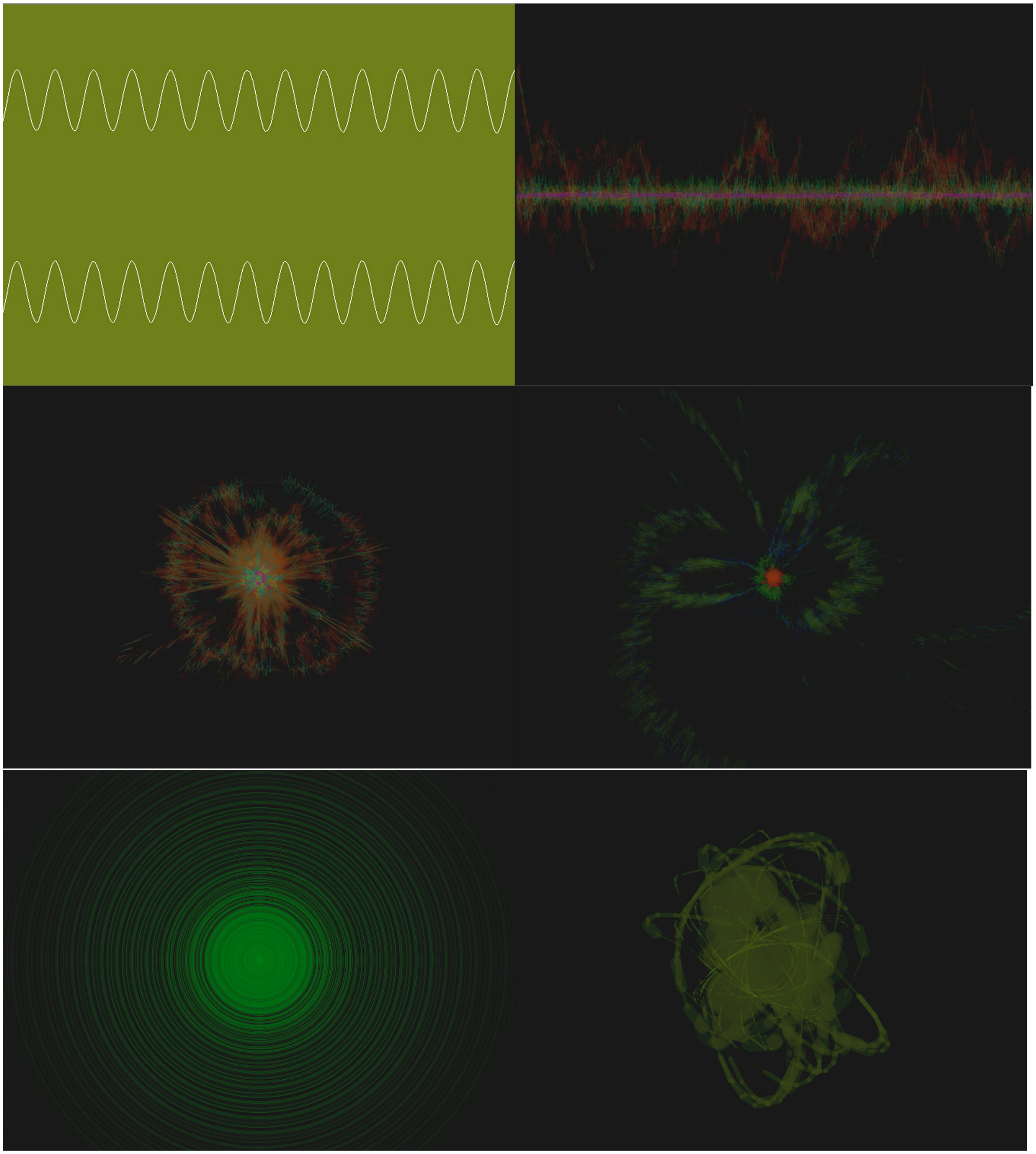
We constructed the plinths from Medium Density Fibreboard (MDF) to form a number of variations of an open rectangular prism, each missing a back panel by design. They range from 1100mm tall to 800mm with a base ranging from 400mm (the diameter of the lazy susans) to 300mm squared. Each has an additional panel that may be used as a shelf for components inside. The plinths have been painted white, to match the iMac computers they support. Currently, the layout design features the four larger plinths in a rough square and two of the smaller plinths in the front and centre of that square.



Left: A preliminary layout for the installation.
Right: The final design.

Speech Visualisations

Because the participants in the conversational are computers with screens, the creation of speech visuals was not only fitting but necessary, as a blank or inactive screen might imply that there is less ‘life’. They are also useful due to the limitations of the turning mechanism, as they indicate which computer is speaking where the degree of animation is insufficient.The visualisations are based on the simple ‘Line In” example from the minim library in processing and expanded to create different aesthetics.

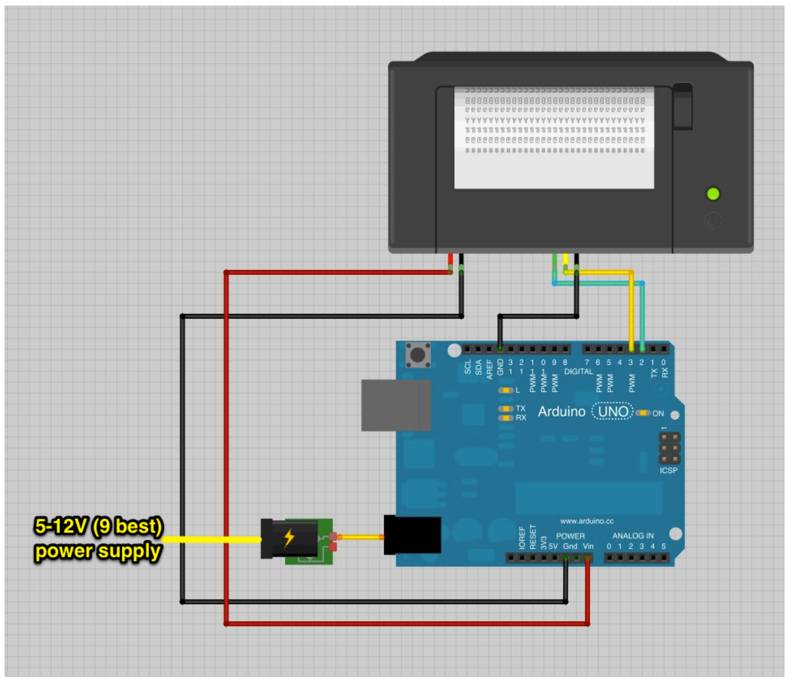


A limiting factor that was not anticipated was the performance of the computers. As such, the visualisations had to be scaled down for several of them, by limiting resolution and particularly the input buffer size, which greatly affects the visualisation density.

