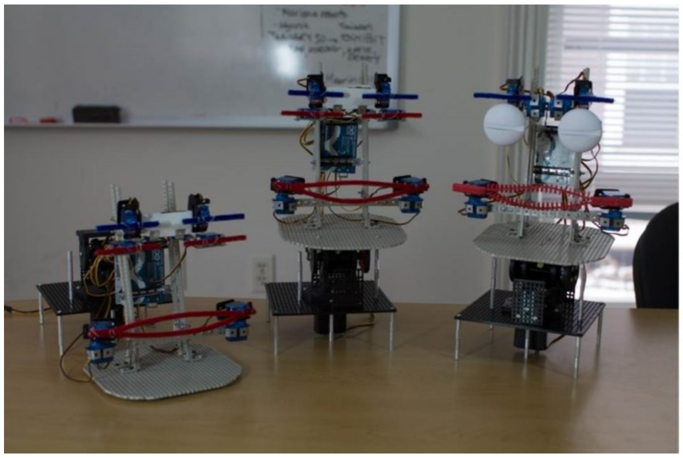
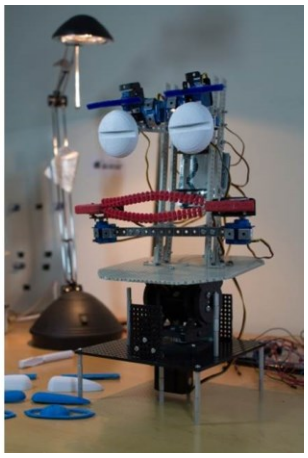
MiRAE (Robot-Face) Construction Manual

Casey Bennett

Sept. 4, 2012

www.CaseyBennett.com

Outline

**Note that there are 3D printer schematics available online as a separate file that will allow you to fully 3D print out the robot face/head (with some minor assembly), averse to physically constructing it

I. Construction Overview and Philosophy

The aim of the robot construction was to build a minimalist robotic face that could communicate human-like facial expressions with identification-rates in humans interacting with the robot. Design principles for the face were taken from computer vision research over the last decade having computers identify human facial expressions. The salient minimal features of such identification were "flipped" back around to create a robotic face, operating on the hypothesis that these same features must be the critical aspects of facial expression that humans are using to identify such expressions and that much of the movement in the face (including individual idiosyncrasies) is superfluous to the actual non-verbal communication. Details of this part have been described in depth elsewhere [Casey C. Bennett and Selma Sabanovic (2014) "Deriving Minimal Features for Human-like Facial Expressions in Robotic Faces." *International Journal of Social Robotics*].

Beyond these bio-mechanical principles, we had a number of broader *scientific* design goals focused on how best to approach the *science* of designing a robotic face. Principally, these goals captured certain desirable characteristics relative to a sound scientific approach, where the hallmark of science is replicable experimentation. In this vein, the focus was on creating an inexpensive, replicable design that could be repeated in other labs, if so desired. As such, the final design cost approximately $150 to build in its basic form, although various optional add-ons could raise the cost to $200-$300 dollars (e.g. higher quality servos, added neck motion, etc.). A prototype could be constructed in approximately 4-6 hours. Parts utilized were easily available either via Internet suppliers or local hobby stores: servos, metal and plastic arms for construction, wires, a base plate, different colored pipe cleaners, and an Arduino Uno with prototype shield. The hope in this approach is that any experimental results we might produce/report could be replicated elsewhere with minimal cost/effort (at least in construction of the robot prototype). Currently, many robot designs fail to meet such criteria, which makes it more difficult to build a sound body of scientific evidence for robotic design, human-robot interaction, and the like.

We also have created a completely 3D-printed version of the robot-face, which will also be available on our website in the coming months. This will allow anyone with a 3D printer to print out the exact same face (with some simple assembly), or modify the face to suit their purposes.

