Chloe, Part III

A Robotic Exploration into Digital Companions
Chloe, Part III: Character

Melissa Kronenberger

ITGM 736: Physical Interactive Media

Fall 2013

I Design Abstract

When we use the term, 'interaction design,' we are not limited to discussing video games and web sites. In fact, interaction designers can be found in many disciplines, from other entertainment-specific fields like theme park design, to broader and more varied subjects like marketing. Interaction design is relative to information systems, the service industry, education, communication, and any other system in which an exchange and an experience both occur.

What follows is my continued exploration into the realms of physical interactive media, through the design and development of an interactive toy named *Chloe*. As a toy, Chloe takes the form of a six-legged robot, or 'hexapod.' As in an interactive entity, she is driven by an on-board embedded system.

To further the research and development of my *Agon & Alea Thesis*, Chloe will be used to gain insight into how physical toys can become the artificial companions of their owners.

At the conclusion of *Chloe, Part II*, Chloe had been assembled and her 'brain' board had been mounted on board her chassis. She was able to stand through an on-board executable, and her code had been abstracted to a more natural and modular form for development.

In *Chloe, Part III*, I will be working on getting her to walk, making her autonomous (independent from any external computer or power source), including external libraries, and designing an exterior skin, or 'coat.'

II Table of Contents

III Introduction

1 Project Overview

Chloe, Part III is the last stage of a three-part project that deals with the construction, design, and programming of a small robotic pet named Chloe. This pet will be brought to life by an on-board embedded system. At the conclusion of this project, the pet will serve as a prototype and proof of concept that showcases the heart of what interested me when I first envisioned her as a digital companion.

2 Project Purpose

This report details my continued experience developing with physical interactive media for the very first time. Its primary purpose is to enable me as an interactive designer. As a result, I will have the guts and intuition necessary to design for upcoming technology, or lead a team of toy developers.

Secondly, because *Chloe* already fits the mold of being an artificial character, I will be using her to further my thesis research into how players can bond with and derive emotional pleasure from interactive characters. Specifically, I will use Chloe to evaluate how bring a character into physical space and animating them can be used to create emotional pleasure.

In *Chloe, Part III* I will be using simple motions and a new exterior to dramatically change how Chloe herself is viewed by her audience. By making Chloe a coat of fur, I will transform her from an interesting but unfamiliar and cold-looking 'device' into a warm 'pet' whose exterior immediately conveys an understanding of her purpose and what she will be capable of in the future.

After this project, I intend to utilize Chloe to evaluate how vocal interactions can be used in conjunction with physical movements to create emotional pleasure.

3 Report Structure

In *Chloe, Part I*, I elected to write my report up as a narrative in the first-person present tense. As the report was a personal exploration and thesis tool, I wanted to remember the tense confusion of trying to solve problems I could barely ask the questions for.

In *Chloe, Part II*, I took a step back from my narrative format and focused instead on recounting a targeted inventory of the progress I made, while at the same time attaching a few investigative accounts in their raw and unedited format at the end so that I could preserve a transcription of the experiences involved in researching her most difficult solutions.

Chloe, Part III once more requires a slightly different approach to documenting the process. In *Part III*, I continued to spend significant chunks of time both in investigative exploration and targeted implementation. However, this new phase of development brought with it a previously underutilized lens:

the lens of aesthetic intuition. Owed to its less mechanical and more artistic-oriented roots, *Chloe, Part III* will feature an increase in visual documentation and design reasoning.

In this course, the focus of *Chloe* is centered on the ways she has taught me to implement a physical interactive device. Therefore, I shall discuss my 'implementation and debugging' for Chloe's coat as if it were any other piece of hardware, while at the same time detailing my goals in an aesthetic light.

I shall briefly touch on the reasoning behind my overarching design choices. This course concerns itself more with exploration and learning than with the reasons behind the end product; however I have built up enough designer's intuition in the realm of digital companions that I feel both qualified and interested in sharing my artistic thought process.

4 Personal Methodology Development

At the conclusion of *Chloe, Part II* I had been slightly disheartened, feeling that my progress during that phase had been less emotionally satisfying. Now at the conclusion of *Part III*, I have trouble remembering what that disheartened sensation felt like.

This dichotomy between my ending emotional states is interesting. *Part II* was arguably the most productive time of my *Chloe* experience. I investigated a significant number of new topics, implemented a wide variety of features, researched solutions, cross-compiled, prototyped code and physical configurations, tried out new powering solutions, soldered for the first time, and more.

Compared to *Part III* and *Part I*, *Part II* gave me a lot to show for my efforts. And yet I did not feel as good about Chloe then as I did at the conclusion of *Part I* or as I do now at the conclusion of *Part III*. Why could that be?

There are many possible explanations for why this is so. A simple one is that emotions naturally traverse a wide gamut while working on any project and that *Chloe, Part II* simply happened during a low phase of interest, motivation, or energy in my natural biorhythm cycle. It was midterms at school when *Part II* began, and I was likely psychologically drained at the time.

I could try and tell myself that *Part II* had more failures than the other components. For instance, in *Part II,* I experienced the disheartening frustrations of trying to cross-compile and use Libserial, a library for which I never did manage to find out what the problem was. I also found out my neat charging solution was not going to work. However, this excuse can easily be debunked, and I should not record it as a valid interpretation for future iterations of myself.

After all, here at the conclusion of *Chloe, Part III*, I have failed to implement Inverse Kinematics three or four separate ways, even after devoting long stretches of time to learning the theories involved in modeling robots, and trying a wide variety of tutorials, work-around, and hacks. The experience was probably even more frustrating because it appeared near the end that I had found a solution, but I ran out of time to see it through and had to eschew IK all-together.

In addition, implementing Chloe's coat was slightly nerve-wracking. It didn't feel like the rest of the project, and so it occasionally felt like I was working on 'icing on the cake' instead of an important

component to prototyping Chloe the way she *needed* to be prototyped. When I spent long and painful hours sewing together her coat, only to end up with something easily smothered, prone to falling apart, hastily glued, eye-less, and graceless, I felt a pang of concern that I was diving headfirst into something unimportant that I had no business bothering with in the first place.

And long before *Part III* even began, *Part I* had been extremely exploitative and investigation-based, and each of those explorations involved a significant amount of failure before they succeeded.

Therefore, *Part II* cannot be said to have had more failures and any other part.

I could argue also that *Part II* was tedious and that this is the reason it did not compel me as much as the other parts. A large chunk of *Part II* was programming labor, which is a task I find relatively simple and tedious. Compared to *Part I* alone I could have agreed that this was the problem with *Part II*. Yet compared to *Part III*, where I struggled to snip hundreds of matching holes into a piece of material and string it through with a disobedient leather lace, *Part II* was significantly less tedious and I could feel my own mastery in the final product instead of being confronted with my amateur sewing skills.

Therefore, we cannot say *Part II* was the most tedious.

We could also argue I always need a challenge, and that I prefer to be upset, disappointed, and frustrated with my slowly developing skill-set, so long as I am in an area that's new. And we could argue I would prefer this to being put into a situation where I am able to demonstrate competency, because the latter would be old and repetitive.

Well, we can discredit this precise interpretation by looking at Flora and Fauna, which is a project in which I was happily traversing old Actionscript blitting routes and programming abstract classes reminiscent of StarFarm. I was doing old things *all over again from scratch* and yet suffering no loss of motivation. In fact, the moment I stopped working on Flora and Fauna was the moment I had to start working on Chloe.

An important thing I have to keep in mind when it comes to my project methodology is not to label myself, which is something all these above 'explanations' would try to do. I am not any 'kind' of person, especially not if that 'kind' has some sort of crazy weakness. I am not a 'kind' of person which is good at dreaming and bad at implementation. I am not a 'kind' of person who can't stick to my guns when working on something 'old,' or who can only work when frustrated.

Chloe, Part III has proven to me that any label I felt as a result of *Part II* was ridiculous, and any I can name now are equally ridiculous. There is no need to form an 'explanation' for my motivation that that starts with 'I am the kind of person who...' or any such similar statement. These labels are incredibly disheartening and counterproductive.

There.

That right there is the reason I became disheartened over Part II. The *real* explanation: I mistakenly permitted myself to label some of my behavior. When I became frustrated with something, I told my-

self I just wasn't good at it, which made me feel even worse. That was the problem, and it *has* been a problem for years.

It turns out that not only are those labels 'unnecessary,' they are deadly to a creative process. In fact, they've been plaguing me forever. The real reason I need to do detailed record-keeping is to keep myself from untruthful labeling.

Psychologically, *Chloe, Part III* was a success story. Over the course of the project, I had frequently tried to tell myself that I was simply no good at fabrication. Each time I was able to successfully counteract those problematic thoughts.

My current tool is Malcom Gladwell's *Outliers.* My outlook is that I am not 'good' or 'bad' at anything; I am constantly becoming more of an expert in *everything* I want to be an expert in, and each field requires a large amount of practice and displays imperfections while it is being practiced.

Of course these are lessons I have heard before with different words. Another thing I have learned about project methodologies is that they are personal. It is not until we are receptive and we hear a message with the right phrasing or from the right angle that its implications truly resonate with us. *Chloe* has made me more of an expert in managing my own negative feelings. And it has taught me that managing my negative feelings is integral to my creative process.

5 Project Motivations

I am a game designer, and I am working on creating emotionally compelling interactive characters that can converse with their players and offer entertainment and pleasure through emotional bonding.

When I first stumbled upon the body I would purchase for Chloe's construction, the complete project came to me immediately, along with the motivation to go through with it. I have always been fascinated with robots, and I suddenly found myself with the opportunity to pursue that interest while at the same time furthering my Thesis.

In fact, I would go so far as to say that I fell in love with the project instantaneously, and that completing it provides its own motivation.

In addition to this simple but powerful motivation, Chloe provides an opportunity for me to grow my personal design and development methodologies, create a toy to play with my cat, test my thesis, and, in the future, test the effect of sound-based interactions with technology. It also provides an opportunity to gain insight into physical interactive systems.

6 Objectives and Aims

Chloe, Part III is responsible for meeting several independent sets of objectives.

Firstly, *Chloe, Part III* must continue working towards fulfilling the course requirements set forth by my ITGM 736: Physical Interactive Media course. The course requirements are defined as such: students must learn to use physical input devices, develop physical interactive media prototypes, research the best software and hardware solutions for a personal project, develop a personal project using custom software and hardware, produce short video documentation of their projects for festivals and portfolio reels, and write research reports.

Secondly, *Chloe, Part III* must work towards the individual requirement of the project as set forth in the syllabus; in this case, Project C. This includes respecting the time table set forth in the course syllabus for the project pitch, prototype, documentation, and presentation.

I have also identified the objectives that I intend to meet by the end of the course, and those which I will play with as part of my extended objectives, but which are not vital to Chloe as a whole. These objectives shall be described below under 'Project Goals.'

My overarching objective for *Chloe* is to give me the confidence and familiarity necessary to start other physical interactive media projects. My learning objectives are focused on gaining a broad understanding of the discipline and the various steps of the development pipeline. In this way I not only gain an entry-point to learning more about the discipline, but I also develop the understanding I will need to communicate with hardware engineers and low level programmers in the future.

At the end of *Chloe*, I should be able to visualize the process for translating almost any idea for a physical interactive media project into a design and then into a fully developed product.

On a personal level, I am eager to explore the difficulties with skinning a mechanical construct, and why it is that electronic toys often look stiff and constipated and their motions are ponderous. I want to know more about cross-compiling for embedded systems and I hope to get a cross-compiled library working to help me feel that my understanding of such tiny computers has fleshed out a full circle.

7 Project Goals

Below are listed two sets of goals. At the start of Project A, at the beginning of the course, Chloe was initially envisioned as being reactive to sound. Later, during Project B, it was determined that focusing on the implementation of motion and an Inverse Kinematics library would do more to exemplify the heart of Chloe than implementing sound responsiveness. Then at the beginning of this project, I pitched a proposal that focused very strongly on Inverse Kinematics.

However, I was starting to get the sense that I was 'missing something,' and that an alternative lens would help me visualize a better and more targeted approach for conveying my vision for Chloe. Mingling with my peers on pitch day helped me form a more solid concept of what I needed. After pitching and receiving approval for my IK plan, I set my proposal aside and posed a question to my peers and coach: Would it have greater impact, and would it still be within the scope of the class, if I eschewed the IK system and focused on making a 'skin' for Chloe, to give her personality and help convey the type of toy I was envisioning?

I received immediate and full approval, both from my instructor and from my peer group. Where the IK implementation had been a sound plan, it had lacked the significance and effectiveness a 'skin' or 'coat' would bring. Furthermore, crafting the coat would involve its own challenges that were just as applicable to the art of creating physical interactive devices as implementing any library, and more laterally-minded than continued coding would have been. In sum, the attributes of the coat proposal were deemed wholly superior to the elements of the IK proposal, and it was quickly accepted in substitute.