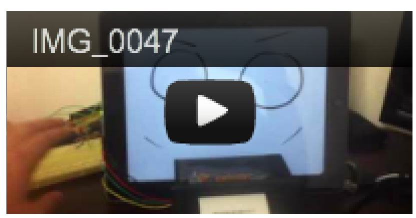
Thermal Printer Development

A receipt printer was proposed as a potential, additional modular component to the installation. Issues involved in setting this up included the sending of text from the computer to the board (this is solved in the below code that demonstrates how to send a sentence from processing to arduino via the serial). It is currently limited to roughly 100 characters. Another issue occurred when weak jumper wires were used, these could not transmit either enough power or data, hence stronger copper wires needed to be used instead.

The printer receives data from the computer via usb and is set up as follows:



A demonstrative video of the printer:



[link]

Code:

- Demonstrative Code for sending sentences from processing to arduino. [link]
- Final Printer Code. [link]

Wireless Arduino Network

At the beginning of this project, initial research (report 1) determined that XBees were the most efficient method of achieving wireless communication between the primary computer (server) and arduinos. Connections to the internet were also considered, none appear to be compatible with Enterprise WPA-2 802.1X networks though. Shields such as the WiFly were found to be inflexible (also pins had to be checked for overlap at shieldlist.org and in schematics). The electric imp breakout was found to be quite useful and worthy of future use.

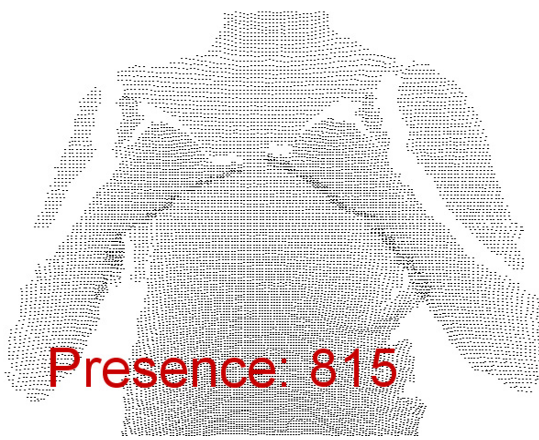
Given the close proximity of plinths, wireless communication was found to be excessively expensive and time-consuming considering that a wired local area network hosted from a single router was preferable for our setup and aesthetics.

Kinect - Human Detection

Kinect was to be used as a means of detecting human presence in the exhibition space in order to prompt a response from the system. Using depth mapping, a single 'presence' integer was output as a reflection of human presence in the area. This took into account two things - the number of people present and their proximity to the work.

The intention was to provide a level of scalability to the response of the system, as well as drive a narrative. For example, the system might have responded anti-socially in the presence of one or two people, and grown exponentially more aggressive as human presence increased.

The kinect was eventually abandoned due to physical variables, mainly the inconsistency and uncertainty of light levels in the exhibition space, which could hamper the functioning of the Kinect. Simple webcams are used as an alternative to much the same effect.

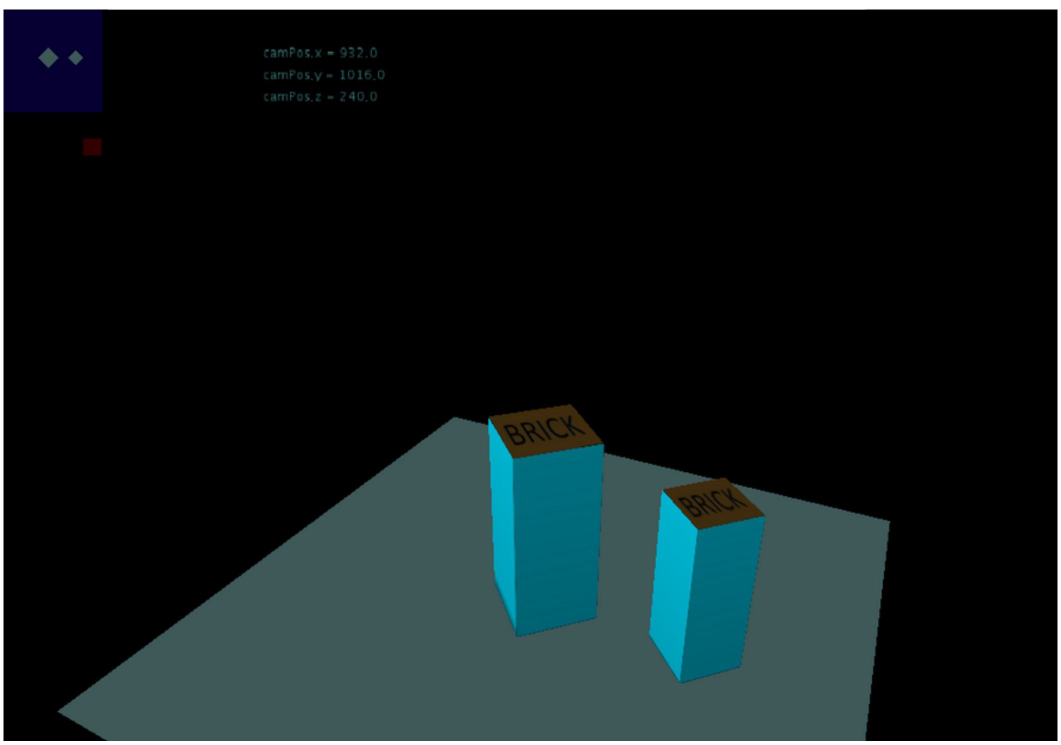


Projected Textures

Projection mapping was one concept that eventually was decided would not be viable for the exhibition due to the logistics of the technique. Work was done in projecting textures onto prisms constructed in 3D space that were analogous, spatially, to real world objects. This was intended to be a way to enhance the impact of the installation by adding a visual layer to the encoded conversation of the computer systems.

The software contained functionality for explicit camera controls (camera position, orientation and focus point as well as field of view) and class based plinths with bitmap textures, potentially displaying anything from generative text to web-sourced images.

Unfortunately, due to the necessary limitations on lighting, required minimum of two projectors and software robustness in a system that already contains a large amount of power consumption and processing resources, the work was abandoned in favour of refining the existing components.



Kinect & Projection Bridge

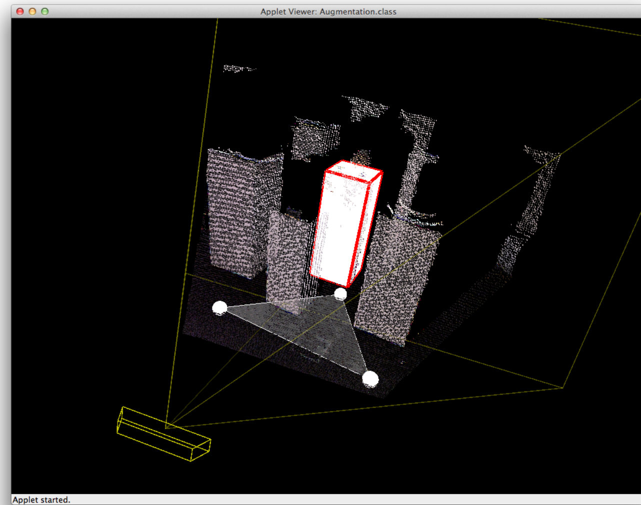
Code was developed to bridge the code done in projecting textures onto objects, and the detection of people with the kinect. The goal of this code is to detect a 3-dimensional space and augment it with projected features.