Additionally, while the focus was on maintaining a minimalist approach to design of a robotic face and eliminating any superfluous and/or conflating factors that could affect human perception thereof, another driving principle was basing the robotic face design on grounded empirical evidence. The source of such evidence was primarily recent research on computer vision facial expression identification on human faces (based on the six Ekman emotions), as well as psychological research on human facial expression identification and perception. Namely, the Facial Action Coding System (FACS) and related Action Units (AUs) that correspond to muscles/muscle movement in the human face were used to guide the construction as well as the programming philosophy, in that at the most basic level programmatic functions operating the robotic face roughly correspond with specific AUs. This approach provides a theoretical basis to the design grounded in existing empirical research.

The primary goal in construction of the robotic face is to answer basic science questions about how best to design a robot face for purposes of interacting with humans, particularly for using non-verbal communication (such as human-like facial expressions) for such interactions. This includes questions around human social cognition and cultural variation thereof.

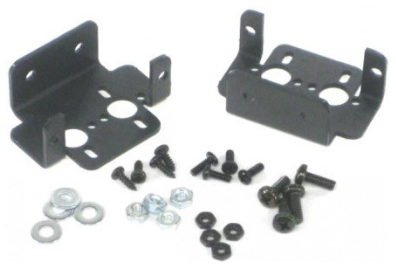
II. Step-By-Step Construction Manual

The following is a step-by-step instruction manual. There is also a fully 3D printed version that we will be releasing summer 2014. Construction time is about 4-6 hours, depending on experience. Cost (as of 2012) is about $150-200 with the neck included. We'll try to explain everything we do in detail, but referring to the many included pictures of construction may be most helpful.

1) Parts List:

Most of these parts are easily available online (e.g. www.robotshop.com, www.pololu.com, www.sparkfun.com, www.tamiyausa.com) and/or from your local hobby store. There are numerous additional pictures of most of these items in the following steps below as well.

- 1 Arduino Uno
- 1 Arduino Uno Prototyping Shield
- A few Breakaway Headers (for the prototyping shield)
- 10 sub-micro servos (we use Hitech HS-55)
- 4 micro multipurpose Servo Brackets (we use Lynxmotion models), look like the below



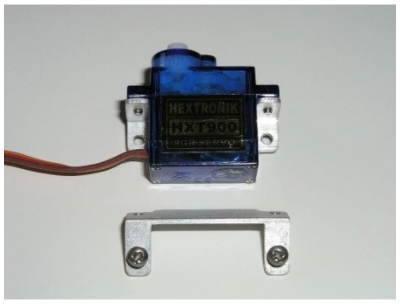
- 2 Tamiya Universal Plate L (210x160mm)
- 1 Tamiya Long Universal Arm set (the longer ones, not shorter)
- 28 Tamiya Universal Metal Joints (come in packs of 4)
- Some jumper wires, assorted colors
- Some spare 3mm screws/nuts, if needed
- 1 roll of Steel Gauge Wire (22-Gauge)

For the neck:

- 1 Heavy-Duty Pan&Tilt system, with ball bearings (we use SPT200 Direct Drive Pan & Tilt System)
- High-torque Standard size servos, for Pan&Tilt (we use Hitech HS-485HB)
- 1 Tamiya Universal Plate Square (100x100mm, ProductID=RDP-801) – we like to use the black ones for this (to differentiate from the rest of the face) but we can only seem to find those in Japan, color doesn't really matter though
- 28 Aluminum Standoff: 1/2" Length, 4-40 Thread, Male-Female (M-F, come in packs of 4)
- 8 Aluminum Standoff: 1/2" Length, 4-40 Thread, Female-Female (F-F, come in packs of 4)
- Some 4-40 Thread screws to secure the standoffs

Optional Stuff:

- Pipe cleaners, assorted colors, we originally used these as simple facial components, they worked well, although we are now using 3D printed facial components
- 2 low-profile servo brackets, aluminum, you can find them online by doing a Google image search for "sub-micro servo bracket", they look like the below. However, for one of the prototypes we just hand-made something similar by bending some metal arms. We use these to attach the eyes.

