University of Regina

Project

Fancy Bubble Bot

Course:	CS 207 – Building Interactive Gadgets
Lecturer:	Trevor Tomesh
Author:	Matthias Zehnder, moz193@uregina.ca
Semester:	Fall 2017
Due date:	18. December 2017

TABLE OF CONTENTS

INTRODUCTION

This paper is the documentation of the semester project for the CS 207 – Building Interactive Gadgets Course. The construction of an automatic bubble machine combines the appliance of the learned theory with a funny output. A bubble machine produces soap bubbles that delight and fascinate children and adults equally.

The basic components for this project are two Servos, a DC-motor, a multicolored LED and a normal LED. The servos build a movable arm, which is able to turn and to dip a ring in soapy water. The DC-motor is used to create an air stream to blow the bubbles. The job of the LEDs is to color up the whole project.

There have been made a few changes in the project since the proposal was submitted. This includes using one multicolor LED instead of LED-Stripes, and also for the laser a normal LED was used for this project. The main reason for those changes was to cut costs, because the end product will only a temporary state and those LEDs were already available. Another change was code based. Because the DC-motor from the kit was used and the whole machine is powered only by the Arduino, it was not possible to have different fan speeds. But the details will be further discussed in the chapter setbacks and failure.

INSPIRATIONS

This bubble machine is not the first of its kind. If bubble machine is entered into any kind of search database, there are hundreds of different results and instructions. Different kinds of material and also different kinds of implementation were seen but nearly all of them just have a turning wheel and a fan which constantly blows air. But what quickly comes to mind is that most do not have a moveable arm. The moveable arm makes this project a little bit more difficult but also more interesting.

The main inspiration for this project was given by Bernard Katz Glass, who described his version, the Bubblesteen Bubble Machine, on the website www.instructables.com (Katz Glass, 2010). The user manual is written in a funny way and which created the curiosity to try it too. Although all materials and construction were described, this project did not follow those instructions. The only parts that were used is the idea as well as the code as a base. This project was implemented with existing material at the lowest possible costs. Instead of the Arduino Duemilanove, a motor shield, a diamond plate and a nice propeller, the Fancy Bubble Bot uses the Arduino Uno, a simple plastic cake box and all kind of other low cost material.

The output of both projects is the same, an automatic bubble machine. The inspiration (A) and the final product (B) can be inspected in the appendix. The difference is in the detail. With the use of a transparent plastic box, the factor light could be incorporated nicely into this project by the use of a multicolor LED.

DESIGNING PROCESS

Before starting to purchase the materials, some basic considerations had to be made. For example, how to get all the pieces on the Arduino Uno or the breadboard. The pieces were figured out pretty fast. Due to the aim, to power the machine only through the Arduino itself, in addition to the DC-motor, two servos, the multicolor LED and the normal LED, it needed a diode, a transistor and several resistors.

The DC-motor moves because of magnetism and can generate power ('Lab 5: Motors', n.d.). "When a motor is turned off the inductor turns its magnetic field back into electricity flowing in the same direction as the previous current. This happens very quickly and the voltage can be very high - up to 10 times of the source voltage" ('Lab 5: Motors', n.d.). This is the reason why a Diode has to be added to protect the other parts in the circuit. To ensure, that the motor gets enough power, an NPN transistor has been added too. "NPN stands for negative positive negative which is their internal silicone structure. This current can only flow in one direction. The positive side of this flow is the collector and the negative side is the emitter. A small amount of current applied to the base can allow very large amounts of current to flow from the emitter to the collector" ('Lab 5: Motors', n.d.). Both LEDs need 560 Ω resistors, that reduces the current flowing through them. Otherwise there is a chance that they burn out very quickly. The servos don't need any modifications to the current. They can just be plugged into the breadboard and be addressed by code.

To get an overview how it should look like, the first step was to create a schematic model as seen in Appendix C. Based on this schematic, the next step was to build the electronic circuit which is available in the Appendix D.