Issues encountered through setup included conflicting java extensions that were solved by clearing “/System/Library/Java/Extensions/”. The mathematical mapping of planes were found to be complicated, and instead easily approximated through 360 degree loops. SimpleOpenNi was chosen as the core library as it was cross-compatible. One issue that remains is the mapping of the realWordXYZ plane to projectedXY plane, a problem that may be solved with the perspective function among other methods.

Due to an uncertainty of the final environment, this has been omitted from current use.

Code. [link]
Instructions: Connect a Kinect. Run the code. The mouse can be used to control the perspective of the environment. Most functions are handled by keypresses, a code excerpt is copied below.

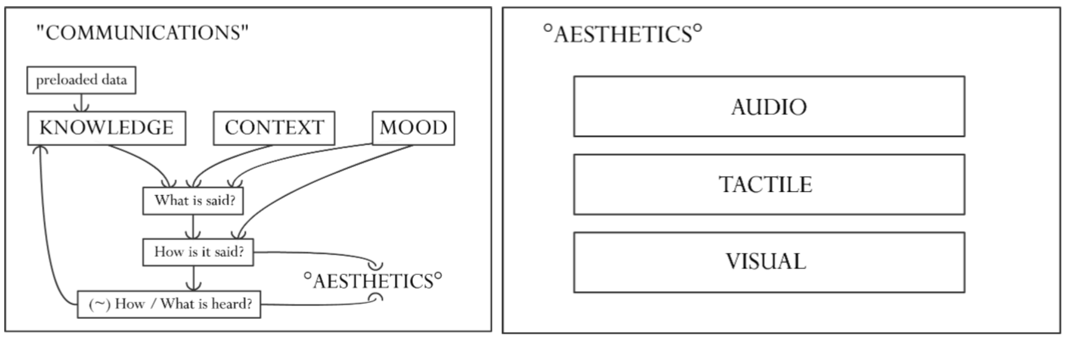
```
public void keyPressed() {
// CREATING PRISMS————————————
// create a cube.
if (key == '1') { ...

// Create smallest plinth 30x30x80
if (key == '2') { ...

// Create medium plinth 40x40x95
if (key == '3') { ...

// Create large plinth 40x40x95
if (key == '4') { ...

// CONTROL PLINTH DIMENSIONS——————————
–
// Increase the scale of the last created prism.
if (key == '=') { ...

// Decrease the scale of the last created prism.
if (key == '-') { ...

// Rotate Clockwise the last plinth.
if (key == 'q') { ...

// Rotate Anti-Clockwise the last plinth.
if (key == 'e') { ...

// Move up the last plinth.
if (key == 'w') { ...

// Move down the last plinth.
if (key == 's') { ...

// Move left the last plinth.
if (key == 'a') { ...

// Move right the last plinth.
if (key == 'd') { ...

// Delete the last created plinth.
if (key == 'z') { ...

// Flip the vertical access of the ground plinth.
if (key == 'f') { ...

// GENERIC————————————————
// Reset Camera to default position
if (key =='c') { ...

// Toggle projectorDisplay (turn off when projecting)
if (key == 'o') { ...

// SET GROUNDPLANE————————————
// Create new Ground Plane
if (key == 'g') { ...

// SETUP GROUNDPLANE POINTS – add a point to
groundplane (use mouse to select ground point)
if (key =='p') { ...
```



[A screenshot of the app with user-set ground plane and a virtual plinth positioned over an existing plinth as captured by the kinect.]

The Spoken Content

Initial research (report 1) found a variety of methods to represent audio, this included roboting mumble, beeps and boops, melodic hums, and direct speech. Speech was chosen as it had a higher impact on audiences, this could be supplemented with other mentioned sounds (e.g. R2D2 & See-Threepio - like dialogue). Conceptually, artificially intelligent debate is a very strong concept (report 2), a diagram of the system that was targeted is as follows:



We have achieved a subset of this as generative and linguistically-aesthetic text is difficult to automate. Through research and pseudocode traversal of different resources of communication content, from "Vade Mecum" (link), bad poetry, religious texts, artificially intelligent linguistics, google scribe, deviantArt comment feeds, news article comment threads, tweets, generative texts, songs, and poetry, wikipedia jumping was decided on as a feasible flexible aesthetic resource of content within our timeframe.

Code was written (below) and stress tested, and follows the repetitive rule: "Grab first few lines of an article. Follow first hyperlink.", whilst inserting intermediary statements.

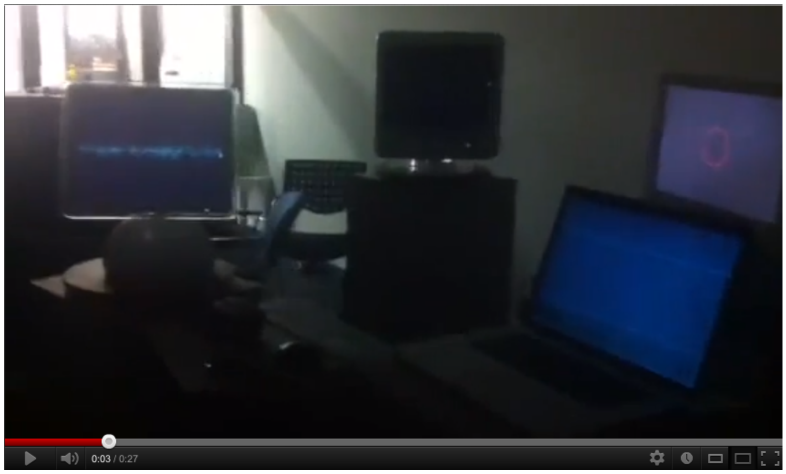
An excerpt of the resulting generated transcript:

> *"A people is a plurality of persons considered as a whole, as in an ethnic group or nation. Collectively, for example, Jews are known as "the Jewish people", European Gypsies comprise the bulk of "the Romani people", and Palestinian Arabs have started to be called "the Palestinian people".*
>
> *Sorry, could you describe person please.*
>
> *A person is a being, such as a human, that has certain capacities or attributes constituting personhood, which in turn is defined differently by different authors in different disciplines, and less formally by different cultures in different times and places.*
>
> *What is human? ..."*

Code. [link]

Server & Client Setup (Hanley Weng, Mark C Mitchell)

The Server & Client Setup refers to the sending of dialogue-commands to computers.



[a clip of the computers talking]

Server Requirements

- Must be run off a powerful computer (post-2000), as processes such as openCv control, printer, robotic hand, turntable, and morse-code control are also handled.
- Requires Mac / Lunix (minimum OSX10.4) for text-to-speech generation and aiff-to-mp3 conversion.
- Requires LAME (installation instructions) for aiff-to-mp3 conversion, this provides access to metadata regarding the length of spoken dialogue. Otherwise, dialogue must be estimated by sentence length. Note Command Line Tools (which can be installed through XCode) must be installed beforehand to act as a C++ Compiler.

Client Requirements

- Mac OSX10.4 as minimum (due to a need to run Java1.5+ for processing and text-to-speech generation).