Below I have listed the two goal sets: the original eschewed IK proposal for reference, and the new coat goal set which ended up serving as the project backbone.

7.a Original IK Goals

- Install new battery pack such that Chloe can operate separated from the computer and other charging stations.
- Software that meets the following objectives:
  - Behaviors can be displayed and modified in real time through a console in order to experiment with and perfect movements.
  - A near-accurate model of Chloe's small physical discrepancies has been ascertained by using the real time console and entered into the program such that Chloe is able to account for it when moving.
  - An Inverse Kinematics library has been included and meshed with Chloe's existing code, and is able to drive her motions.
- Chloe displays the following behaviors:
  - Chloe is able to walk in a relatively straight line.
  - Chloe is able to turn herself around 180 degrees to face in the opposite direction.
  - Chloe can wave a limb in greeting.
  - Extended Goal: Chloe can drum her forward limbs on the ground like a jumping spider.
  - Extended Goal: Chloe can 'dance,' even if just a little.
  - Extended Goal: Chloe can hop.
  - Extended Goal: Chloe can 'kiss' which is to stop, look in a different direction (wherever I will be standing during the presentation) and lift herself up just a bit with her head tilted upward and her back end tilted down, and then to hold that position for a few seconds to await a kiss.
- Extended Goal: Chloe reacts to sound
  - Extended Goal: a library has been implemented that can detect changes in sound.
  - Extended Goal: Chloe has a single microphone mounted on her.
  - Extended Goal: Chloe can respond to loud sounds by looking around.

- Extended Goal: Chloe can recognize and respond to a near match for her name, and will respond by looking around and waving.
- Extended Goal: Chloe has two microphones mounted on her.
- Extended Goal: Chloe can triangulate the direction sounds come from and will either look towards loud ones or her name, depending on which functionality is implemented.

- Video documentation is to be taken of Chloe's movement during development.

7.b New Coat Goals

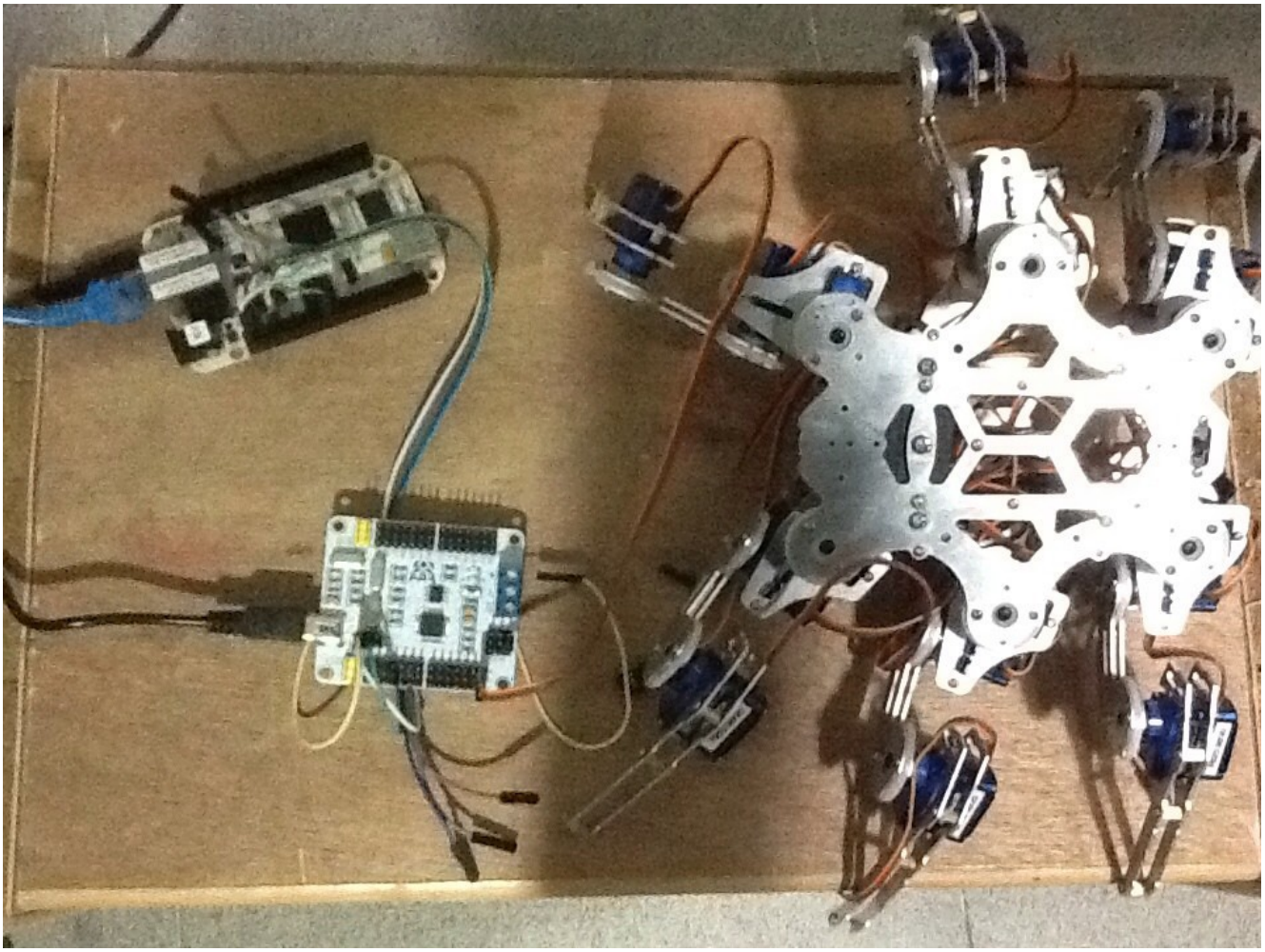
- Chloe is to be 'autonomous'
  - She is to be powered entirely by an on board battery.
  - From boot, she is to be driven entirely by her on board computer.
  - She is to require no physical connections to any outside entities, including data lines to controlling host computers or initializes.
  - She must appear to be a completely independent and self-contained entity.
- Chloe must have a 'skin' which will henceforth be referred to as her coat, and which must meet the following logistical criteria:
  - It must be tailored to the full range of her movements, so that it does not significantly inhibit her functioning.
  - Its loose threads and fur must not damage or be damaged by her servos.
  - It must be removable, so that alterations or repairs can be made to the internal components.
  - It must have rubber feat to increase traction on the ground and reduce damage to the cloth material.
  - It must not cause a short circuit due to any internal metal armature. The areas most at risk are the BeagleBone pins and power supply, as these are the areas with exposed live pins.
- Chloe's coat must exemplify the following aesthetic choices:
  - It is to be fluffy, furry, and soft.
  - It is to make Chloe more familiar and less threatening.
  - It is to be perceived as cute.
  - It is to be huggable.
  - It is to be fairly robust, and difficult to damage.
  - It is to consist of 'natural' colors.
- Extended goals
  - To implement an inverse kinematics (IK) library
    - To attain an understanding of how kinematic modeling works
    - To model Chloe for IK implementation
    - To cross-compile the libraries and gain further insight into cross-compiling
    - To make a smoother and more efficient walk-cycle
  - To implement additional motions
    - Waving
    - Turning
    - Dancing

- To extend Chloe's aesthetic appeal
  - To give Chloe a 'head' or face.
  - To give Chloe large eyes
  - To give Chloe ears, which may or may not be the intended microphone ears for later in the project
- To address cooling issues relative to Chloe's coat.

- Video documentation is to be taken of Chloe's movement during development.

IV Background

1 List of Parts



1.a BeagleBoard BeagleBone

1. White, Revision A5
2. Needs 5V power
3. Thorough documentation, community, and guides at www.beagleboard.org
4. Embedded distribution of Ubuntu

1.b Control board for direct interface with servos

1. Needs to be powered by two different inputs for power and signal: 7.4V and 6.2V.
2. UART driven, capacity of 32 analog servos
3. Can supply 5V power for powering BeagleBone.

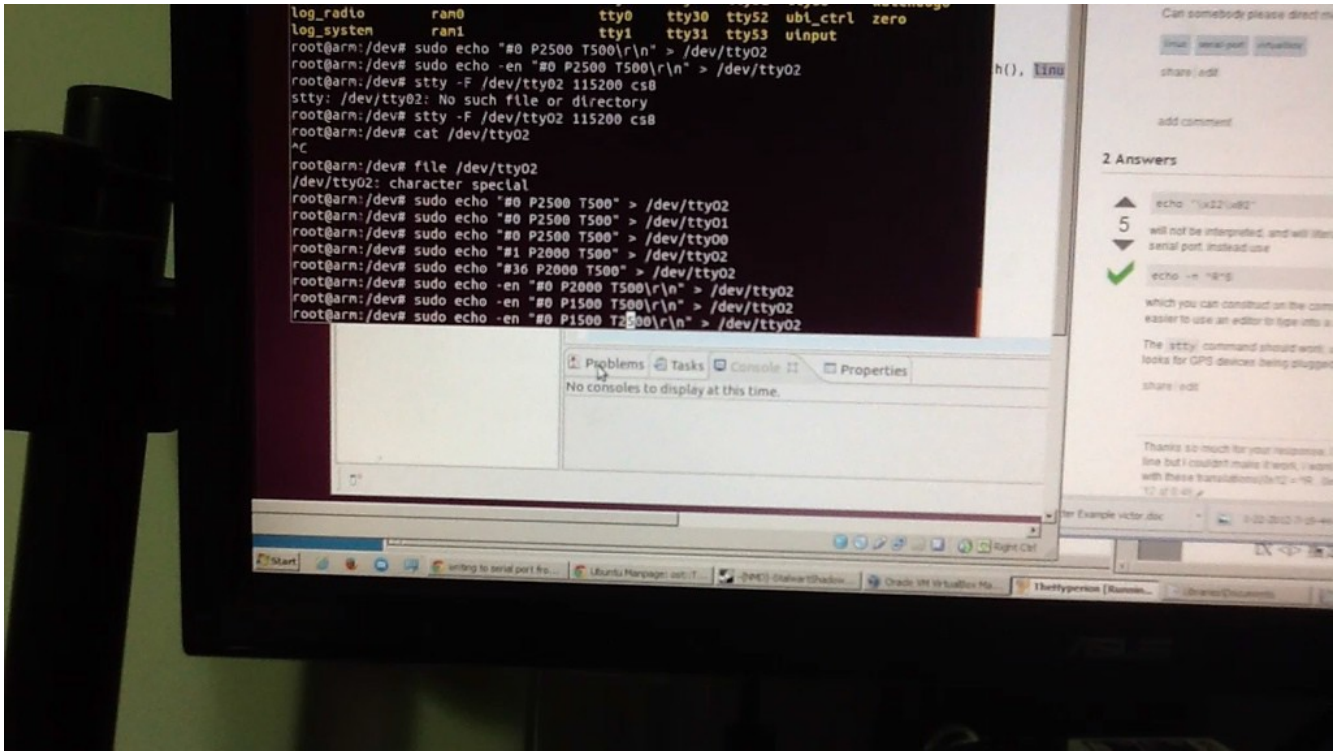
1.c Chloe's Chassis and Legs

1. 18 analog servos, 3 per leg

2. Assembled

1.d Windows PC

1. Windows 7
2. Running Ubuntu Desktop Linux distribution over Oracle Virtual Machine for ease of communication with board
3. Eclipse IDE
4. GIT repository at Bitbucket.org for tracking changes to code and enhanced portability



1.e Software from *Part II*

1. Program that can drive all motors
2. System that can abstract, group, and naturalize commands for rapid development
3. Console for debugging, modeling, and real-time editing
4. Framework to load and save data, if needed
5. Hard-coded instructions for initialization protocol , standing, and walking.

1.f New Battery

1. 7.4 volt battery instead for RC (remote controlled) systems, with 5500 milliamp capacity.
2. Regulator to convert 7.4 volts to 6.8 volts for control board powering needs
3. Lithium-Poly Ion (LiPo)

4. 2 Cell DC 7.4V system
5. 2 Cell DC 7.4V charger

1.g Fabrics

1. Two and a half yards of a soft, robust brown fabric, for Chloe's lowest coat layer
2. Ten square feet of 'furry' fabric in three colors to make her fluffy
3. A yard of bland, tan, fabric scraps for prototyping
4. Two yards of velcro tape

1.h Sewing Needs

1. Small portable sewing machine
2. Assorted colors of thread, including white, brown, and black
3. Assorted sewing needles
4. Sharp scissors

1.i Backup 'Lacing' Plan

The sewing machine is not regarded as reliable, and there is no backup sewing machine available. Because hand-stitching the coat may prove too time-consuming to be feasible, several backup plans were conceived of in the event the sewing machine fails.