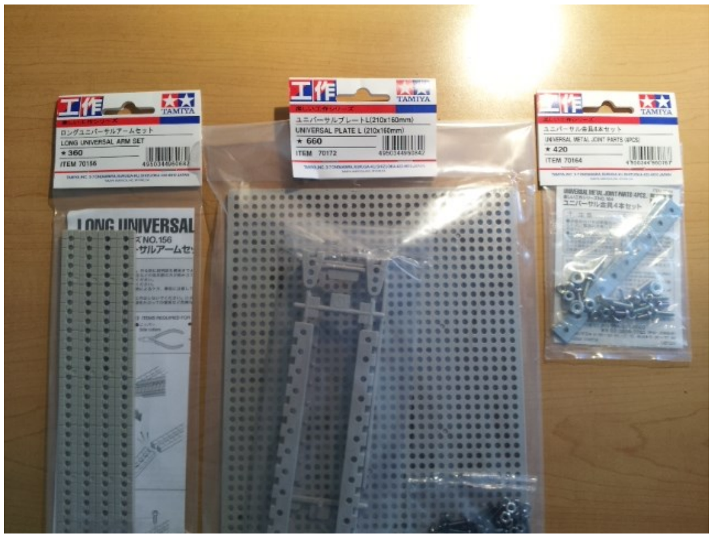


Tools:

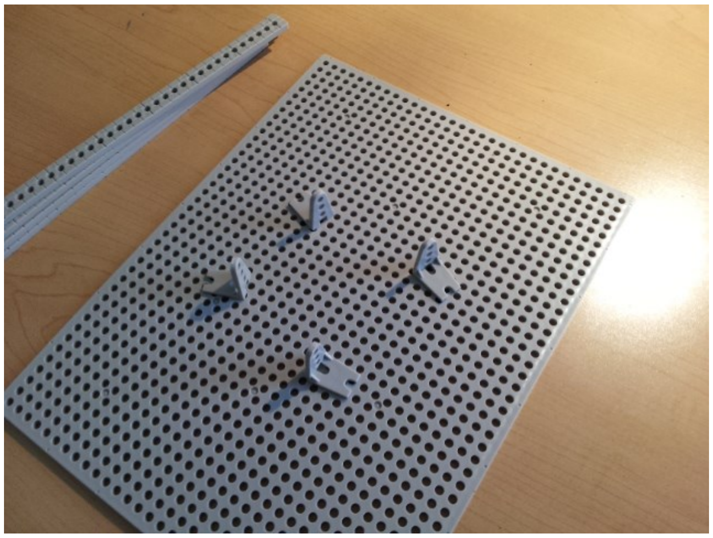
- A soldering gun (and solder, of course)
- Assorted pliers
- Precision screwdrivers
- Electrical wire, assorted colors ("stranded" is easier to bend)
- Electrical tape
- Wire strippers
- Wire cutters (for cutting wires and metal)
- Snips (for cutting plastic)

<u>2) Build the Basic Skeleton Support Structure of the Head</u>

In this stage, we are going to construct the basic structure that will support all the facial components, the arduino, etc. We start with the Tamiya Universal Plate L, Long Universal Arms, and Universal Metal Joints, as shown below.