Server Program Overview

- All dialogue (speech (speed, content, and voice), tray, and event scheduling) is handled by the server program. It is also serves as a controller for various arduino components.
- It should be noted that voices in the server program have been limited to options that are compatible only with 10.4 systems, however, there are additional options with higher operating systems.
- The server consists of some initial variables in it's first few lines of code:
    - boolean initialTurntableControl - dictates if the program can control the turntables.
    - boolean printerControlOn - allows for printer control
    - boolean trayControlOn - allows for eject tray control
    - int printerPort - sets the serial usb port of the printer as recognised by the computer through Serial.println()
    - int turntablePort - sets the serial usb port of the lazy-susan-arduino as recognised by the computer through Serial.println()
- The server utilises key-presses to toggle certain commands:
    - 'd' toggles the dialogue (this is the main program, consisting of speech, turntable control, tray control, etc.)
    - 't' toggles turntable control (only accessible if initialTurntableControl is true)
    - 'c' toggles tray control

Client Program Overview

- Client attempts to connect to a preset server address on start up.
- Client attempts to connect to server every 30 seconds it doesn't hear from the server. (hence server can be closed and reopened)
- Client attempts to connect to server if 's' is keypressed or it is clicked on.
- Client listens for commands to open it's tray, or speak input text.

Connection Instructions

- The IP Address of the server must be obtained (it will show in the console when run, or can be found in network preferences) and set as the target address of all clients (this will not change if the router and ethernet cables remain connected).
- Currently, all clients need to be connected to the server in sequence (assuming the installation faces South, the clients need to be opened in the sequence: 1. SE, 2. NE, 3. NW, 4. SW). Work is being done to omit this reliance on sequence. Once a computer is recognised by the server, it will audibly say that is has been recognised with an id beginning at zero.
- The computer running the server can simultaneously run a client.

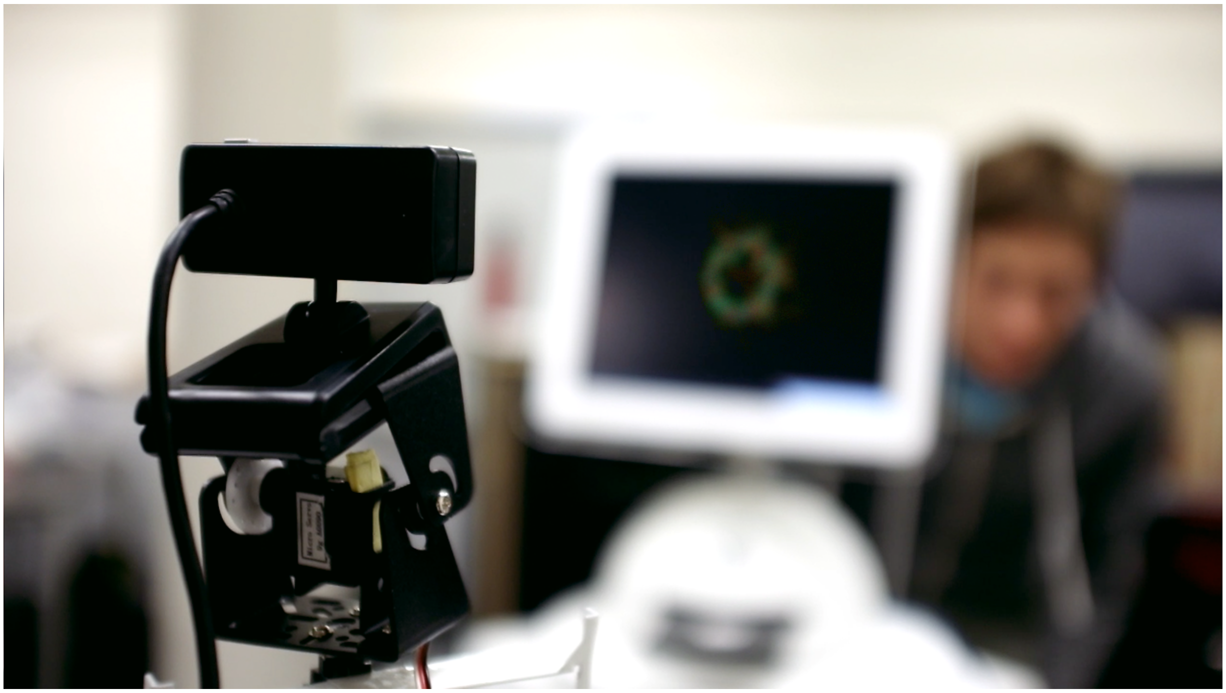
Sentinel (Client_OpenCV Instructions)

- Main sentinel on centre plinth:
  - x-axis servo: pin 9. y-axis servo: pin 10.
- 2nd sentinel on front 30x30cm plinth:
  - x-axis servo: pin 11. y-axis servo: pin 3.

<u>Error when trying to run java applet in Eclipse?</u>

- Make sure the "Run Configuration" for Client_OpenCV.java has this flag under "VM arguments":
  - -d32
  - source: http://www.eclipse.org/forums/index.php/m/518424/
- Make sure the arduino is connected to a usb port of the master server.
- Make sure the webcam is connected to a usb port of the master server.
- Make sure there isn't an instance of Client_OpenCV.java already running.
- **Error about a port already in use?** You may need to change the port number which sends Serial messages to the arduino. The line is here:
  - serialPort = new Serial(this, Serial.list()[6], 9600);
  - Try changing to Serial.list()[0]. If that doesn't work, try Serial.list()[1], etc. You're just checking every usb port to find the one thats sending to the arduino that's controlling the sentinels.
  - **note**: with the mac mini, we had a problem where, according to the code, the arduino controlling the turntables was on one port number, and the arduino controlling the sentinels was on another separate port number, *yet information going to the arduino controlling the sentinels was also going to the other arduino*! I resolved this **by simply plugging the arduino USB's into ports as far away from one another as possible on the master server computer.** This means for both Client_OpenCV.java and Server.java, you'll have to trial and error Serial.list()[*] to find out which port is for which arduino, and make sure they're receiving the right info.

Sentinels (Mark C Mitchell)



[Link to sentient behaviour video]

During the situated layout experimentation, it was suggested that one of the sentinels could be placed at a higher elevation, rather than both sitting on the lowest plinth at the front of the installation. From its new elevation, the sentinel is able to track faces more easily: because it is at the height of the computer screens, visitors naturally look towards the webcam, and its prominent position also makes it clear when a visitor is being watched.

On detection of a face, the sentinel relays a message to the server, which relays a message to the arduino controlling the lazy susans. All four computers stop discussion, and turn to face the front of the installation, where the viewer will be. Unfortunately due to lack of precision with the servo's controlling the lazy susans, they cannot turn to the exact location of the viewer, and depending on the final situated location at The Rocks Popup and the flow of visitors into the space, this relayed message might have to be adjusted.