1. Five yards of leather string
2. Hand held 'snipping' scissors for cutting small holes through which to pass the string
3. Large needle with room to pass through leather string
4. Stapler, to rapidly 'sew' large strips of fabric, for prototypes

1.j Helping Tools

1. Electrical Tape (Insulation, prevention of short circuits, use with power components)
2. Masking Tape (Sewing aid)
3. Assorted pins, including safety pins and paper clips
4. Paper and drawing implements, for making patterns
5. Glue gun, with glue
    1. For use with Chloe's chassis screws
    2. For use with Chloe's velcro tape
    3. For use with constructing any internal armature

1.k Soldering Tools and hardware

1. Soldering iron, with stand

2. Sponge
3. Soldering paste
4. Soldering medium (wire)
5. Assorted screwdrivers
6. Pliers
7. Wire cutters
8. Additional wires, headers, pins

1.1 Accessories

1. Two feathers, for 'ears'
2. Two yards of ribbon
3. Several yards of iron wire, for building an internal armature if needed.
4. Plastic ties for securing wires and other components.

2 Intangible Resources

Most of what I took form *Chloe, Part I* was in terms of learning and understanding. I could assemble a new chassis in a day, set up my work environment in an hour, and reconnect my boards in a minute, but the conceptual models I require in order to perform those tasks was forged over the entire length of *Chloe, Part I.*

Out of all of the information I aggregated and interpreted for *Chloe, Part I*, there are certain components that will serve as immediate launching points for development in *Chloe, Part II.* Chief among these is my knowledge of what a ‘pin multiplexer’ is.

Using the Linux terminal application during *Chloe, Part I*, I was able to locate and run a script that activated my UART2 Rx and Tx pins. In layman’s terms, I found the tool that would allow me to stick one half of a wire into my BeagleBone, and the other half into my control board, and transmit commands that would get translated by the control board and relayed to the servos.

I also learned to identify how Linux was able to talk to devices, and that the serial device I wanted to transmit across would take the shape of a symbolic file named ‘ttyO2’ that would appear in a certain folder on board my Linux distribution.

I demonstrating my understanding of this by sending commands in via the terminal and observing as the leg moved.

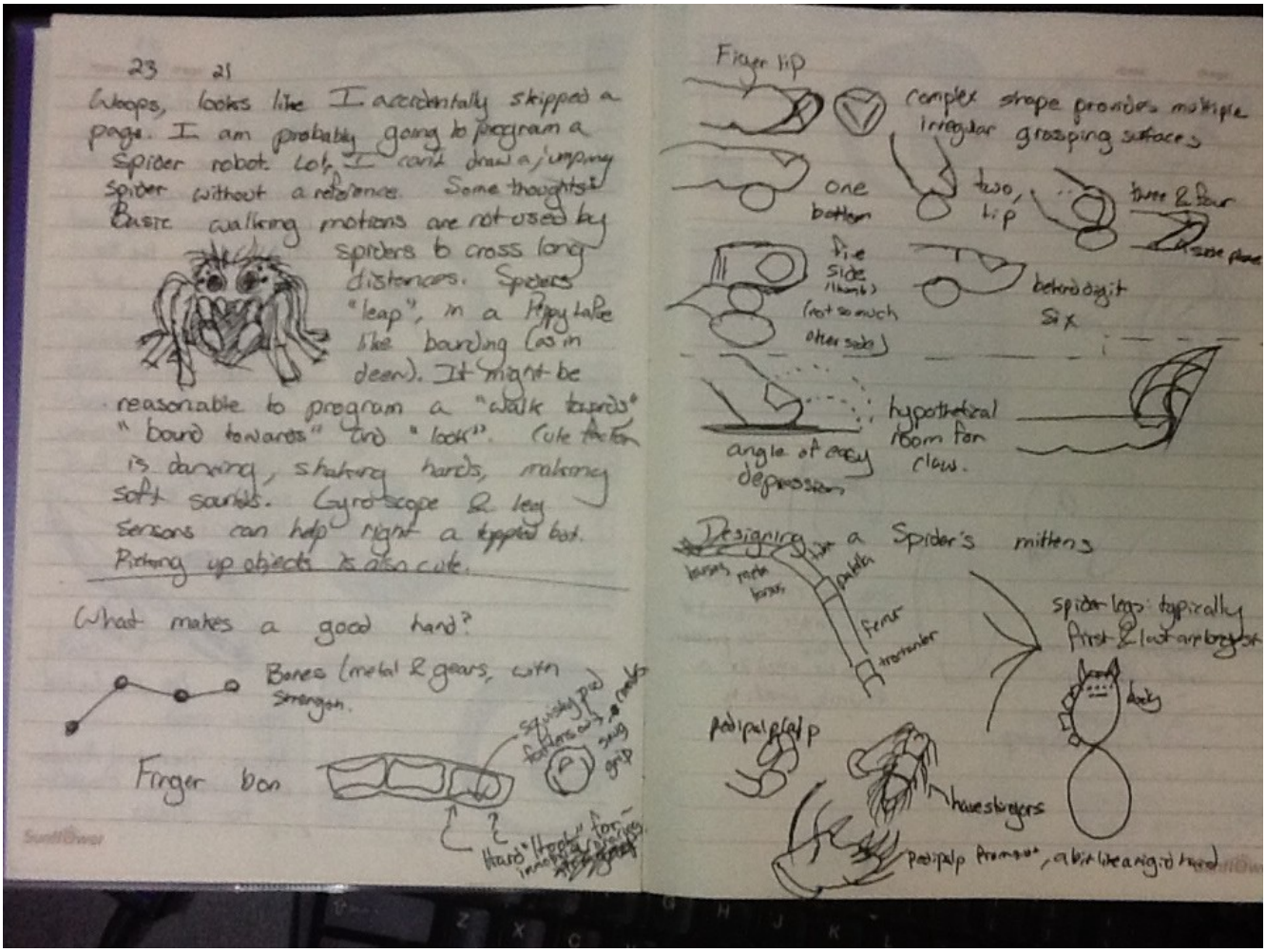
However the ‘ttyO2’ logical port does not remain open when the BeagleBone is restarted. I will have to go into my kernel’s files and determine how to ensure the port is activated on boot. That way, any C++ program I write will be able to talk to through the port without me hacking into a Linux console and initializing anything. This is referred to as ‘pinmuxing’ or configuring the pin multiplexer.

In addition, I need learn the API necessary to communicate with the port via C++. With something simple like the on board LED's of the board, this meant using commands like 'fopen' and 'fwrite.' In communicating across a serial port, I anticipate I will have to use additional functionality just to initialize the port, much less write to it.

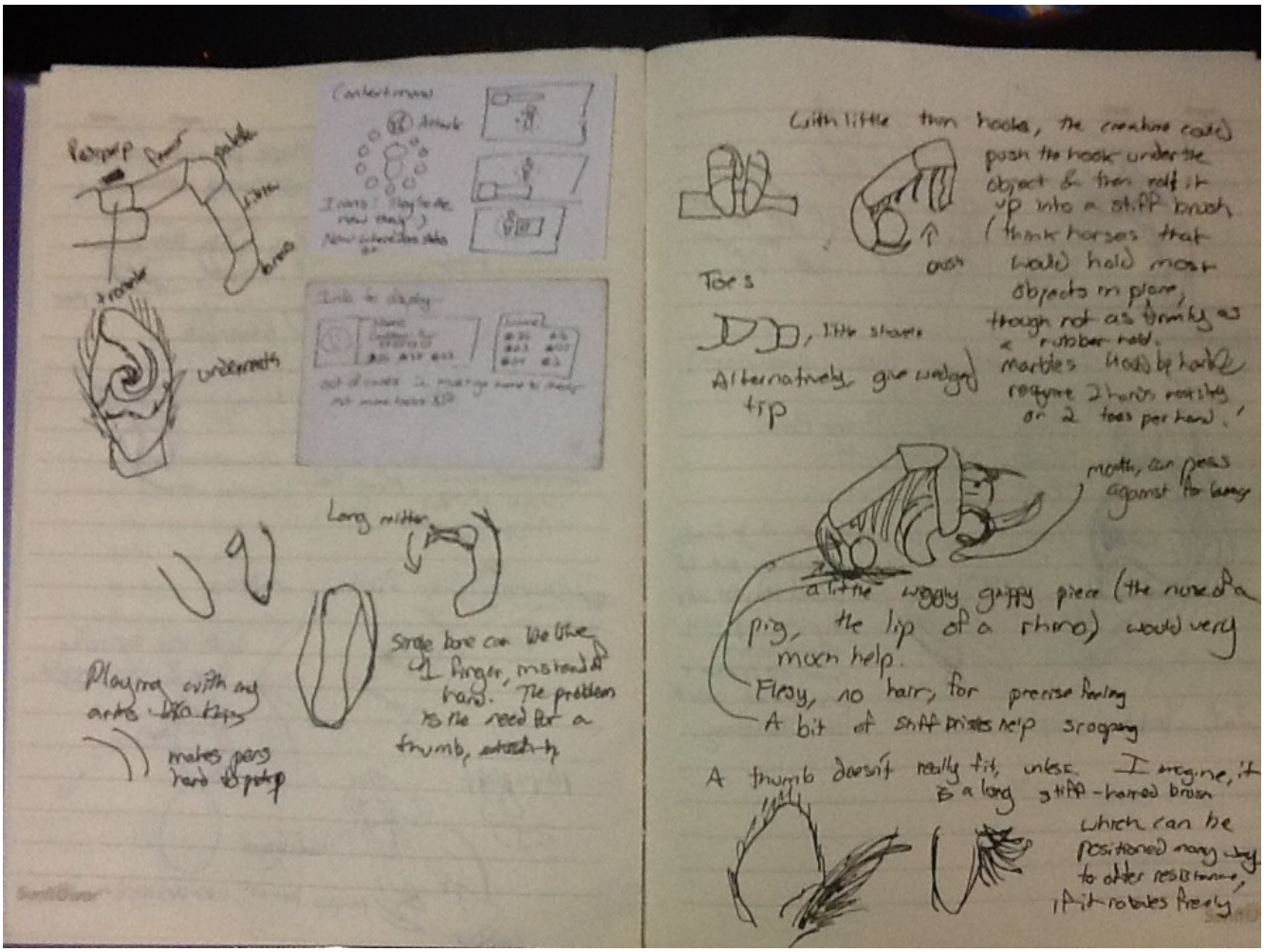
As a result of *Chloe, Part II*, I discovered the basics of cross-compiling and how to port libraries to on-board systems. I configured the pin-multiplexer and had implemented the communications pipeline. Perhaps most importantly, I had discovered several errors in my understanding of Chloe's powering system and now had time to discard my original implementation and construct a new one. As of this point, I have never seen Chloe stand up properly with her power needs being fully satisfied. I am eager to see that.