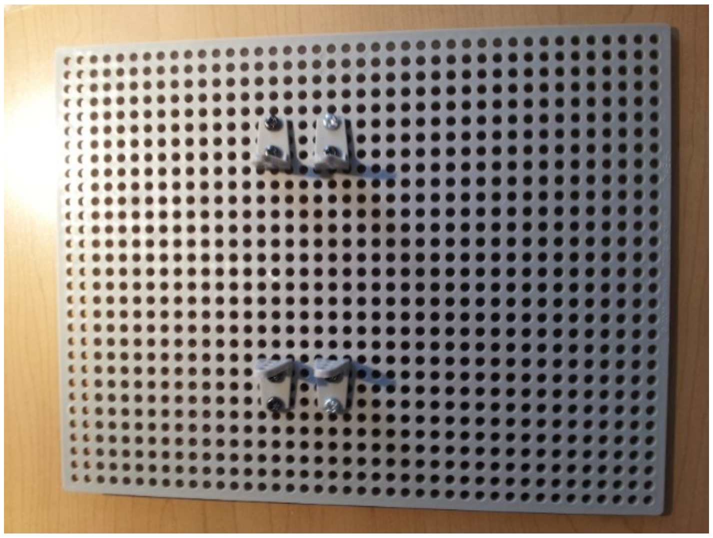

TAMIYA
LONG UNIVERSAL ARM SET
ITEM 70156
UNIVERSAL PLATE L (210x160mm)
ITEM 70172
UNIVERSAL METAL JOINT PARTS (4PCS)
ITEM 70164

The first we need to do is connect the four angled braces to the Universal Plate.

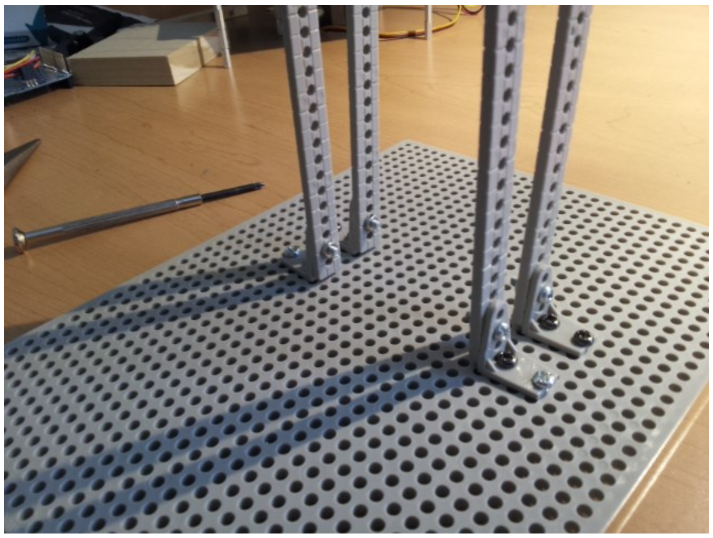


Using some of our 3mm screws/nuts (which come included with the Universal Plate).

The connected angled braces should be assorted like the below. Note the precise number of holes from each side (5 on one side, 6 on the other, as well as from each end). This is important for weight distribution, especially if you want to attach it to a neck mechanism. <u>It's also critical for other steps to line up correctly below.</u>

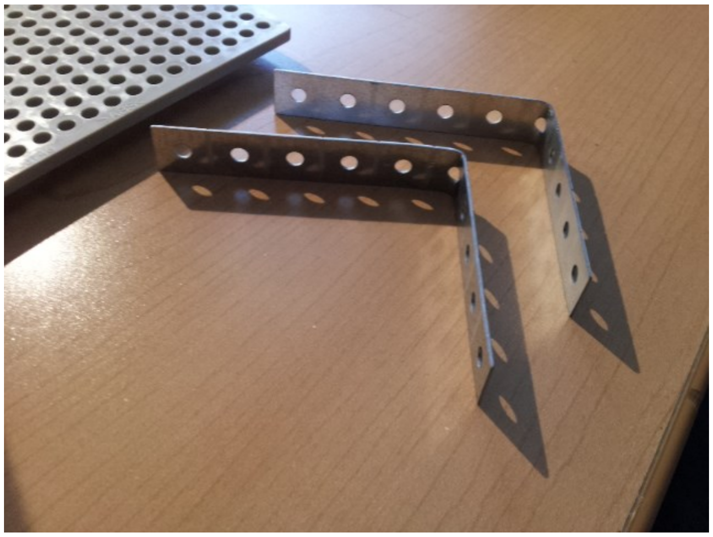


Now, we need to attach the four Long Universal Arms to the braces (one to each brace).