Sentinel construction

- The sentinel code has been improved, and they now move in what could be considered a smooth fashion, both on the x-axis and y-axis. When no human is detected, the sentinels have a timeout — once they are satisfied the human has left, they go into ‘curious’ mode, searching around the space until a face is detected.
- There are two sentinels connected to an arduino, however the second sentinel on the front plinth has a dummy webcam — it is not active. It moves however in correspondence to what the first sentinel is tracking, with slightly adjusted movements to compensate for its difference in position.
- Having the two sentinels connected to one arduino board means four micro-servo’s are getting sent individual messages from the serial port, and each are moved independently. After much trial and error trying to pinpoint the cause of jittery spasm-like movements of the servo’s, even when no serial messages are being sent through, Matt informed me that, generally, servo’s are a lot more stable if the 5v power connection comes from an external source rather than from the USB port. Switching to an external 5v wall wart power supply has greatly stabilised the movements of the sentinels, as well as providing more power so that the tilt-axis no longer struggles with the weight of the camera.

Lazy Susans

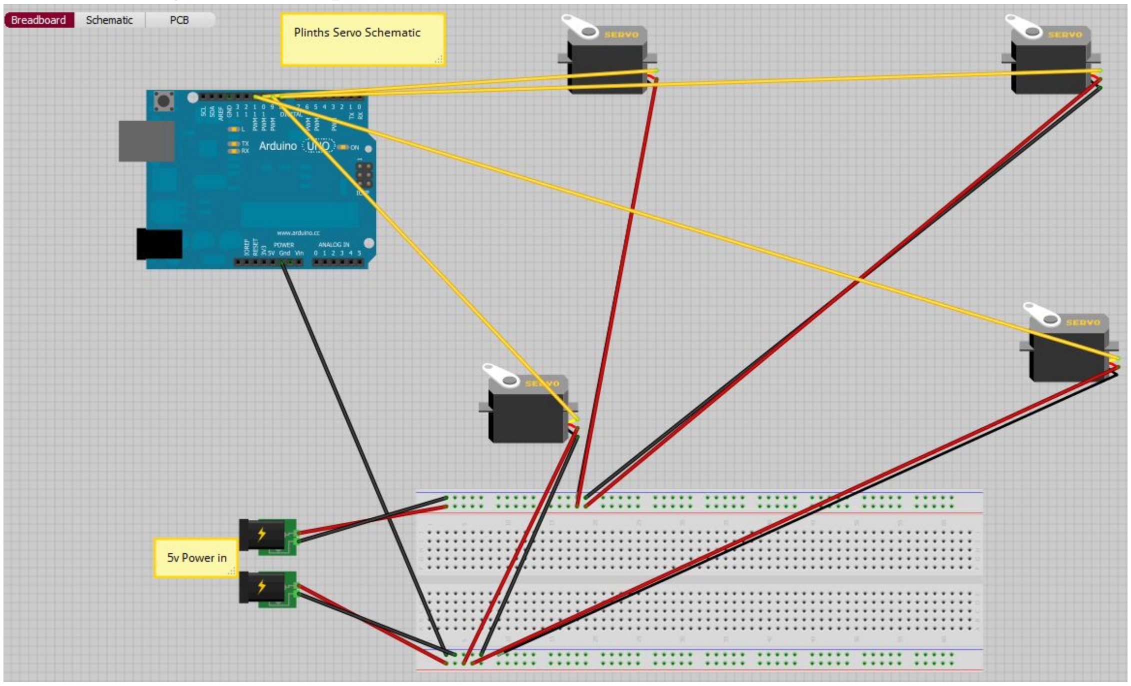
The concept of the lazy susans was to give the 4 main computers a dynamic movement so they could face each other as well as facing any observers of the exhibit. The servos were built into the plinths that were supplied by Adam. The following construction process was used;

1. Mark out the area to be cut away for the slot so the tyre could fit through and then cut this area out.
2. Stick deck grip tape to the bottom of the lazy susan were the wheel will come into contact with it.
3. Mark out and drill the holes for the brackets on the topside of the plinth.
4. Bolt the brackets to the plinth.
5. Mark out the holes to be drilled to hold the servo to the brackets and then drill them.
6. test fit the servo with the wheel and lazy susan, then apply spacing as required.
7. Bolt the servo on then use double sided tape to fix the lazy susan to the plinth.
8. Wire it up with the arduino.
9. Paint the plinths.

The coding for the lazy susans was done to make the screens face the computer that was talking at any given time as well as facing an approaching user. This was done sending a message through to a specified serial port to the arduino which is connected to the main servo through its usb cable. Some discussion about how we want these rotations involved is still to be had as well as finalising the code as a result of these discussions.

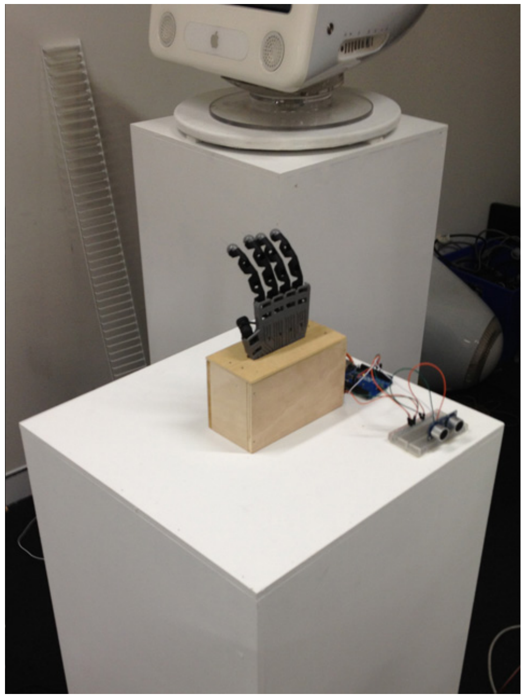
[The motor and wheel used to drive the rotation of the lazy susan]

Layout of the wiring for the four plinths.



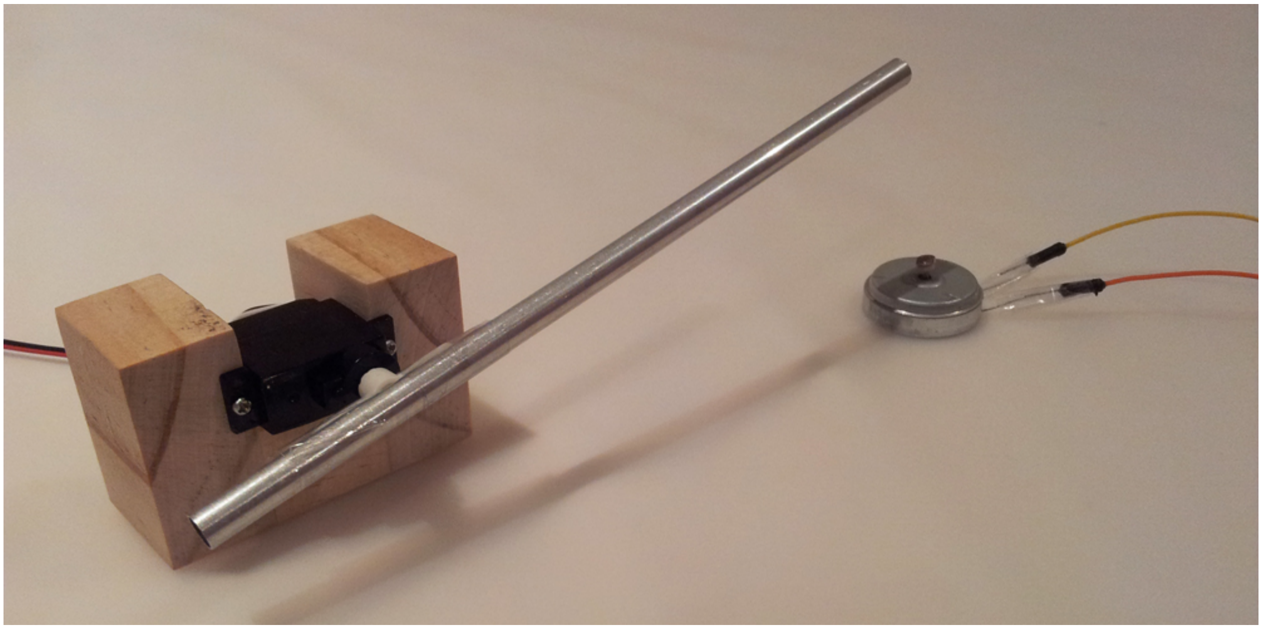
(Two 5v power sockets are used to power two pairs of servos so the servos have no lag due to low power.)