V Project Management

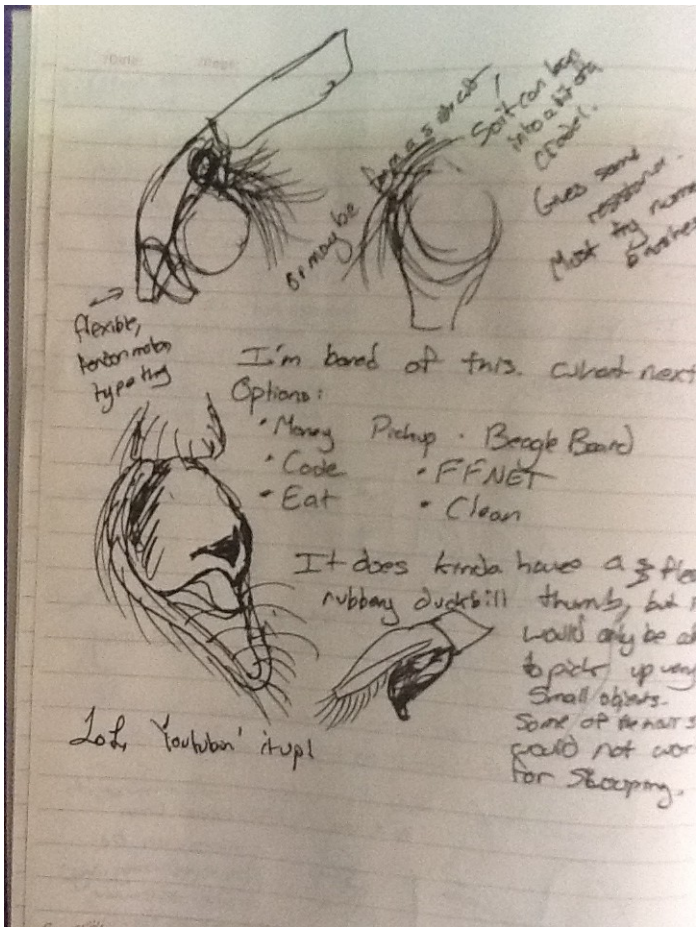
1 Sketching as a Design Methodology



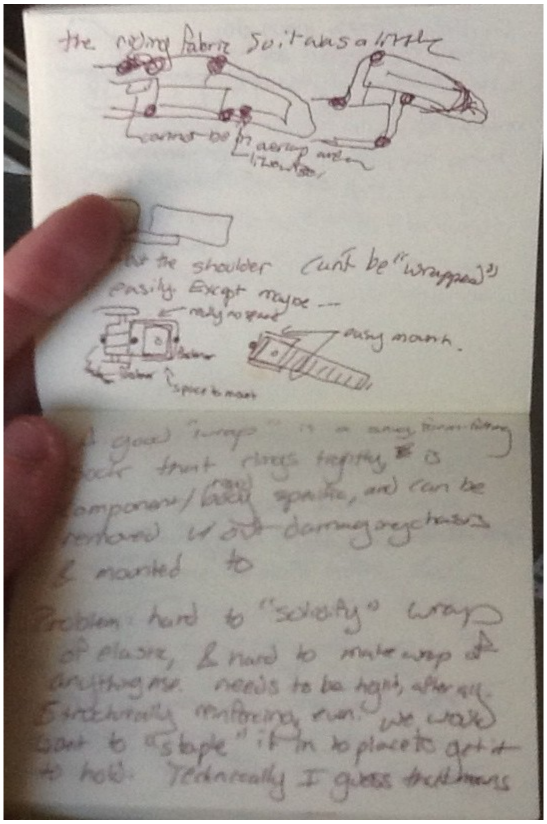
Sketching has been a design methodology for me since the very beginning of *Chloe, Part I*. Even when it did not directly relate to my understanding of logical principles or help me implement things, sketching helped me run through a gamut of interesting ideas using few words so that I didn't easily get sidetracked from expressing the idea. Capturing all of my ideas helped me separate fact from fiction in terms both of what I was capable of implementing in the allotted time frame and what I was interested in working on. Furthermore, it cleared my head of many questions by asking them and answering them on paper. Once my head was clear of those questions, I could truly design.



Sketching has also helped me in other ways. It has helped me significantly in designing complex systems and in later explaining them. Chloe's system design for *Part II* was established using a pad of paper and finalized using sketching tools. I maintain numerous notebooks, not just for written ideas but also for drawings. When trying to conceive of an external appearance for Chloe, it helps to vet 'ideal' but unfeasible appearances, such as what would happen if I added two additional legs and an abdomen.



As I established previously, this 'vetting' or 'clearing' experience is very important; but in fact it is also fun and pleasurable. I get to give life to my ideas, while at the same time getting them out of my head. And this, in turn, would lead me to a real, functional design pathway.



the riding fabric suit was a little

cannot be in a wrap anyhow

And the shoulder can't be "wrapped" easily. Except maybe ---

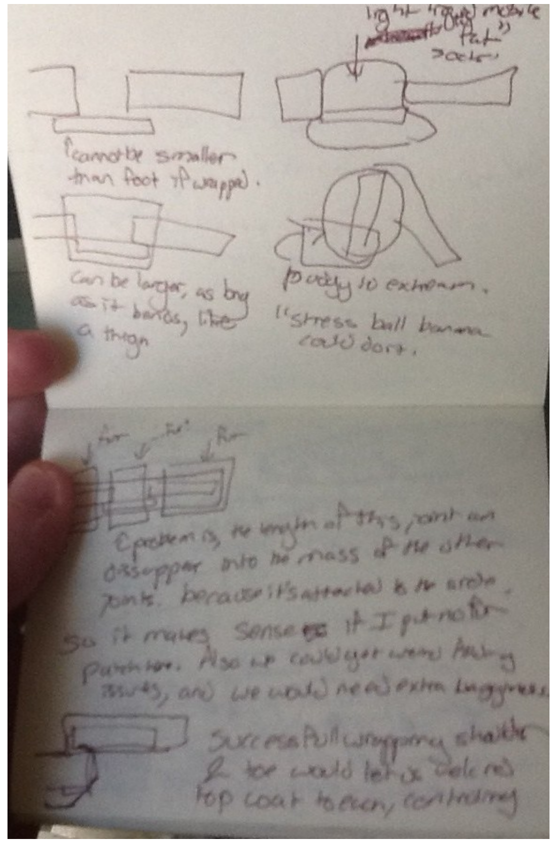
really no space

easy mount.

space to mount

A good "wrap" is a snug form-fitting sock that clings tightly to a component/body specific, and can be removed without damaging anything it's mounted to

Problem: hard to "solidify" wraps of elastic, & hard to make wrap of anything else. needs to be tight, after all. Structurally reinforcing, even. We would want to "staple" it in to place to get it to hold. Technically I guess that means



light "round" mobile "flat" socks

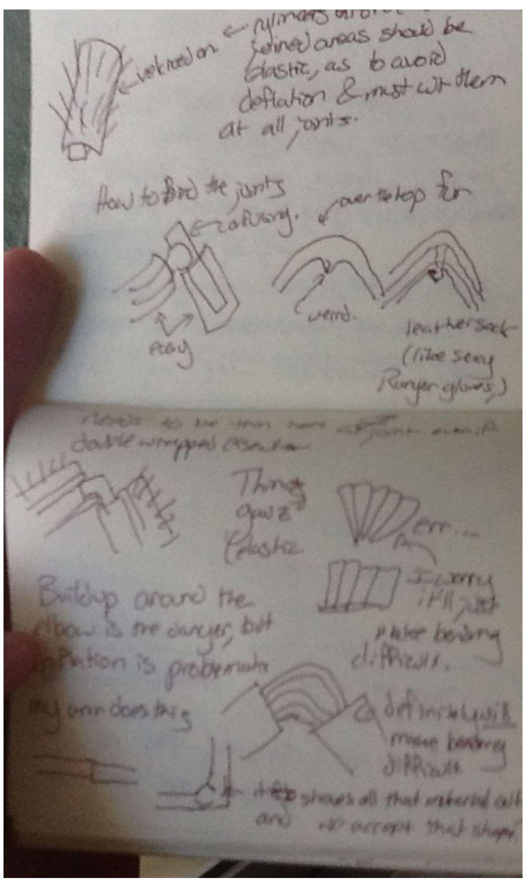
cannot be smaller than foot if wrapped.

Can be larger, as long as it bends, like a thigh

pudgy to extremum. "stress ball banana" could dent.

Problem is, the length of this joint can disappear into the mass of the other joints. because it's attached to the ankle. So it makes sense if I put no fur patch here. Also we could get weird bulging issues, and we would need extra baggyness.

Successfull wrapping shoulder & toe would let us delete top coat to each, controlling



velcroed on ← cylinders around joined areas should be elastic, as to avoid deflation & must wrap them at all joints.

How to bind the joints

← rotating.

over the top for

easy

weird.

leathersock (like sexy Ranger gloves)

needs to be thin here at joint even if double wrapped elsewhere

Thing gauze elastic

err...

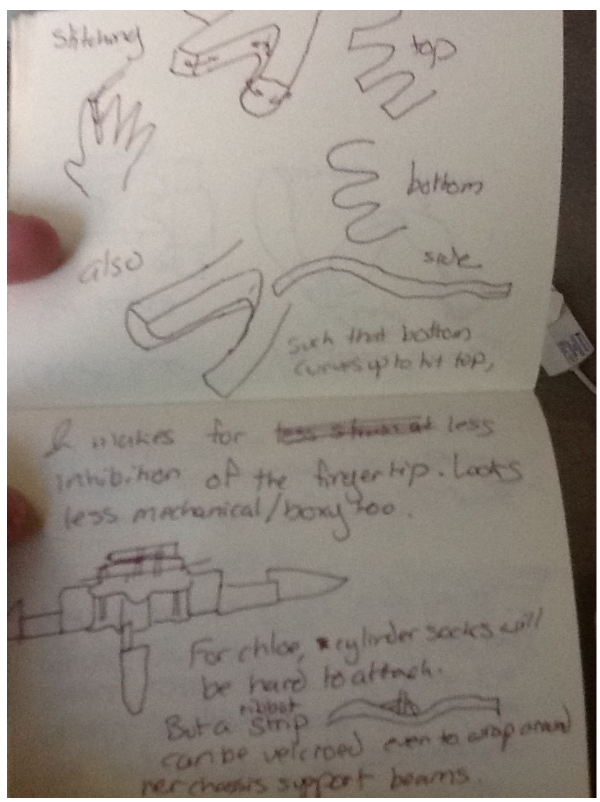
I worry it'll just make bending difficult.

Buildup around the elbow is the danger, but inflation is problematic

my arm does this

definitely will make bending difficult

it shoves all that material out and we accept that shape



stitching

top

bottom

also

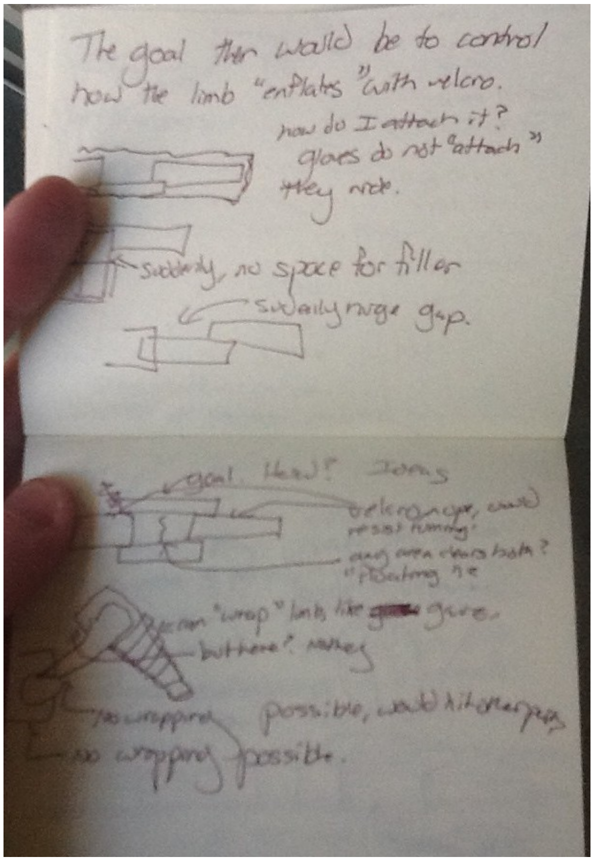
side

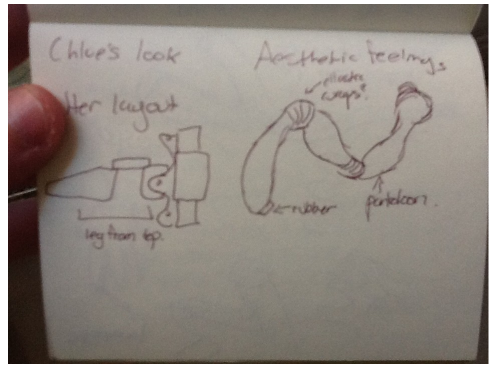
Such that bottom curves up to hit top,

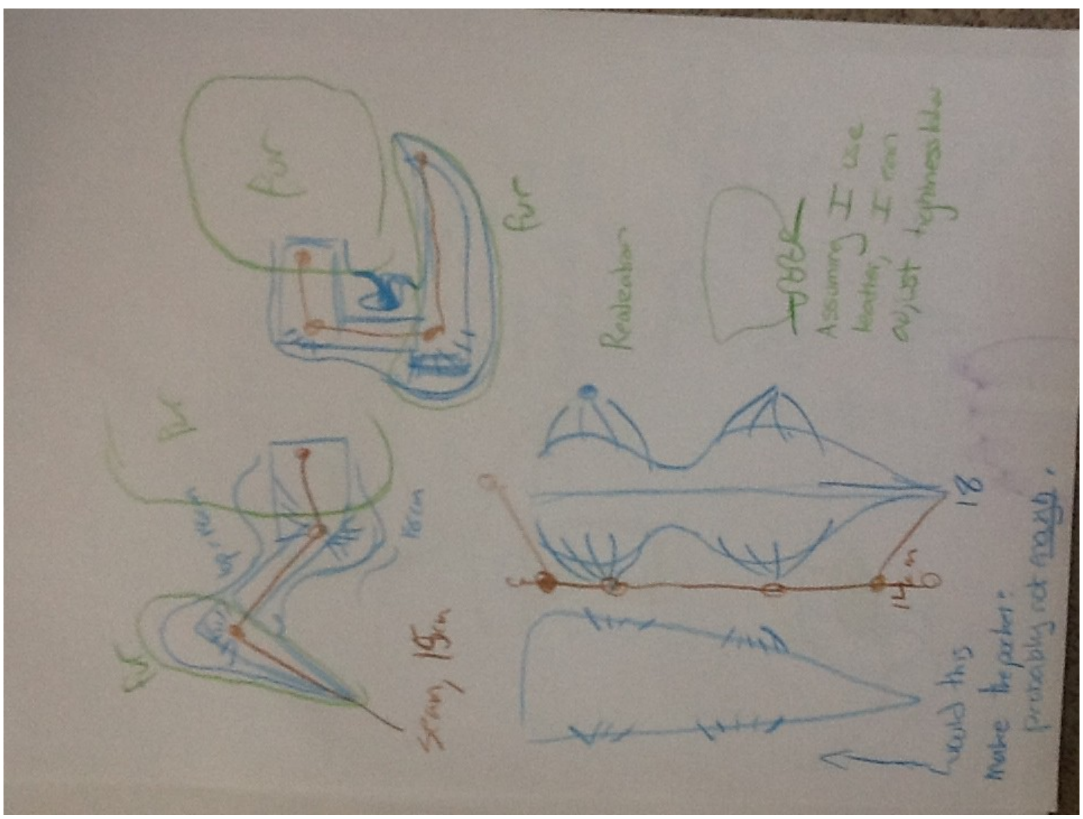
& makes for ~~less [illegible]~~ less inhibition of the fingertip. Looks less mechanical/boxy too.