BUILDING PROCESS

The basis is a transparent plastic box. To build the fan holder, first the small metal angle was screwed to the bigger one. Then this construct was screwed on one side of the box. On top of the smaller angle the DC-motor got attached with poster putty. The propeller is made out of multiple layer of duct tape. The LED, which should represent a laser got attached on top of the motor. It was important, that it points in the right direction to illuminate the bubbles. Then it was time to build the moveable arm with the servos. Therefore, the “panServo”, being the lower servo, was rotated 180 degrees, and the “tiltServo”, being the upper one, was rotated 90 degrees. Then they were glued together with poster putty. With duct tape, the ring for the bubbles was added to the tiltServo. The last step was to stick the moveable arm onto the box, which has been done with poster putty too. It was essential, that the moveable arm is not disturbed in its rotations but still close enough to the fan, that the motor can create bubbles. Due to the fact, that the wires are too short, jumper wires with a self-built aluminum foil duct tape adapter were used connect the servos, the DC-motor and also the LED to the breadboard. On the side of the plastic box a whole was made for all wires. It is pretty reliable that a wet microcontroller doesn't work very well. With this set up the Arduino and the electronic parts are protected from the bubble soap. The last step was to connect everything to the breadboard and to the Arduino as described on the schematic in the Appendix C.

This completed the physical part, which was the starting point for the digital part. The first thing was to address the servos. To use a servo, the Servo.h library has to be included. A library is a collection of functions bundled together in a package. For each servo, an object has to be created, which then can use all the functions from the library. With for-loops and the functions attach() and write(), the servos were able to do the correct movements. The DC-motor and the LEDs didn’t need any special treatments. Firstly, for each pin a constant was defined and in the setup function, all pins were set to output pins. For the motor a motorOnThenOff() function was defined. It turns on the LED on top of the motor, turns the motor on for 2 seconds before turning off the motor and the LED again. For the multicolor LED a changeLedColor() function was created. Inside this function a second one called hsv2rgb() is called. This function was copied from the lab 9 and not was not programmed by the author itself ('Lab 9: Light and Sound and Storage', n.d.). This function converts HSV colors, an alternative representation to the RGB color model, to the red, green and blue representation which is compatible with the multicolor LED ('Lab 9: Light and Sound and Storage', n.d.). With the use of this, it was simple to alternate through all the colors. More information about the source code can be found in Appendix E.

SETBACKS AND FAILURES

In the building process, the first issue that came up, was that the wires from the DC-motor, the LED and the servos were too short. It took several tries to get a working solution. The adapter out of aluminum foil and duct tape finally worked.

Another big issue was the DC-motor and the propeller to make a strong enough gust of wind to create bubbles. In contrast to the model of Bernard Katz Glass, a simple DC-motor without any special libraries and only powered by the Arduino was used (Katz Glass, 2010). Therefore, it was not possible to produce different speed levels, which means that one of the novelties could not been achieved. Anyway, the power of the engine was barely enough to produce soap bubbles. Propellers made of plastic, cardboard, paper, foil and finally duct tape in various shapes were produced to get it work but none of them worked. The working solution was several layers of duct tape in a four-bladed propeller. This was light and strong enough to create bubbles, when the angle to the ring was set correctly.

The idea to illuminate the bubbles with a laser was good. Unfortunately, the laser pointer that was initially thought to enable this feature was destroyed when disassembled. As an alternative, a normal LED was used due to time reasons and to cut the costs. This proved, that the concept would work, but the LED was not bright enough to illuminate the bubbles and was only unicolored.

MILESTONES

In the proposal following milestones where set:

Milestone #	Date	What
1	5th November 2017	All parts gathered
2	15th November 2017	The physical parts assembled
3	22nd November 2017	All the coding finished
4	26th November 2017	Finished all testing, documentation, and cleaned-up the git-repository.
5	27th November 2017	Presentation, hand-in the Project.

Unfortunately, only the first one of these were reached in time. The main reason for this was bad planning and not executing the planned things. The designing process of this project was started in the last November week. The assembling and the coding were done after the end of the semester, during the exam preparation phase. The bubble machine may could have more features and could be improved. Nonetheless, the project should still be considered a success because the learned theory from class was implemented in working solution, that produces colorful bubbles. Thus, the main goal was achieved.

USER MANUAL

The use of the Fancy Bubble Machine is very easy. Upload the source code onto an Arduino Uno. Refer to Appendix E for information where to find the code. Once the code is uploaded on to the Arduino fill the bowl two thirds full with bubble soap. Then connect the Arduino to a power supply, either use the USB-port or with an AC-DC adapter. Enjoy the automatic produced bubbles. When the lights in the room are turned off, the color effect is even nicer.