The Hand



The hand was an evolution of an idea that came from initial brainstorming and idea generation, which later came to fruition when it replaced the previously proposed agitated robot. The development of the hand over the robot was attributed to concerns about the aesthetic of the agitated robot and how it would appear in conjunction with the other modules. The hand itself is merely a children's toy altered with motors to control the actions of the fingers creating different gestures. The design utilises a ping)) ultrasonic sensor to measure the proximity of a viewer to the work which it then uses to change between two set hand gestures. The movement of the fingers is controlled by a servo motor pulling down on the Some issues have occurred with the connection of the ping)) sensor to the rest of the circuit, a result of using low quality jumper wires to join them together, stronger wires and soldered connections were needed to resolve this issue.

Morse Code Machine

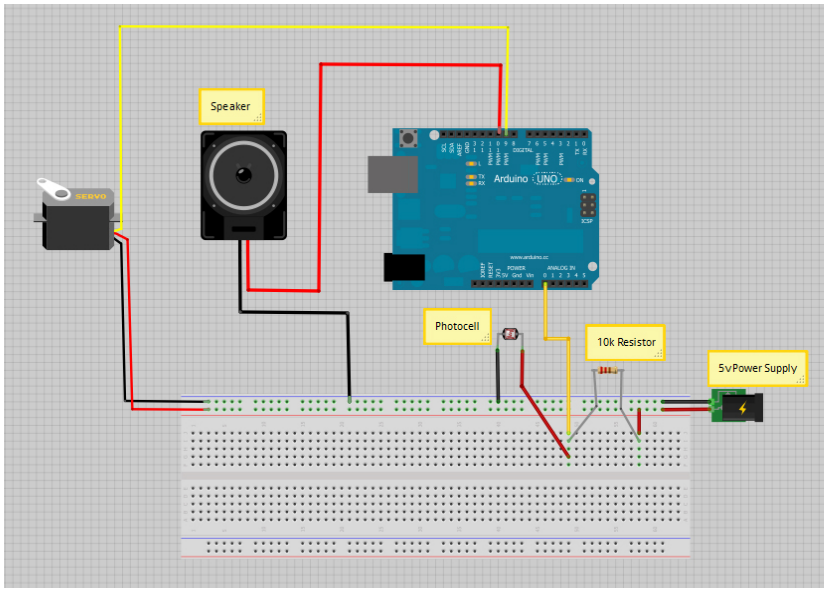


A morse code machine was proposed and developed to complement the spoken text produced by the talking computers. The rationale was to add a layer of coded communication, to create a level of uncertainty and and curiosity amongst our audience. A morse machine is conniving in nature as the audience can detect a rhythm of speech without being able to understand what is said. They are prompted to wonder what insidious messages are being spoken, and this supports the misanthropic & anti-social nature of the system.

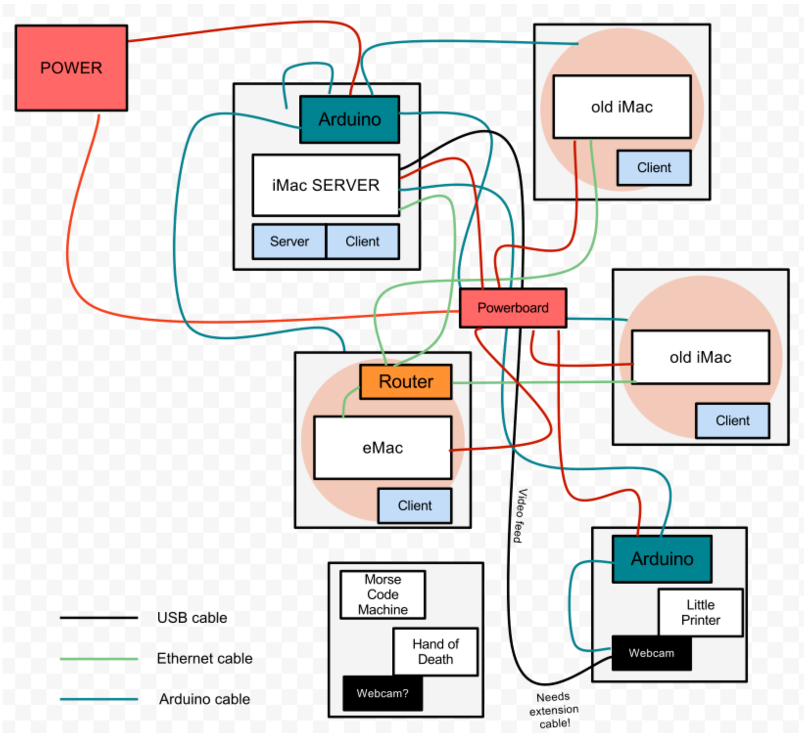
Initially the Morse Code Machine employed a speaker with a broken circuit to create beeps. When the servo moved an arm to its down position, the circuit was physically complete and for a brief period a constant tone is output.The problem with this approach resides in the fact that a single Arduino microcontroller was being used to control both the servo and the speaker. Upon re-completing the speaker circuit, a power surge travels through the Arduino and causes the servo to pulse. This undesired behaviour was not only destructive to the board, but hampered the accuracy of the servo itself.