For chloe, cylinder socks will be hard to attach.

But a ribbon strip can be velcroed even to wrap around her chassis support beams.


The goal then would be to control
how the limb "enflates" with velcro.
how do I attach it?
gloves do not "attach"
they ride.
suddenly, no space for filler
suddenly huge gap.
goal. How? Ideas
no wrapping possible.


Chloe's look
Aesthetic feelings
Her layout
leg from top.
rubber


fur
fur
Radiator
18
14 in
15 in

Of course, through all of the *Chloe* projects, I never found myself sketching quite so much as I did for *Part III*. This must be obvious, as *Part III* left me working on the design of material components which I wanted to be both functional and aesthetically pleasing. In fact, in order to create Chloe's coat from scratch, I had to plan out and sketch my own patterns for the material.

This shall be covered independently under 'Coat Design.'

2 Project Schedule

Chloe, Part II was planned on a much tighter time table than *Chloe, Part I.*

- 31 October – Project C assigned
- 5 November – Due: Project Ideas
- 7 November – Due: Project Pitch
- 14 November – Due: Prototype
- 18 or 19 November - Due: Presentation (Unknown to be Monday or Tuesday)
- 21 November – Due: Documentation (Inflexible, School Quarter Ends)

3 Project Log

01 November 2013 (F):

- Got up at 8:30 to apply for my own visa. Headed out, got breakfast, and then headed off. Went to visa office twice when my application was judged unfinished the first time for not having a flight to Wuxi pre-booked. Weird experience, but the second time I was cleared without a problem since I had an address in Shenzhen.
- Soldered Chloe's new rear cable, but it didn't work. Will have to get Jitao's advice as to what happened and what we can do over the weekend to get it functional. Tomorrow is SCAD open day and I have said I will come.
- Still feel like I got hit by a truck. Not feeling good (depressed) and feeling incapable. Will Moodjam.
- Asked Jose about a SIGGRAPH related event. Should probably ask him about the Hong Kong Toys and Games Festival.
- Signed up for the Hong Kong Toys and Games Festival. Cause why not? It's January 6th.
- Cooked fish heads :) But kinda had a temper tantrum about mold. Glad Travira doesn't speak much English.

02 November 2013 (A):

- Helped with SCAD Open Day and chatted with kids and parents about my school! I always love talking about my school. I hope I convince kids to come instead of scaring them away! Maybe just my friendliness is enough to make many of them come. Jose mentioned that one thing many parents is looking for is not just whether their kids are employable, but whether their 'special needs' will be tended to because kids in Hong Kong willing to pursue art have been uninterested in following more traditional paths, and the parents are are little more... concerned about them... about

whether they'll be able to learn... I watched a man behave towards his eighteen year old son as if the boy were a lot younger, because- as I imagine I am with my mother- they are just a lot more entangled than the average parent. I wish I could show them how 'odd' I was and how happy I became :)

- Shopping for Chloe at the conclusion of SCAD Open Day.
- Bought new battery in Mong Kok after running a large number of errands. It was only a few days ago that I realized that my powering solution wouldn't work and now I'm already committing to a completely new one. Well, that's rapid development for you. I need Chloe autonomous; it's part of this project's goals.
- Began soldering work for battery.
- Got battery on board.

03 November 2013 (U):

- Went Hiking. Whole day. Was a long time in coming.

04 November 2013 (M):

- Paid rent. It was a big thing because I felt like I 'hadn't gotten anything done' and I was antsy about leaving the house when I could be working. But it was important I got out of the house.
- Reading more Donald Norman and Bill Schneiderman for Information Design course. Reflection papers will be due on Tuesday.
- "remove extra steps in thinking"
- "remove actions that distract attention from the task intended, or disrupt flow"
- Studying User-Centered-Design and HCI is an amazing tool for learning to design one's own life.
- Wrote Schneiderman and Norman reflection paper.
- Have a lot due tomorrow since I missed Thursday (Halloween).
- Wrote Project C Ideas Pitch.
- Realizing I work in spurts.

05 November 2013 (T):

- Finished Chloe, Part II Final Presentation in the morning and presented.
- Presented Schnedierman and Norman reflection paper.
- Wrote a healthy chunk of code for Chloe at school, but forgot to push changes or log out. Will have to come in Wednesday. I'm basically blind-coding her console.
- Noticed an error in Flora and Fauna when I tried to demo it. Currently working on brainstorming to try and track down the cause...
- Came home and tracked down the cause of my Flora and Fauna errors at like 3 AM wednesday morning,and pushed my first commit to FAF in forever!

- Did some coding on Notepad ++ on my home computer. I was testing whether or not I could comfortably code in Visual Studio or any other platform instead of straight on my Linux virtual box, which is slightly awkward. I didn't enjoy it.

06 November 2013 (W):

- Did some brainstorming for how exactly I plan to tie everything on to Chloe. I'm concerned the BeagleBone will rub against the screws on the chassis, and I don't want to anchor it to the battery because I feel it would overheat! I might have to do that for now, however...
- I need to get coding done today. However, I also have code in Sham Shui Po that I need to push!
- Got Chloe, her Beagle, and her Battery all tied on, and filmed her standing up properly for the very first time without falling over!
- Went to Sham Shui Po, pushed, and had to merge/resolve my Chloe code.
- Spent about three more hours in SSP, coding for Chloe. Staying on task with so many abstract classes impressed even me!

07 November 2013 (R):

- Created Chloe Pitch III in the morning and pitched it for my course.
- Had a lot of strange coding repository glitches. Turns out when I coded on my Linux machine yesterday, I didn't commit changes and now I'm paying for editing the same documents both at school and on the Linux machine. Had to resolve all of those.
- Came to the realization in class today that I need to be working on Chloe's coat, not her IK. The IK implementation is an extended goal now. Even though I've spent a lot of time preparing for IK, and working with the external library will teach me a lot, I've realized that her coat is more vital to establishing my vision for her- and will teach me more- than IK.
- Ran Chloe's console for the very first time after mild debugging. It basically works! Which is impressive given that I coded the whole thing blind! It won't be quite as integrally useful to me now that I'm not focusing utterly on IK, but it is still giving me a great testbed! Readfile is the only command that's implemented that's failing right now.

08 November 2013 (F):

- I'm currently on top of or ahead of everything and I still feel nervous and jittery and like I'm not getting things done! Maybe when one's ahead its just harder to figure out where to go next. I'm a week and weekend away from presentations and I'm going to have to spend a significant chunk of that time on my speech. It's time to do what I can for Chloe :)
- I spent a large chunk of today researching IK libraries and learning about IK modeling. I realized I needed to know a lot of theory before I could even understand what the code was offering me, much less if I needed it or could use it; or how to use it. I began modeling Chloe and wading through all the math.

- I have found a library called 'RL' or simply 'Robotics Library' and I suspect this is the one I will be using. My physical notes are filled with lessons about IK modeling, and I've created a 'investigative blurb' to myself online to track what I've been learning.

09 November 2013 (A):

- I got really stressed out today, but the Captain helped me calm down and I worked rigorously into the evening on Chloe. In fact I ended up going to bed at 7 AM. Doh! I'm going to go in to SSP/Mongkok tomorrow to get pieces for Chloe's coat.
- Sent some XML files to Windows 7 to browse through them using my own tools at my leisure during debugging.
- I've done a ton of development work trying to implement the IK library. What I got so stressed over was CMake, and being unable to understand why it existed or what it was doing for me. Eventually I figured out how it fit into my already existing understanding of cross-compiling. I worked long at hard at cross-compiling and I made a lot of advances; when things didn't work I looked for workarounds, and I currently have two avenues of exploration going on. I captured these explorations in an 'investigative' text blurb.
- As I discussed with my peers while pitching Chloe, Part III, I know that today (Sunday) is the cut-off date by which I should know whether I'm capable of implementing IK or not, and whether I should default back to a backup plan of manually programming Chloe's motions by simply driving her motors the traditional way and focus instead on the coat and other vital parts of her implementation. It's a tough call to make, especially because of how much work I got done, but since I set up the rule set for how to deal with things ahead of time, it's a little easier. After all, I know this isn't the core of my necessary implementation, and that I was specifically dabbling with IK just in order to see if it would behave in a seemly manner and allow for quick implementation. That isn't so, so I'm making the call: IK won't make the cut. That said, I'm going to have to work on it after the quarter is out, because I really want to give myself the experience of successfully cross-compiling and enjoying the fruit of that labor.

10 November 2013 (U):

- It's Sunday! I have to work on Chloe's coat, which is the core of my implementation. I won't get to show of any IK code at all, so in order to illustrate progress I'm going to have to make a lot of ground to show off that same kind of work in the final coat for the prototype showing. Initially I was hoping that would be able to wait till presentation day, but no big loss. I've also been told that Michael is coming to visit Tuesday and I want to make sure I have something nice to show.
- With the Captain as my shopping partner, I tracked down some starting materials. SSP doesn't have many fabric shops open today, and we arrived late (because of when I went to sleep last 'night') but I was able to get new shoes for a meeting I have Monday with Dr. Wu, new hair clips to control my hair, and materials for Chloe including fur, elastic, thread, needles, and feathers.
- Began brainstorming Chloe's coat seriously. Drawing a lot of pictures of her and what her coat might look like. Thinking I'll sew her like a glove, with a top and bottom layer and a side strip. Drew pages and pages of ideas. Trying to figure out how I'm going to handle the space under her arm

joint, where cloth can pinch but it's impossible to put much of a 'limb' because that area becomes occluded by the shoulder and tarsus when the arm is drawn up against the body.

- I know I want Chloe to be pleasurable to touch. I want her fluffy. I'm thinking that under the fluff I should put fake leather or rubber material, or even elastic. I'm not sure what I might do with the materials I bought, so I'm experimenting with them.
- Bought Chloe little rubber feet!
- I cut a prototype of Chloe's coat using some leftover fabric before going to bed. I haven't stitched it yet, but I've realized that I cut it slightly wrong. I will have to buy more fabric after my meeting with Dr. Wu tomorrow.

11 November 2013 (M):

- Created a nice slide reel of my work on the way to Sai Kong.
- Meeting with Po Chi Wu and the Masters of Business Administration at HKUST. I road two hours from my home to go and meet them and... and only one MBA showed up! How odd. Everyone keeps telling me I need to meet MBAs and when I finally show up they do this XD. Well, Cristophe was nice and explained that the 'kids' were in a crunch time. Goofy. How frequently do tech people come and offer to supply the other half of a business team for these people? There's a whole Entrepreneurship club here for goodness sake!
- Met Steve Lee as a result of going to HKUST and he told me about their 'batting cages.' I won't be able to go to a meeting on Tuesday or next week because of SIGGRAPH and finals, but I can definitely try and go to one of the batting cages. Coming down with a cold :(
- Purchased some fabulously soft and robust brown fabric for 100 HKD. It's definitely worth it. I will serve as Chloe's new 'skin' under her fur. I also found some fur I like better, and which I've elected to buy.
- I'm really nervous that I'll get home and my sewing machine won't work. It's a cheap 75 RMB little thing and it's old and hasn't been tested recently. I need an alternative solution, so I buy five yards of leather laces and a pair of precision 'snippers,' which is what I'm calling these tiny sharp scissors used for cutting slits in fabric. In a pinch, I think I can use this setup to do something neat.
- Stopped in with my professor on the way home and talked with him to let him know how my meeting had gone and to make sure I was still on the right track with my project.
- The sewing machine failed me. I build a leg from my discarded cut of prototyped fabric, and I use a stapler to simulate stitches. The exterior actually looks quite nice- just like a glove... But I'm realizing there were some holes in my reasoning. First of all, I can't stitch this much by hand. If my sewing machine is broken, I need as few seems as possible. Second of all, with all the ends (seams) of the fabric turned inward, it greatly increases the chance that something will break on the inside, get caught in a servo, or get wound up in a gear.
- I create another prototype, this one is just some scraps of the brown fabric with slits cut in it and the leather lace strung through them. I like what I'm looking at. This will have to be my sewing substitute.