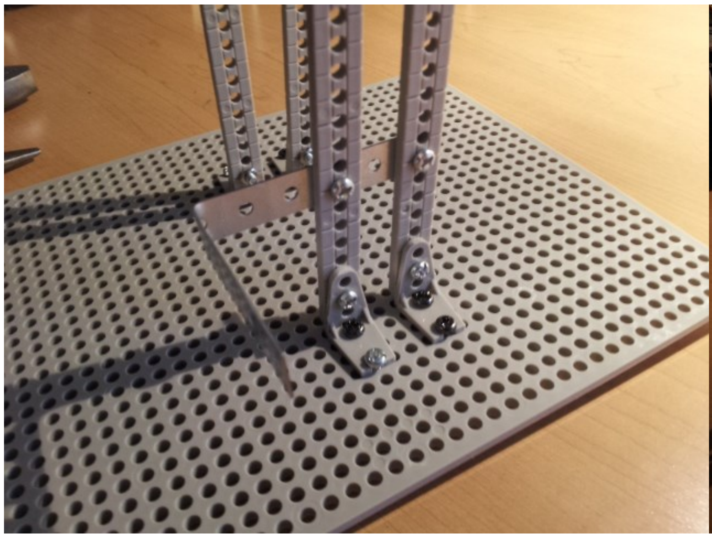
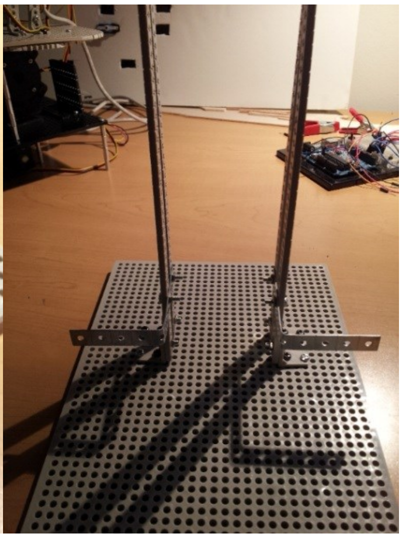


2) Add the Skeleton Support Structure of the facial components

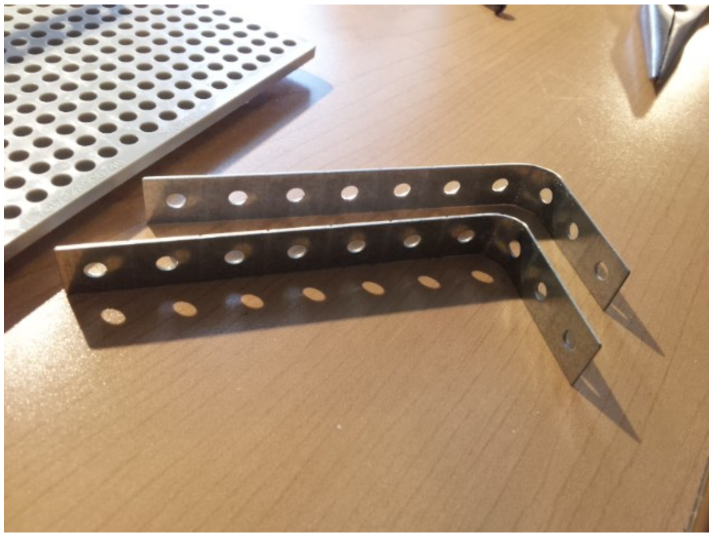
Next, we need to use the Universal Metal Joints to add the support structure for the various facial components. The first part is adding two metal supports for the mouth. Using two pairs of pliers, bend 2 of the Metal Joints as shown below (bending them with pliers is a bit of an art, but with a little practice, it's not too hard). Note that they are each bent on the 6th hole (out of 10). The flat side with 4 holes will serve as the "front".