CONCLUSION

To build the Fancy Bubble Machine was a lot of fun. It was very interesting to see if the theory and the concepts, which were previously made, really will work in practice. It felt amazing when the robot whirled bubbles into the air for the first time. On the other hand, there were moments that were frustrating. For example, when everything was set up and the output was just nothing. It needed patience and the help of friends to find new ways and solutions to make it run.

Due to transport limitations, this project will be disassembled and will not be processed any further. But the ideas, concepts and the applied theory behind it will stay. This was the first project of its kind. The inspiration to try out another, more complex project is definitely given.

REFERENCES

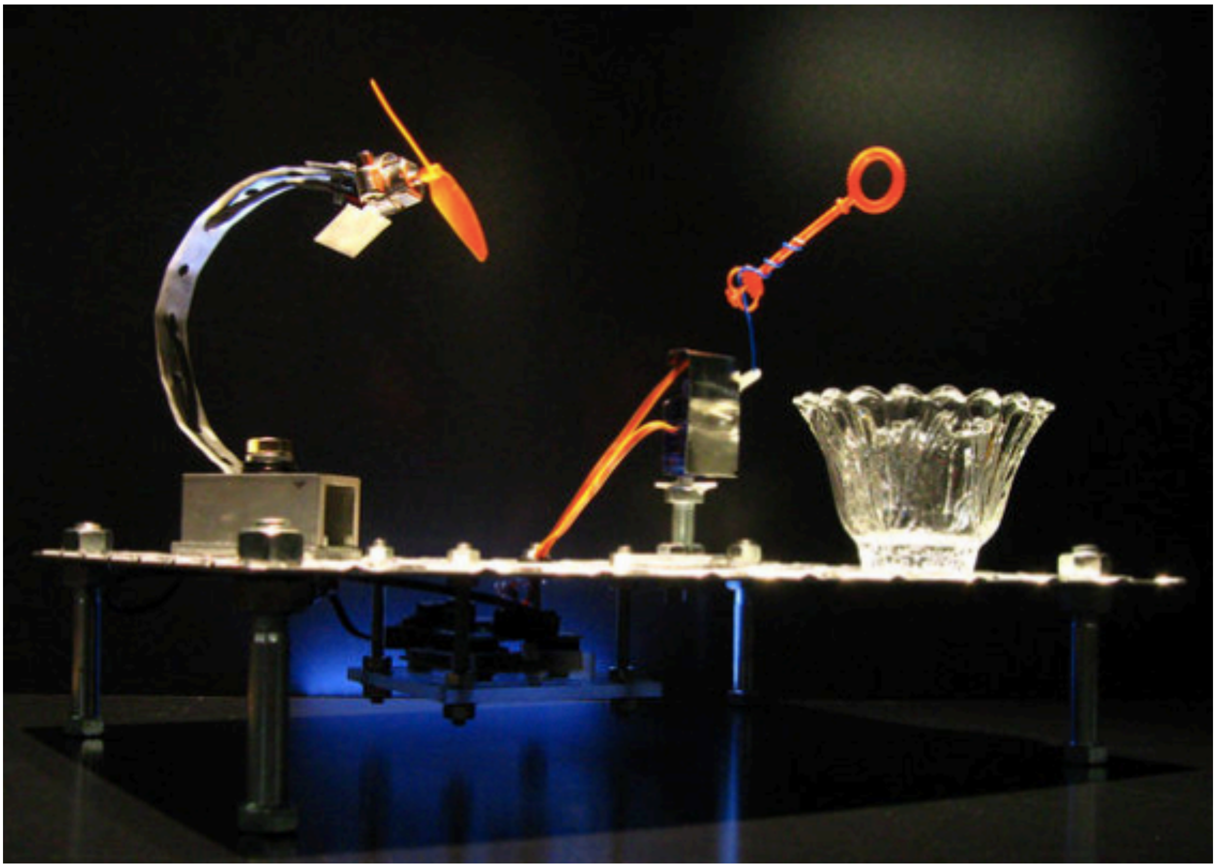
Katz Glass, B. (2010). Bubblesteen Bubble Machine. Retrieved 29 October 2017, from https://www.instructables.com/id/Bubblesteen-Bubble-Machine/#step1