- Changed my strategy. I have to completely redesign Chloe's body and remake my patterns. I need a 'clamshell' seam where I make just a top and bottom piece which are sewn together at the sides. I go back to the drawing board and I make sure there is enough fabric. There will appear to be /too much/ fabric at all times, actually, but in fact that is how much I require in order to give Chloe her flexibility.
- I am not going to sleep tonight. I have to design the patterns, cut them, sew them, velcro them, cover them in fur, put the result on my robot, and then plug the little rubber feet into place.

12 November 2013 (T):

- Oh my God, I am so tired. But it was worth it. I got to show an extremely fluffy Chloe off not only to Jose and Ian, but to Michael :) She looks like a hat! Prototype day is a success :) Everyone loves Fluffy Chloe. I have to go to bed at like 5:00 pm this evening.
- Ate lunch with my professor and asked him about my next moves for starting my own company, and how to best follow up with Michael. He gave me a good critique and taught me some better ways I could put what I wanted into words.

13 November 2013 (W):

- Woke up this morning and broke down crying. That's what happens when you pull two all-nighters in such close succession! I woke up at 9:00; it turns out somehow my alarm clock had been deactivated and I missed the Canadian Chamber of Commerce event I was supposed to attend this morning. I turned over and went back to bed after sending a letter of apology to my teacher and bawling for nearly thirty minutes nonstop.
- Did some secretarial work to track my meetings, where I'd been, who I'd been with, and who I need to follow up with. Sent an email to Christophe asking him if we could continue our conversation after both of our finals.
- Brainstorming my other papers, writing, reading, and basically doing lower-key work on all my classes. Trying to figure out what my end deliverable for Information Design course will look like.

14 November 2013 (R):

- Worked on my speeches heavily this day. Had part of them written when I got up to speak, and then wrote the rest through a dialog with my professor, which is how I prefer to write them. He really helps me adjust and steer myself as I'm brainstorming, and I get on the right pathways and I'm able to ask questions about things I want to express or focus on or avoid and uncertainties I have. He advises me to steer away from emphasizing 'negative points.' For example, instead of saying that I started a class knowing nothing, I should say that from the start of class I learned everything. Subtle.
- Went to TEN meeting this night. Did some great networking and immediately got a follow-up from Michael, who also went, that seemed to suggest he was impressed by my peoples' skills. Good! A second chance to impress that I do know what I'm doing.

15 November 2013 (F):

- Today I had to rest a lot. Had a long week.

- Worked on my presentations. Started putting together the slides, especially for Chloe.

16 November 2013 (A):

- Brainstormed what I could do for follow-up letters, realize I need Jose's coaching.
- Worked on a neat hand-written GDD look for Agon and Alea, although I eventually discarded it. Got down a lot of my ideas.
- Something EXPLODED in Eclipse. I'm going to have to hand-repair it. Must have happened when I did a hard restart.
- Worked on IK a ton more since Chloe's fur is done, trying to get cmake to work, and modeling the joints. Downloaded a program to play with visualizing the joints. Still I know I'm not going to be able to implement IK; much of this is just for 'fun' now and because I want to get Cross-Compiling down.

17 November 2013 (U):

- Emergency! Chloe needs a head. I feel it! There's little more to do. There's no time to get her proper eyes or shop more, but now it's the little touches I'm realizing that I should be working on with what time I have left.
- She also needs to be autonomously powered. I rig up her autonomous powering solution myself, and figure out how to get her to start a program from boot. She does, for the first time ever, operate autonomously!
- I switch the battery to the underbelly and come up with a solution for mounting the Beagle on top. Chloe was once overheating, but this should help.
- I ask The Captain about a length of iron wire and whether I can use it. Then I get to work rigging an armature atop of Chloe's head
- I use a hot glue gun to fix the places where Chloe's velcro tape has come free. I also all my remaining velcro to attach a newly cut piece of top fur and a face. I make the eyes by covering a black piece of cardboard with plastic and putting it beneath fur that has two holes cut out in it. Chloe is ready to show.

18 November 2013 (M):

- I have woken up with an extra hour and a half of time to spare.
- I do some of my best secretarial management this morning. After writing down everything I'm not allowed to do, I triangulate one last thing I can do. I can make Chloe wave. I write the code out by hand in my teashop.
- For a moment I run into all sorts of problems trying to implement the code. my exploded Eclipse destroyed the settings I used to use to push code to my board. But now I'm so familiar with Linux that I know what to do. I get the program on board and I test. Chloe waves. Time to present.
- I give my presentation. Documentation is due Thursday. The rest of the week: SIGGRAPH.

VI Coat Design

Chloe was conceived of as a project in which the process and learning experience were more important than the end product. Yet the vision behind Chloe was the guiding force that informed the process, established the milestones, and was used to determine the course of action for each project and phase of development.

In this section, I highlight the aesthetic design choices that were made concerning Chloe's coat.

1 Need for a Coat

It was established that the coat would add great impact to the project, help convey the type of product I envisioned for Chloe, and teach me more about the full gamut of creating toys (and skinning them) than working with an inverse kinematics library would be capable of.

2 Inspiration

I drew inspiration from the shape of the hexapod, which appears alien or insect-like, with my desire to create a lovable, cute, and touchable pet or companion. In my mind, I needed something that crossed an insect with a cat. And in my experience, that meant I needed to look to the jumping spider for inspiration. With its tendency to tap its front legs, its large, round eyes, its apparent 'fluffiness' when viewed up close, and its 'eyelashes,' the jumping spider is a great example of an otherwise terrifying animal that can be made to appear strangely adorable. In fact, jumping spiders have their own internet memes, which emphasize their 'cute' factor in contrast with traditional feelings towards spiders.

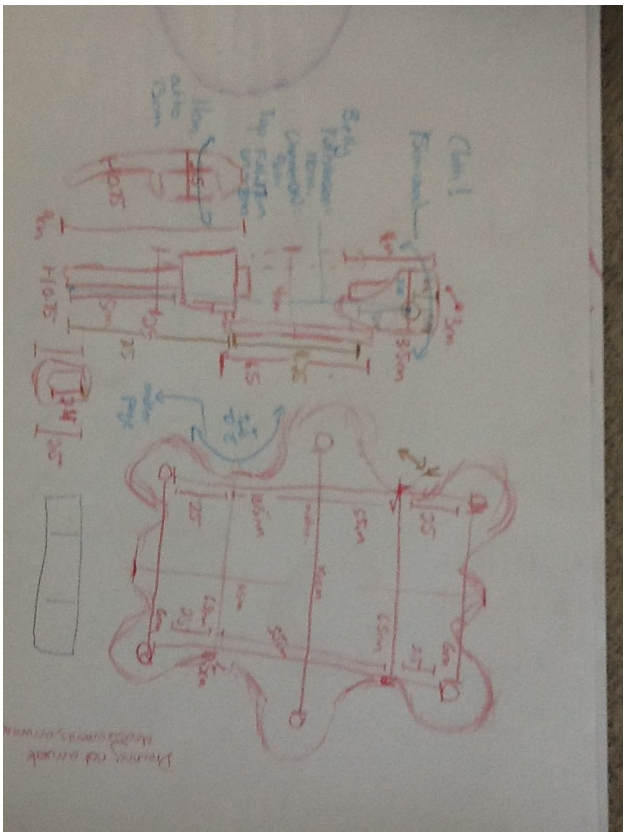
I also looked at several other insects for inspiration as to how fur could make 'unsettling' animals less intimidating when viewed up close. Moths, with their large eyes, fluffy antenna, and fluffy bodies, frequently look less intimidating than other insects and give a good idea for what a fluffy insect might look like. The following images are of pale Tussock Moths, who are ridiculously fluffy with antenna that look like large droopy rabbit ears.

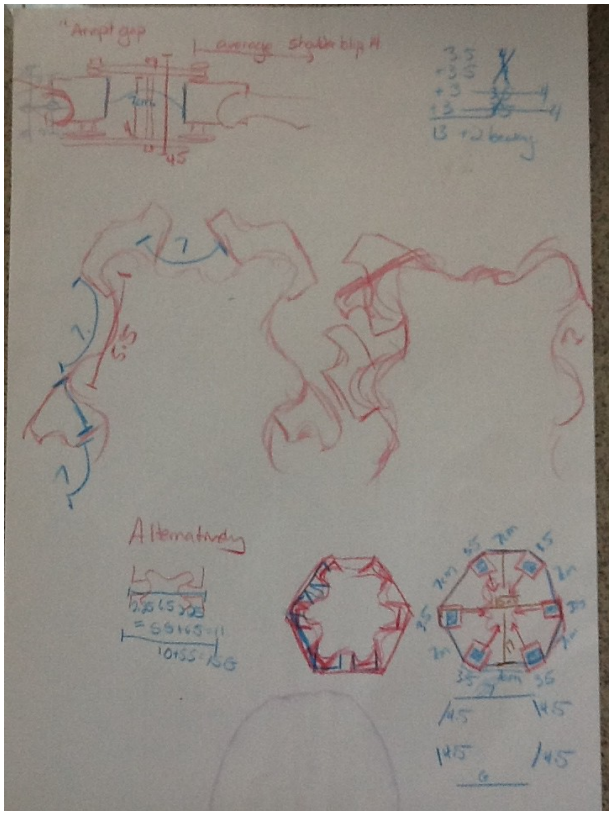
The following image is of a pussy moth, which has an even more fluffy larvae form:

I realized that these moths frequently had faces similar to an owl's face, which I also obtained a reference for.

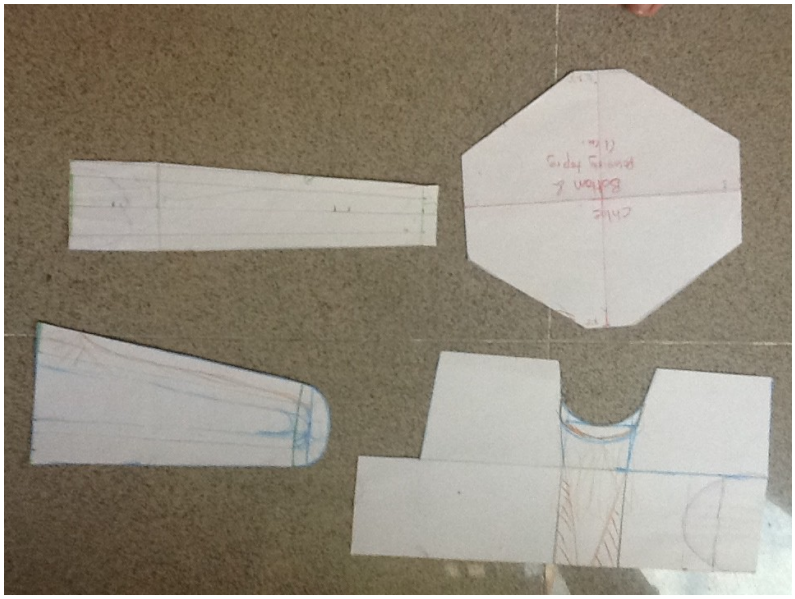
3 Planning

I conducted several quick prototypes using staples for stitches in order to try and determine the coat's form-factor. It had to allow for considerable 'stretching' in order to account for the rotating servos. I then began measuring and planning the look.





Initially I planned for the pattern to be cut like a 'glove' with a top and bottom piece and a side strip that ran between them. This, I felt, would have been optimal. However, I had to create a backup plan



because my sewing machine was untrustworthy. This second plan involved stitching together the coat using a length of leather lace, and was much more difficult. When my sewing machine gave up, I resorted to this backup technique. As I no longer had the advantage of a sewing machine to quickly make new seems, I redesigned Chloe's coat to work as a sort of 'clam shell' where there were only a top and bottom part that were stitched together at the sides.

Here, I have included an image of the patterns I used, with the discarded 'glove' pattern on the top and the clam shell pattern on the bottom. The underbelly was designed with a slit so that it could be opened and removed.

4 Production

Here I have included images of the production of Chloe's coat.