The re-designed Morse Machine uses a photoresistor to detect light levels, in place of a broken speaker circuit. In it's down position, the arm blocks all light to the resistor which triggers the tone to be sounded

Morse Code Machine Schematic



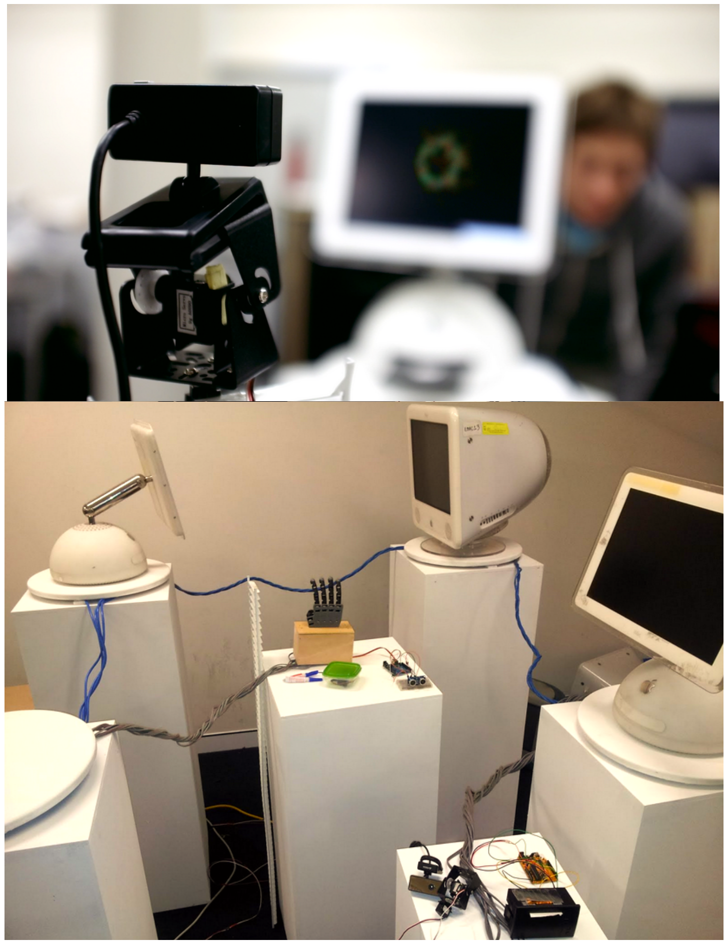
Server Client Setup / Cabling (Mark C Mitchell)



This is a diagram we created to visualise how each of the modular components are to be connected, and how much cabling we would need to set up our situated prototype after the plinths were painted (usb hubs and extension cables especially). It quickly became apparent that the mass cabling could become an aesthetic choice in itself, if we are to make it look intentional. When prototyping the mass cabling idea, a new idea emerged: the centre plinth can have holes cut through the sides, with cables running through the plinth as well as around it.

There was also the suggestion of piling a multiplicity of cables to a depth so high that the entire floor of the installation can no longer be seen — we are currently attempting to find the most feasible way of sourcing such a large amount of cables for minimal cost.

Final Setup



[Images of the Installation as it stands]

Conclusion

In conclusion, we have seen an idea through to fruition, sowing and refining the seeds of many concepts to accomplish a singular whole. Through effective collaboration, we have explored and traversed down many different avenues stemming from our roots in a misanthropic system. This is evidenced in our past reports. Theory was practiced, and practice informed future research as we acknowledged the modular, yet symbiotic processes we developed in bouts of technical development and conceptual progression.

Regarding future work, the installation will need to be altered in accordance with the site and through iterative interaction with audiences. For example, the orientation of sentinels and turntables will need to be adjusted and the frequency of different actions within our system (the opening of trays, speech, turning, and printing) will need to be altered to fit our spectators and the environment.

Bibliography

This is an accumulation of references from the current and previous reports.

- 7 Groundrules For An Intelligent Conversation, Jim Berman, 2008
- affect-logic.com, 2007. http://affect-logic.com/5.html
- Andrew G. Brooks and Ronald C. Arkin. 2007. “Behavioral overlays for non-verbal communication expression on a humanoid robot.” Auton. Robots 22, 1 (January 2007), 55-74.
- Androctor.com, 2012. http://androctor.com/ai-doctor.php
- Bench-Capon, Dunne, 2007. “Argumentation in artificial intelligence”, Artificial Intelligence 171 (2007) 619-641
- Bienkowski, M.A., et al. 2008. “SOCAP: System for Operations Crisis Action Planning” ARPA/Rome Labratory Planning Initiative, 2008.
- blogs.forbes.com, 2012. http://blogs.forbes.com/narrativescience/
- Cutumisu, M., et al. 2005. “Generating Ambient Behaviors in Computer Role- Playing Games”, Proceedings of Intelligent Technologies for Interactive Entertainment (INTETAIN), November 2005.
- Dragoni, A F., P. Giorgini, and M. Baffetti. 1997. “Forming opinions within a group of partially reliable agents: an architecture and a simulation study”. International Conference on Modeling Autonomous Agents in a Multi Agent World, 1997.
- electricSheep.org, 2012. http://www.electricsheep.org/
- Google.com, 2012. http://www.google.com/insidesearch/features/search/knowledge.html
- Gray T, 2009. “THE SECRET BEHIND THE COMPUTATIONAL ENGINE IN WOLFRAM|ALPHA”, http://blog.wolframalpha.com/2009/05/01/the-secret-behind-the-computational-engine-in-wolframalpha/, Wolfram Alpha Blog.
- Isaacson A, 2011. “Are you following a bot?”, the Atlantic.
- Jakobovits H. 2000 “On the theory of argumentation frameworks”, Ph.D. thesis, Dept. of Comp. Sci., Vrije Univ. Brussels, 2000.
- Jinks B, 2012. “IBM’s Watson Gets Wall Street Job After ‘Jeopardy’ Win”, Bloomberg
- Johnston, 2011. “ Jeopardy: IBM's Watson almost sneaks wrong answer by Trebek”, ars technica
- Kafka, 2012. “Twitter + Robots = Instant Stories, No Humans Required”, All Things D
- Karl Sims (July 1994). "Evolving Virtual Creatures". SIGGRAPH '94 Proceedings: 15–22.
- KurzweialAI.net, 2012. “AI game bots ‘more human-like’ than half of human competitors”, http://www.kurzweilai.net/ai-game-bots-more-human-like-than-half-of-human-competitors
- Kyle L. 2008a “Project report: Automatic behavior customization of non-player characters using players temperament.” School of Computer Science, McGill University, Montreal, QC H3A 2T5, Canada, 2008.
- Kyle L. 2008b. “Reading report: Automatic customization of non-player characters using players temperament.” School of Computer Science, McGill University, Montreal, QC H3A 2T5, Canada, 2008.
- Lin T, et al., 2011. “Filtering Information Extracted via User-Contributed Knowledge”, University of Washington.
- McKie R, 2012. “Did Stone Age cavemen talk to each other in symbols?, The Guardian
- Park S, Moshkina L, Arkin R C. 2010 “Mood as an Affective Component for Robotic Behavior with Continuous Adaptation via Learning Momentum”, Humanoids 2010: 340-345
- Pavlov H, 2011. “Will Artificial Intelligence Replace Your Family Doctor?”, Huffington Post
- Pease A, Colton S. 2011. “On impact and evaluation in Computational Creativity: A discussion of the Turing Test and an alternative proposal”. In Proceedings of the AISB symposium on AI and Philosophy, 2011.
- Rahwan I, Zablith F, Reed C. 2007 “Laying the foundations for a world wide argument web”, Artificial Intelligence, this volume, 2007.
- Reed C, Rowe G, 2004. “Araucaria: Software for argument analysis, diagramming and representation,” Int. J. AI Tools 13 (4) (2004) 961–980.
- robococo.net, 2011. http://robococo.net/text_e.html
- Sautoy M. 2012. “AI robot: how machine intelligence is evolving”, The Guardian
- Scaefer M, 1990, “A.I. Doctor! An Expert System Shell for the ST”, START VOL. 4 NO. 7 / FEBRUARY 1990 / PAGE 78
- Simon C, 2008. “Creativity versus the perception of creativity in computational systems”. In Proceedings of the AAAI Spring Symp. on Creative Intelligent Systems 2008.
- Singh T K, 2011. “The memory bound on AI systems: The move towards self-awareness”, http://amateurtks.blogspot.com.au/2011/10/memory-bound-on-ai-systems-move-towards.html
- ThePaintingFool.com, 2012. http://www.thepaintingfool.com/
- Torre C, 2010. “The AI Doctor is ready to see you”, Singularity Hub
- Tugui, A., 2004. “Reflections on the limits of Artificial Intelligence”, Ubiquity, ACM, 2004.
- Uren, V., Buckingham Shum, S., Li, G. and Bachler, M. 2004 “Sensemaking Tools for Understanding Research Literatures: Design, Implementation and User Evaluation.” PrePrint: Scholarly Ontologies Project, Knowledge Media Institute, The Open University, UK
- Waibel M, Floreano D,Keller L. 2011 "A quantitative test of Hamilton's rule for the evolution of altruism" PLoS Biology