Lab 5: Motors. (n.d.). Retrieved 22 November 2017, from http://www.cs.uregina.ca/Links/class-info/207/Lab5/

Lab 9: Light and Sound and Storage. (n.d.). Retrieved 27 November 2017, from http://www.cs.uregina.ca/Links/class-info/207/Lab9/

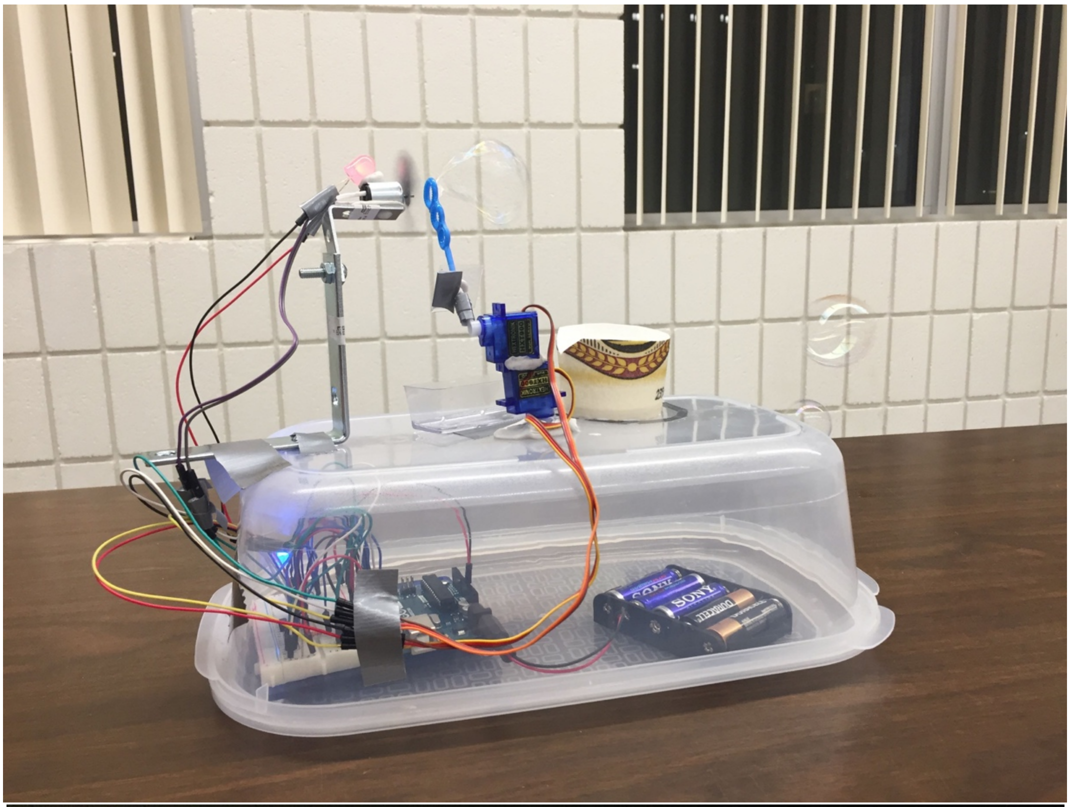
APPENDIX

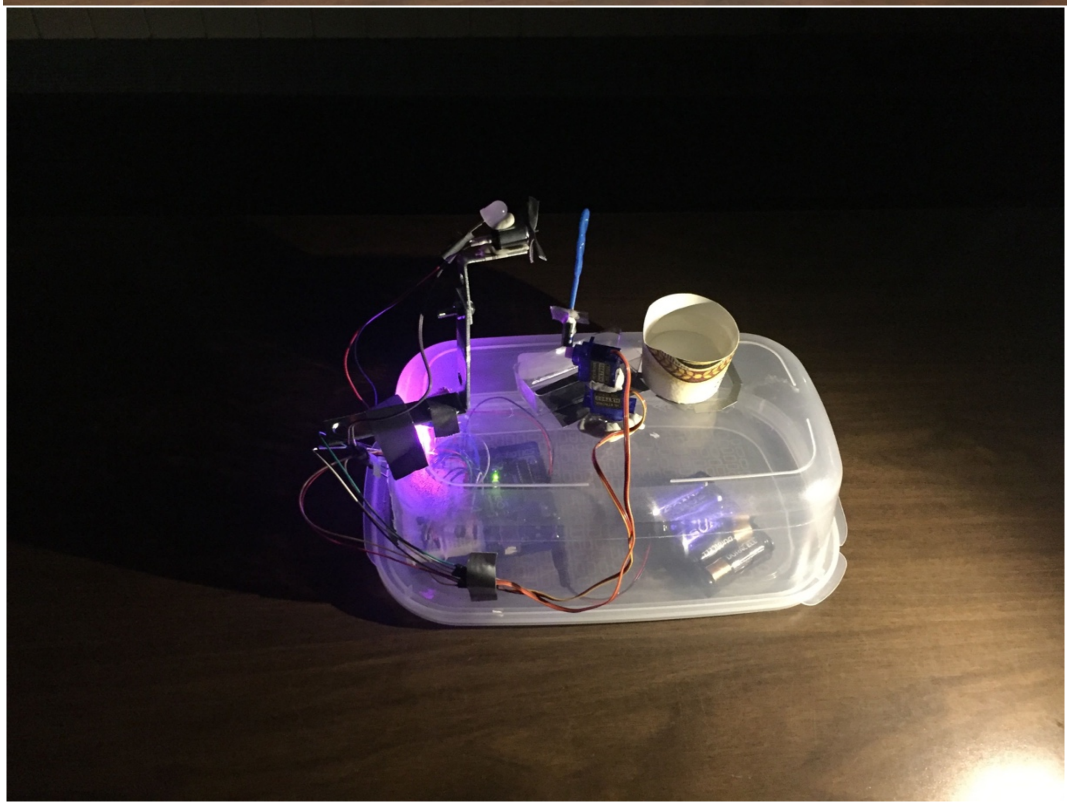
Appendix A: Inspiration for the Project



This is the Bubblesteen Bubble Machine by Bernard Katz Glass, which gave me the inspiration to build this project.