Using the pattern with a white pencil.

Cutting and stitching the coat.

Putting on the fur and feet

5 Final Product

VII Implementation and Debugging

The implementation and debugging of *Chloe, Part III* was relatively simple because the majority of *Chloe*'s showcased development was in her coat, which has already been described. Implementation could be said to be broken into six phases: Console, IK, Motion, Automation, Coat Prototype, and Coat.

1 Console

Chloe's console was implemented quickly and easily early on. Most of the console was coded blind on my tablet device or at school, and yet ran flawlessly when finally integrated. I guess that is a good indication that I still remember my basic programming. The Cconsole included commands previously listed in the extended goals of *Chloe, Part II.* At the end of *Part II*, Chloe's console had the anchor points for much functionality but was only able to list commands. By the end of *Part III*, I was able to edit, remove, and add commands and behaviors, and read and write to files.

2 IK

In order to get an idea of what IK development was like, it is important to look at the project log and at the attached appendix that goes through an investigative-styled exploration of the the themes necessary. I spent a lot of time understanding how to model robotics, and attempting to cross-compile libraries. Although I made a lot of progress, I wasn't prepared to push an IK addition by the end of the course. Most of the 'implementation' of the IK had to do with me learning more about the area I was working in, and gaining more experience in the area of cross-compiling. In the future, as I continue to work on Chloe, this will be valuable information not just for implementing the IK library, but for all future sound and vision libraries.

3 Motion

Chloe was able to stand up and walk properly by the end of *Chloe, Part III*, and she was also able to wave both of her front arms. After implementing my console and behind-the-scenes architecture, such adjustments were trivial. They were both quick and enjoyable to implement, and the minor debugging I was required to do was relatively painless owed to the fact that all commands had been abstracted to human-readable language.

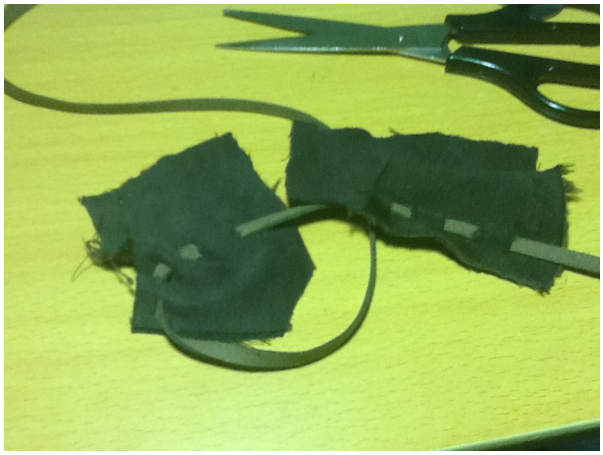
4 Automation

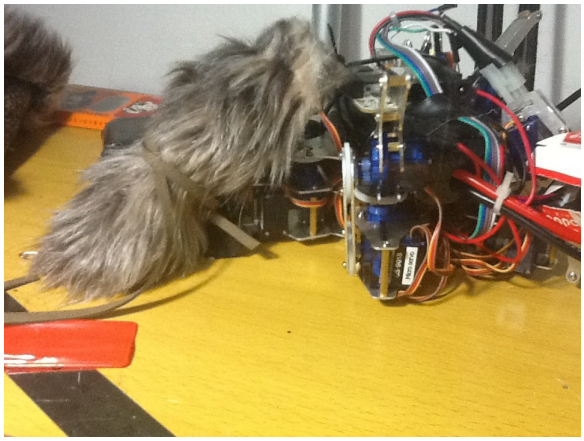
I installed a 7.4 volt battery to Chloe and then soldered in a regulator to convert part of that current down to 6.2 volts. These were the needs of my servo control board. The battery finally powered Chloe the way she was meant to be powered for the first time during all three parts of the Chloe project. By routing a 5V pin from the control board to the BeagleBone, I was able to also power the BeagleBone. Previous to doing this, I was unaware that I could 'transfer' current both ways! I was quite excited to learn this.

Tinkering with Chloe's onboard scripts in order to get her to execute her program at boot was similarly trivial. There was some confusion at first as I encountered multiple different techniques for creating boot scripts for different operating systems, boards, and schools of thought. Eventually, however, the script itself made it into a working directory, and Chloe moved under her own power without any connection to any external input device.

5 Coat Prototype

I created numerous prototypes to try and understand what kind of coat I was making. Some taught me that I needed to turn my seams outward or else risk getting fibers stuck in Chloe's motors. Others confirmed that I would be able to use my alternative clam shell stitching technique in order to hold Chloe together. A simple test at whether velcro would hold the fur in place revealed that I would need to use a glue gun to get the effect I wanted without stitches being available. Prototypes also helped me size elements appropriately. I realized that in order to have full flexibility, it was unavoidable that the cloth would look baggy and clump in some areas. This also altered ways I thought about Chloe's movements, which I realized needed to be more exaggerated and farther away from the body.





6 Coat

Most of the details of the coat implementation are listed above in coat design. It should be noted that coat development occurred in two stages. In the first, I had no plans to include Chloe's head, as I did not have the time or resources to make convincing eyes and I was uncertain how to create an internal armature that wouldn't deform. Later, however, I found her incredibly disturbing without a head, and many people commented that she was unidentifiable as an animal (she appeared to be a hat) and that they were confused by which side faced forward. In lieu of this, I constructed a wire armature over her chassis to hold up a head, and gave her a crude faceplate and feathered ears. After that, confusions abated.

VIII Conclusions

The *Chloe* projects, in all, were a very rewarding experience. It was amazing to get up in front of an audience and explain that I had only been working on her for nine weeks in total. She was definitely a confidence booster, and a phenomenal medium in which to test my new design/development methodologies and my anxiety-reducing tactics. I have no regrets about the path the project took. However, I am now keenly interested in whether or not I will be able to carry through my development methodology to remain inspired in working in her over the winter break and winter quarter.

I am learning that just because I never have worked on school projects over break before, doesn't mean I never will. And that labeling myself is counter productive. *Chloe* is and was an amazing project that carries the own motivation for working on her. She was a great tool for working out designer's fatigue, and if I don't work on her over break it should only be because I work on Flora and Fauna (my own personal project) instead.

My future goals for Chloe are to finish what I started when it came to cross-compiling. The fact that I never got a cross-compiled library to function correctly is still a sore point with me. I feel that it's something I need to do to 'complete' my round circle of physical interactive media education. I continue to want to implement an IK library, and reactions to sound.

IX Appendix A: A Note to Self on Kinematics Research

It has been very interesting trying to get started on the IK LIbrary. I've ended up skipping full implementation of the console or modeling (which is what I thought I'd be doing) to work with the IK Libraries. I want to document what I've been doing so far for the work on them.

The library I decided to work with is called RL for Robotics Library. It's used in many research projects at Universities, its pretty highly lauded, it's in C++, it's thorough, and it has what I need (I think).

It scares me for several reasons. Its bulky. IT was built by engineers. Its got a lot of parts I don't need. What if it glimps me in some way? There are decisions that need to be made quickly and tried out. Not much time can be devoted to researching, but if a mistake is made the result simply wont work; there won't be enough time to switcheroo easily. This is less a 'termios/libserial' thing and more a 'panda3d/unity3d' thing- but at the end, not the middle or beginning.

Or perhaps there will be time and I'm psyching myself out? Perhaps what I need to understand is that it is only by diving in that I will learn the problems, and the longer I take making decisions, the less time I am actually learning what blocks I'm going to reach.

I also had to get to the point after reading a lot where I remembered and re-realized this: Mathmaticians and Engineers teach a certain way. They don't document well. They don't do /words/ well.

Consider Steward's Calculus. Lessons in that book are taught from practice problem to practice problem. The problem sets, in fact, are lovingly crafted. It is like the text and math were written by two authors. The first of these two 'authors' is verbally oriented but a layman. The second cant speak English, only math. And from problem to problem, a dialog flows out from this speaker, teaching nuances, tricks, gotchas, and variants on the same melody, all in the language of logic and mathematics.

When It comes to anything that's been built by a scientists, mathematician, or researcher, I know- and I need to remember- that the language they speak best is logic. They build things for expert users only. They only really know how to communicate by creating multiple examples in which the basics of the universe are embedded in some over complicated matter so as to demonstrate more protocols with as little real functionality as possible. Worst of all they expect you to come from the point of view of already knowing everything you could ever possibly want from the library already, and the basics of how you'd implement it on your own, with the assumption is you want a well-integrated already-built implementation that you can just grab and use in your project.

Well that's not the case for me at all. I don't have the robotics background, not from math, nor engineering, nor computer science. I learn well by being taught, I don't know the terminology or how in theory I'd implement IK if I was to do it myself. I don't know how IK works, even.

I don't know the math, I don't know what functions I need, or what variables, or how or why things are organized, or what all the variables like alpha and beta and d-1 refer to, I don't know where I'd start measuring my robot, I don't even know where I'd assume the global origin to be! I can't learn how to use the API by 'wanting' a component know nothing about (much less the name) and then looking for that term in the documents, because I don't know what functionality it is this library wants to be the best at offering!

Heck, reading their explanations, I don't even know why it is they've chosen to implement anything the way they have! I don't know the names of the theories, what they were designed for, what they weren't designed for, what they compensate for, etc, etc, etc.

But. I have to start somewhere. As Schneiderman erroneously said: Most people solve large complex problems by breaking them down in a hierarchical manner into smaller and smaller problems until they become manageable. And although he's totally incorrect about how 'most people' solve problems, he's right about a good objective way of stepping back from the problem, keeping anxiety low, and tackling large, difficult, monstrous adversaries.

The Robotics Library was kind enough to provide wikipedia links to the description of the framework that it used for kinematics. I was then able to google this further, looking through slides and slides written in impossible-to-parse-math with too few annotations to digest without a previously existing background to give it context.

At last I came upon an illustrated guide that helped to break down everything. It started off with pictures, and vector and matrix math (which I understand) and built up through the whole process of creating a model and solving for that model. It was clean enough that in theory, and with a lot of time to spare figuring out all the details, I could have implemented the FK and IK solvers myself.

That taught me the theory, although after spending so much time working on it and turning it over in my head, I was a little ready to attack something and had to clean a lot, hehe.

Now, armed with the basic knowledge of what's going to be expected of me, I have to learn the API of some library and figure out the tools I have to create the same basic things I understand through my mental model. The library I've chosen to work with is the Robotics Library, as I mentioned. There are a couple reasons for this, let me scramble for them...

1. It's got what I need.
2. It's got a lot of peer review around it. Its used by universities, and has some good credentials.
3. This is in contrast to random things I find on github with a four line README in bad english that have who knows what functionality implemented to what extent or why
4. It delivers the impression it was designed to do its job correctly, with good quality, along standards that make sense to everyone else I'll be reading the papers of in the industry, without skipping anything or failing to implement any big holes.
5. It's pretty cutting edge, and if I get comfortable with using it, I predict it will do my heart good and segway into doing other neat things with robots, including work with the Pointcloud computer vision library (Which I just realized I really can truthfully run on Chloe.
6. Its already in the programming language I'm using (C/C++, the 'natural language' at which all nitty gritty low level thingies out to be written in), which means fast integration and I'll get to play with it faster and figure out if there are problems (and what to look for in another engine) sooner.
7. It was specifically designed for embedded systems, robotics, and linux/ubuntu. That's more fast integration, and targeted documentation/help.

8. It has numerous examples and tutorials, and although the documentation is a ltitle difficult to parse, there is a lot of it.
9. I checked out the License, and it's completely open source and can be used in commercial proejcts.
10. I like the modular structure of the components, which actually add other functionality I believe I will find desirable, such as quick XML loading/saving, Timers, threads, and mathematics. If only I had known about the timers previous to writing my own! At least it wasn't that hard.

Holy crap. I just realized that this library comes with what I believe is a communications API. Yup. Yup, this library can be used to handle serial communications. Holy crap. Why wasn't I looking at this more in depth earlier? Oh let's face it, I wouldn't recognize it for how valuable it was earlier. I only recognize it now! Wow. Wow, this has... I could rewrite my whole program using this, ha.

Wow, yeah. It's under rl::hal::Serial. Wow. Beautifuls. Although, looking at it, it's just a wrapper for termios. It doesn't take in strings, it takes in a const void *. See, these are the types of things you only find after you already know what API you're looking for.

Okay under hal:Device I'm seeing a ton of sensors, cameras, etc. I see a Joint PositionACtuator but its an abstract class and its implemented classes don't seem to work for it. Hmm, There's a JointTorqueACtuator. Ohhhh, I get it. /I/ am supposed to extend these classes with my own classes. Then there's some sort of engine already in place. I implement each piece, which is there for my pleasure to give me an architecture.