- anticulture - RandomLyrics, http://www.anticulture.net/RandomLyrics.php
- Calvi, Buchanan, 2010, A Case on Generative Art: Digital Poetry, NHTV University.
- Cox, 2000. The Aesthetics of Generative Code, CaiiA-Star, University of Plymouth
- deviantart photographs of civilisation, http://browse.deviantart.com/photography/civilization/
- Evans, 2009. Permute Meaning with Generative Poetry, University of Alabama
- feralcreature - lyrics-generator, http://feralcreature.com/lyrics-generator/
- frontiernerds comments –'text-to-speech-in-processing', http://frontiernerds.com/text-to-speech-in-processing
- generating plot scenarios, http://www.archetypewriting.com/muse/generators/plot.htm
- Generative Poetics, http://www.dataisnature.com/?p=82
- Generative Poetry App, http://www.wired.com/beyond_the_beyond/2008/03/yet-another-gen/
- Generative Poetry by geniwate, http://leonardoflores.net/post/15238579726/generative-poetry-by-geniwate
- Generative Poetry with L-Systems, http://itp.nyu.edu/~ap1607/lsys/
- Hall, Edward T. (1966). The Hidden Dimension. Anchor Books. ISBN 0-385-08476-5.
- kimchiandchips blog 544, http://www.kimchiandchips.com/blog/?p=544
- kinecthacks, http://www.kinecthacks.com/
- Lai, 2012. Generative Literature. School of Creative Media, City University of Hong Kong
- metagenhaiku, http://www.tpolm.org/~ps/recyclebinladen/metagenhaiku/
- Mystery Engine, http://mystery-engine.com/
- nearbytweets, http://nearbytweets.com/
- petelevin - countrylyrics, http://www.petelevin.com/countrylyrics.htm
- projection mapping with-kinect, http://forum.libcinder.org/topic/projection-mapping-with-kinect
- song-lyrics-generator, http://www.song-lyrics-generator.org.uk/
- songlyricsgenerator, http://www.songlyricsgenerator.com/
- Sydney Morning Herald, http://www.smh.com.au/
- Connelly, K., Siek, K. A., Mulder, I., Neely, S., Stevenson, G., & Kray, C. (2008). Evaluating Pervasive and Ubiquitous Systems. Pervasive Computing, IEEE, 7(3), 85-88. doi: 10.1109/mprv.2008.47
- Evaluating Pervasive and Ubiquitous Systems. Steve Neely, Graeme Stevenson, Christian Kray, Ingrid Mulder, Kay Connelly, and Katie A. Siek
- Iachello, G., Truong, K. N., Abowd, G. D., Hayes, G. R., & Stevens, M. (2006). Prototyping and sampling experience to evaluate ubiquitous computing privacy in the real world. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Montreal, Quebec, Canada.
- (in)security camera by benjamin chang, silvia ruzanka and dmitry strakovsky. http://www.vitagrrl.com/art/insecurity.html
- Marion Buchenau and Jane Fulton Suri. 2000. Experience prototyping. In Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques (DIS '00), Daniel Boyarski and Wendy A. Kellogg (Eds.). ACM, New York, NY, USA, 424-433. DOI=10.1145/347642.347802 http://doi.acm.org/10.1145/347642.347802
- Napper, V. S. Cardboard Computers: Mocking-it-up or Hands-on the Future. Design at work: Cooperative design of computer systems, Greenbaum, J. and Kyng, M. (Eds.). Educational Technology Research and Development, 42, 1 1994), 97-99.
- The Voight-Kampff machine, http://en.wikipedia.org/wiki/Blade_Runner#Voight-Kampff_machine
- Hofstadter, D. R. (2000). Gödel, escher, bach: An eternal golden braid. London: Penguin.
- Baldwin, A. C. (1853). The traveler's vade mecum; or, instantaneous letter writer by mail or telegraph. United States
- Technology Poem- The Things We Can Create, https://sites.google.com/site/gutierrezsergioeshs/poems/technology-poem
- Technological Dream Series: No.1, Robots, 2007, by Dunne & Raby, http://www.dunneandraby.co.uk/content/projects/10/0%23
- Curious iPhones by InteractiveLab, http://www.creativeapplications.net/iphone/curious-iphones-by-interactivelab-arduino-processing-opencv-ios-of/
- Audience by Chris O'Shea, http://www.creativeapplications.net/openframeworks/audience-openframeworks/
- Glories of Accounting by Rafael Lozano-Hemmer, http://www.lozano-hemmer.com/glories_of_accounting.php
- The Desire of Codes — A nightmarish interactive work by Seiko Mikami, http://doc.ycam.jp/work/index_en.html
- AI vs. AI. Two chatbots talking to each other, http://www.youtube.com/watch?v=WnzlbyTZsQY
- Nathaniel Mellor's Hippy Dialectics, http://www.youtube.com/watch?v=SANv6JK4hYQ
- Tutorial: How to build a Robotic Head that Follows People's Faces, http://www.benfarahmand.com/2011/11/tutorial-how-to-build-robotic-head-that.html
- Ernest Adams, 2004, Gamasutra, '*Postmodernism and the Three Types of Immersion*', <http://designersnotebook.com/Columns/063_Postmodernism/063_postmodernism.htm> viewed on 17/09/2012.
- Lima, M., 2011. *Visual Complexity: Mapping Patterns of Information*, Princeton Architectural Press.
- Griffen, B., 2012. The Graph of Ideas. griffsgraphs.com/2012/07/03/graphing-every-idea-in-history/
- Two 'Hot Issues' in Cooperative Robots: Network Robot Systems and Formal Models and Methods of Cooperation" , HL Akin, 2008
- Coordination of Robotic Systems, LectureSlides, Francesco Bullo, NICTA, 2007