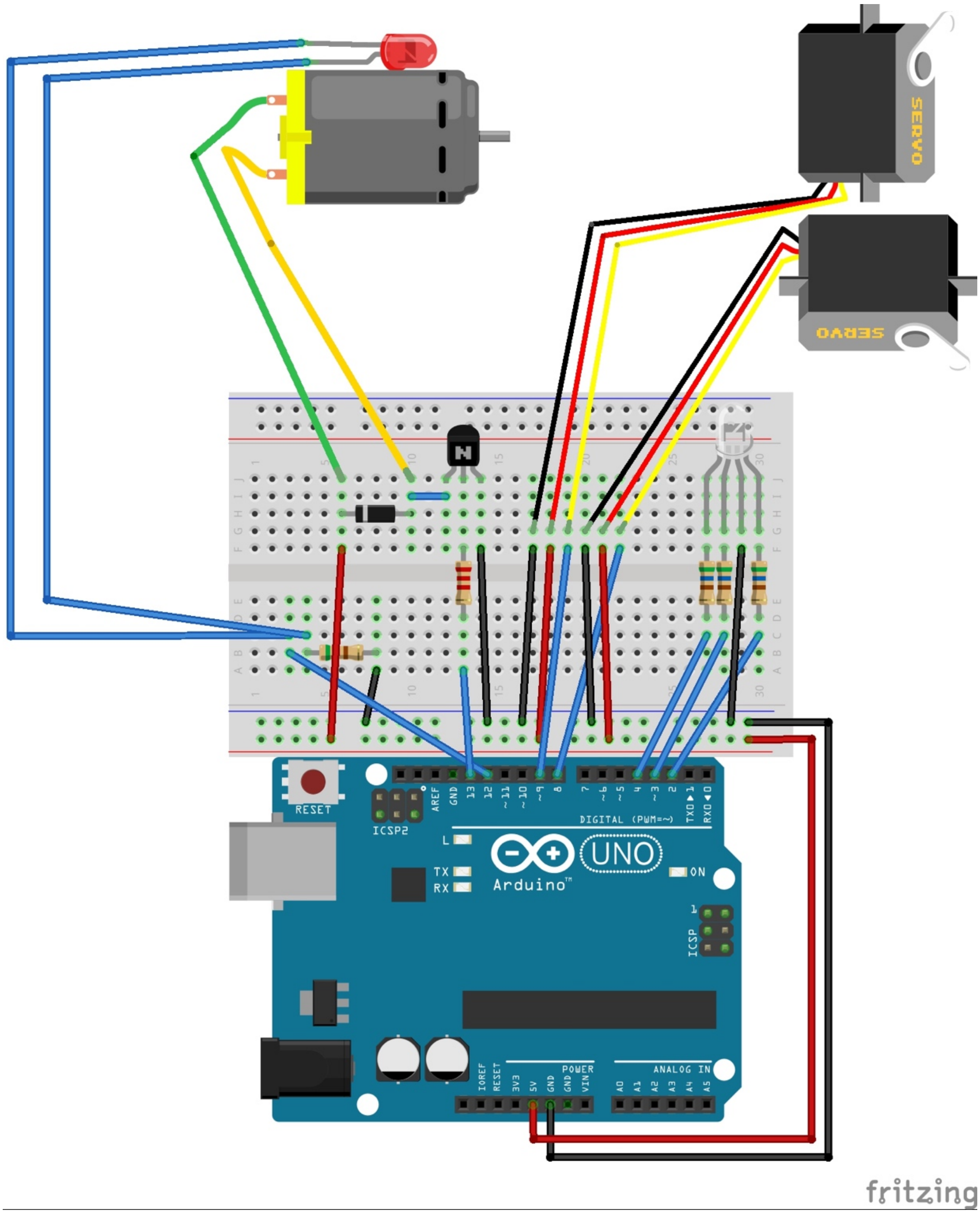
Appendix B: Final Build





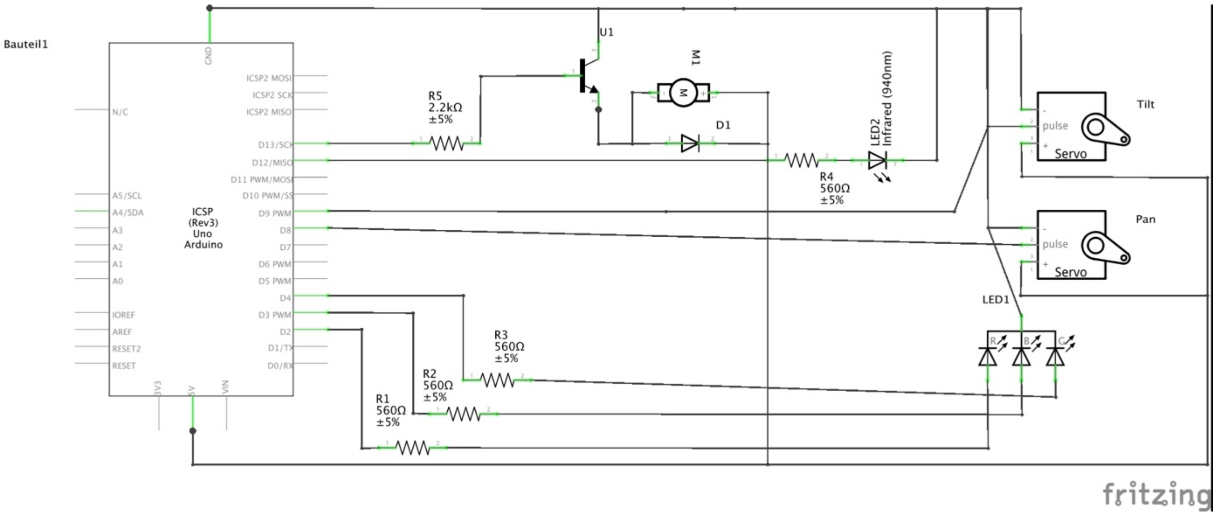
The final product with and without light.

Appendix C: Schematic



The schematic representation of all parts, created with fritzing.

Appendix D: Circuit



This is the circuit of all parts, created with fritzing.

Appendix E: Code Listings

All code useful to this project can be found on the Github repository located at

https://github.com/Risnar/CS207SemesterProject.