Perhaps perhaps perhaps. But here I am in rl::kin (kin stands for kinematics and can handle both forward and reverse to the best of my understanding). Here I see kin::element which is divided into links and joints, which can be prismatic and revolute. And based on what I've learned from my IK lesson today, the only two types of joints worth talking about in this kind of kinematics simulation are prismatic and revolute!

'kin's joints and frame don't seem to borrow from other parts. Interesting, because from what I can see, revolutes are in also mdl (rigid body kinematics/dynamics as opposed to denavit-hartenberge kinematics). Hal is called "Hardware Abstraction." I wish a little blurb was written about why the components had been segregated such and how they worked. From what I can see, Rigid body Dynamics adds another level to the kin library, and that is to account for the properties of the components, such as velocity, force, and weight. Brilliant, but likely beyond what I need

Spacial Vectors are 6D vectors that simplify the task of describing, analyzing, or calculating rigid-body dynamics says this link from the wiki. Ah, learning. Learning is good.

Alright well, I had a little hiccup getting the lib on ubuntu but I'm ready to start messing around with the samples eventually. All this functionality is beautiful... But overwhelming. the viewers are overwhelming. The options are overwhelming.

What do I neeD? Let's focus on that. I need the following:

A world origin. Joints. Links. Those things need measurements placed in them. The functionality I require from the library is actually very small: it's the calculation of all the matrices needed to solve for each joint... and some sort of end-effector or something that I can modify/translate/etc that I intend to represent the end of the foot of my bot. The next thing to do after that is to make a best guess at what I want the walk cycle to look like in terms of end effectors. Included with all of that, as the model is making its 'best guess at

movement' for me, I need something that grabs its 'instructions' and sends them so my servos will understand them.

Only the samples will show me what that means...

It's late tonight

X Appendix B: An Excerpt From Cross-Compiling the Kinematics Library

First I drew all the kinematics code into my own project. There was some weird error with templates taking two arguments. I tried to fix it using -std:c++0x and c++11 but I was unable to do so. While doing this I played around with the beaglebone.

I removed all the files installed by geddit to free up space. then I send the Cmake.txt file to my home machine. I edited it there in gedit, and then just used SCP to transfer it back.

I removed the references to zlib and inconv in demandig packages, after looking through all the kinematics code and realizing it wasn't necessary. I looked up something online when I recieved errors about "set target directory" not having enough arguments and learned that after VERSION an unset variable was defaulting to a blank space. By putting quotes around that variable, I ensured at least an empty string was read in.

As I kept working I got an error that LIBRARY was not specified correctly for the TARGET. I surfed up to the rl directory and looked to see what variables were set in the cmake files there. I found that the directory was /rl. Wherever I saw this variable in kin's cmake file, I replaced it with /rl, no quotes needed.

Progress. I cmake and there are no errors. Huzzah.

Okay, now what? Is the library installed? I guess it's not. I run make only to find that things crash on Frame.o and demand rl/math/transform.h. Hmm. For a bit I'm confused. I know, for example, that this is an #include <rl/math/transform.h> statement, which means I have to guess how the interior of the compiler works and where it's going to be looking for libs. Decisions, decisions, where and what to try next...

Wait, i can specify include directories, can't I? I go into my cmakelists.txt and I specify to look in usr/include. I feel that this should be 'normal' behavior, but then maybe since CMake is designed to work anywhere, usr/include/ isn't normal for anyone but Linux and cmake doesn't realize its on a linux machine or know where to find defaults that Eclipse already sets for itself and any other IDE would set.

Hmm, still no dice.

Surfing around into usr/include to see if I understand where my previously installed libs should be, I realize much to my surprise that there is no rl folder in there. Now I'm not an expert, but to my understanding math, util, and xml are all just straight header files. That's why it was so easy to cmake and then make and make install them; I didn't have to worry about any real compilation. In fact, far as I understood it, it just copied the headers files to... to...

Wait a minute. Maybe this is an 'empty string' thing where cmake made the files, and installed them, but they ended up 'instaling' over themselves in their own folder. Basically installing to the folder cmake ran in. On top of themselves.

Okay but basically to 'install' header files on linux just means to throw their .h files into a folder that they default into in usr/include right? So I make a folder there with mkdir called rl and I cp -R math, util, and xml into that directory. Later I get confused and disheartened because it still can't find them.

I copy the header files into kin/rl/math,util,xml hoping it will at least look in its own directory, but no, this is a #include <> not a #include "" Maybe i can put .. into the search directories? I don't try that because something else occurs to me.

I take one last look in math. And I find another math folder inside it. Aha! I accidentally mkdir math and then cp -R to that folder, and that must have thrown one folder inside the other instead of supplanting what already existed. I put math in the correct position.

Progress! usr/include leads the way, and math is successfully found! next we have an error with finding Eigen, and I realize NONE of the paths are specified because the variables were supposed to be set in rl's Cmakelist.txt. So I put in usr/include/eigen3 and usr/include/boost and usr/include/libxml

Holy monkey butts it's running. It's at Kinematics.o right now, and seems 'stuck.' Can't figure out if this is because compilation just takes an awful long time on this tiny board (it took a long time just to compile .o's for things that were basically empty classes) or if it's hanging. I'll give it some time.

I appear to have hit on a strange error. I got the following message: C++:internal compiler error: Killed (program cc1plus) Please submit a full bug report, with preprocessed source if appropriate. See file://usr/share/doc/gcc-4.7/README.Bugs> for instructions. The error was in Kinematics.o

Could this be the same pair<T,T> error I was having earlier on the main machine? Do I have gcc-4.7 installed? lol. I go check it out and find out that gcc-4.7 exists, although g++-4.7 and C++11 (the yummies in question when I was trying to bring the kinematics lib into my program raw) needs a repository and 60 extra megs of space. I can do that, but I'll do it on my development machine and not the bone just yet. It's still working on Kinematics.o and I'm pretty sure it got stuck again.

I interupted it on kinematics.o and tried to install gcc-4.7 It's already installed, so no dice.

Lol just realized the beaglebone wasn't plugged into the itnernet. Plugging it in now. tried to install gcc-4.7 again, meh, didn't work. Trying to add the repository for g++-4.7 in the hopes I might be able to install it. But before I throw aroun 60 additional megs, maybe I should see if the new compiler relaxes eclipse on my main machine?

Lol I'm retarded, I would need -arm-linux-gnueabi-g++-4.7

Such a thing doesn't exist yet. But on the beagle, there could be hope. Let's try it. We'll record what to remove. Can't hurt, after all. It says it's only going to require 316 additional kb or disk space. I doubt my salvation could be in such a tiny bundle, but we'll see! After all, at this point there is really nothing to lose. I'm just playing around with what time I have left to see if what I want is possible. And if it's not, I don't wmakorry about it. Chloe walks after all, and my project is essentially complete.

Forgot to designate the compiler. It's hung on Kinematics.o again.

Lol! apparently when I used apt-get install on gcc and g++ I got gcc4.4 instead of 4.6, but I already had gcc 4.7 all along for some reason. I also have g++ 4.4 which apparently was what I was using up until this moment. Not sure why that is... but I think I'll install gcc 4.6 and g++ 4.6 just for whatever and rm the old 4.4 installations. It's only a few Megs and I'm getting less bothered about what I install as long as I remember to remove later.

libstdc++6-5.6-dev came with g++-4.6. gcc came with cpp-4.6 and gcc-4.6-base

I'm looking up 'internal compiler error' on the internets.

Symptom:

fatal error C1001: INTERNAL COMPILER ERROR

Cause:

The compiler exceeds the allocated memory limit

Resolution:

1) remove template class header files from precompiled header file

2) Use 'automatic use of precompiled headers' (/yx switch) or 'not using precompiled header'

3) use compiler switch /Zm#nn to increase heap memory allocation limit.

I haven't tried any of these yet, but its interesting (though its in microsoft, might not be using G++)

I just cannot get past that kinematics.o. Nope, it's still not working. I have tried a variety of things since then; mostly looking at the C++ code to see what might be modifiable. Unfortunately it seems that pair is simply unwilling to cooperate for reasons I just don't exactly understand. I tried things like setting

Trying to get somewhere with Kinematics.o. The only way to go from here is to fix other problems and hope that deep down RL knew what they were doing and that i have the right libraries. Realizing that the console was actually dying at -std=c++11 or whatever that was, and so the 'build' was never hitting through Kinematics.h and perhaps resolving some of those errors. I realized too that I only had part of boost isntalled, and not things like boost-graph which is actually needed. I also didn't have any lib inclusions at all for boost, which might have been creating the errors.

Then i started getting errors in external resource Math that it couldn't find Eigen. So I created a symbolic link to eigen3/Eigen using keyname Eigen in usr/include. I'm about to do the same for libxml2. The beagle has been trying to compile kinematics.o for over an hour. I can't tell if this is because it's hanging on something it's unable to communicate to me, or if kinematics.o, which is seriously larger than all the other files, is simply that big of a task to the little guy.

Okay everything I asked it to link, linked. We still have a moat load of errors, but It's interesting to look at... I'm thinking it could be the linking phase that's going awry on the beaglebone too. After all, I specify the necessary libraries in the Cmake file.

I'm starting to remember that for a significant number of errors out there in the world, it is necessary to fix the ones the compiler smashes into earliest because sometimes it just randomly fixes the ones that come later. For instance, i fixed the std error and now all sorts of new problems are cropping up I can technically try to fix.

The next error is a linking one: cannot find -lboost and cannot find -lboost-graph. Let's see what I can do to fix that.

Ah, let's try boost_graph.

/usr/lib/libbost_graph.so: file not recognized: File format not recognized.

Okay, I need a .o? I'm guessing based on what my linux is compiling!

No, wait! I got it! It's libboost itself! I need to install the raw source files and compile them for usr/arm-linux-gnueabi/lib using my own cross compiler!

(It should be noted that I installed libboost-all-dev (which includes the graph, and was the component I recently installed to my host machine) and then tried to compile again. It is still hanging. I am going to put in the Eigen and other symbolic links. Maybe there's a way to make make verbose?

make -d for debugging information. After Builing CXX object Kinematics.o, I see "Reaping winning child (numbers I don't feel like repeating,b ut are probably memory?) PID 3725

Live child (same numbers) /((Path to)Kinematics.o/) PID 3727

Then it hangs.

Alright I'm extracting my newly downloaded boost source. I'm going to ahve to cross compile it, which I'm pretty sure is as simple as... Hmm. Actually I've forgotten XD I've done this once before haven't I?

Okay, let's see what I see. # ./configure --host=arm-linux --prefix=/opt/arcom/arm-linux

I found a guide specifically for boost. It says first, run ./boostrap.sh. Then modify the configuration file, which should be project-build.jam. change gcc to gcc | arm | arm-linux-gnueabi-g++;

Doh. Failed to get it to work, and then messed everything up. Actually it's possible I was mid actually executing commands correctly, wrote a command I thought was incorrect and then ^Ced to get out and screwed everything up...

Holy crap. I ran /b2 (not /bjam like my tutorial said) and awhile after a line I thought said 'sorry i can't do it' (link.jam missing) stuff started appearing. It really is trying to install. Let's see what happens. I should note I used ./boostrap /usr/arm-linux-gnueabi/lib as my command instead of just ./bootstrap. It should go over what I need it to...

Meanwhile, 2 hours later, kinematics.o is still attempting to build :3 I don't think its ever going to make it XD.

myself: I knew when I started on Chloe today that there was a good chance I wasn't going to get to where I wanted to be. I accepted that. I did it because I had the extra time (Chloe basically does everything she was advertised to do, except she's not autonomous, something I have to fix this weekend).

I did it because I had explorations left to do and things left to try. I did it because it was a great learning experience and honestly speaking it was very good for me. I did it because it wasn't good enough to just write the IK library myself; I had to learn how to import these big unwieldy library constructs if I ever wanted to get to the point where I could implement sound. This is just another important step in my learning process. It's okay that I didn't get quite to where I wanted to be. I knew it would go down somewhat like this. I'm also really happy about how far I got.

I'm aborting kinematics.

Before I surrender, I'm curious about one last thing that I want to record. What again all happened when I tried to cmake on all of rl? Oh yes, it wanted me to track down tons of libraries like bullet and coin that I was unwilling to dl.

I should record where I'm at. Right now I'm trying to cross-compile boost libraries to work with my eclipse. It's the only library I had to cross compile that rl was dependent on for kin. In theory, if it worked, my strange errors in Kinematics might disappear.

HA! It's 3:30 and I'm regretting staying up, but I finally cross-compiled boost for the first time such that all the files ended up in the correct folders in usr/

Now I'm seeing all sorts of 'undefined reference to 'xmlFreeProp' which says to me I need to include all the libxml2 directories :) I'm Learning!