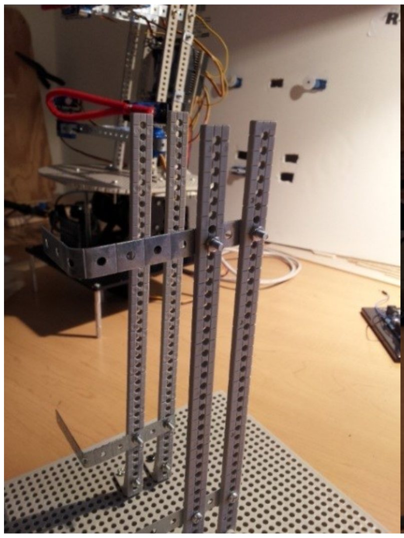
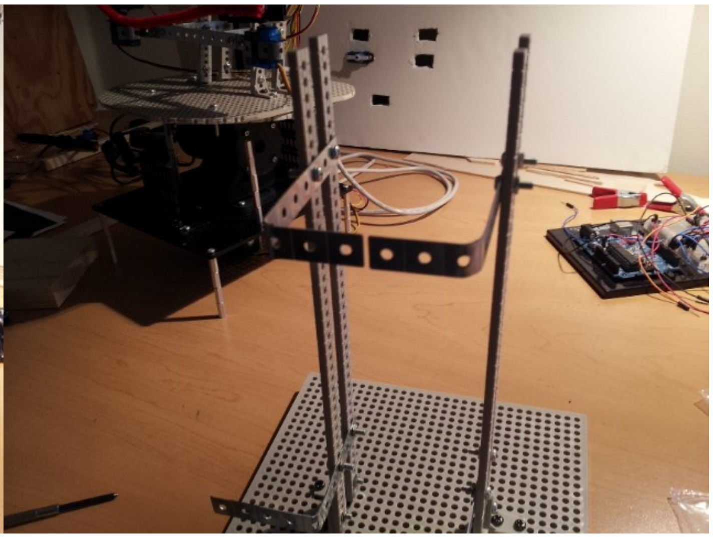
Now, attach the bent Metal Joints to the Long Universal Arms as shown, one on each side. Note that we are attaching each one 5 holes up the Arms from the angled brace. Also, note we attach the 5-hole bent side to the arm, leaving the 4-hole side facing the front.

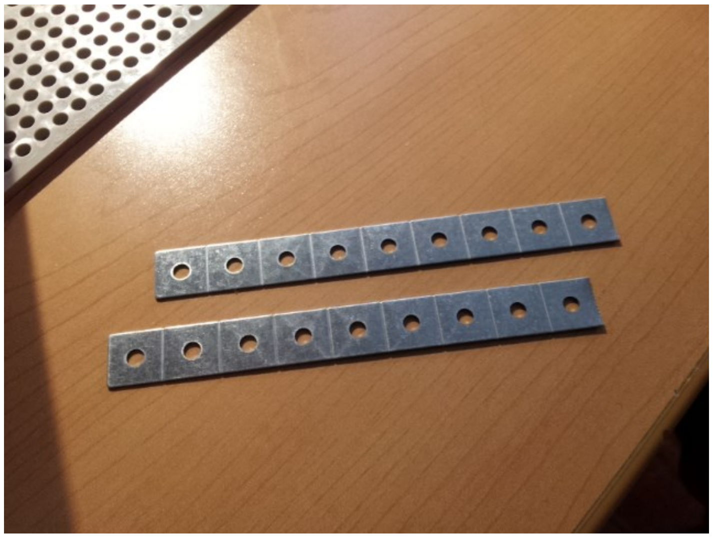
Now on to the supports for the eyebrows. We need to bend a couple more Metal Joints as shown below. Note that we are bending them this time *between* the 7th and 8th hole. The two-hole bent side will face to the front.



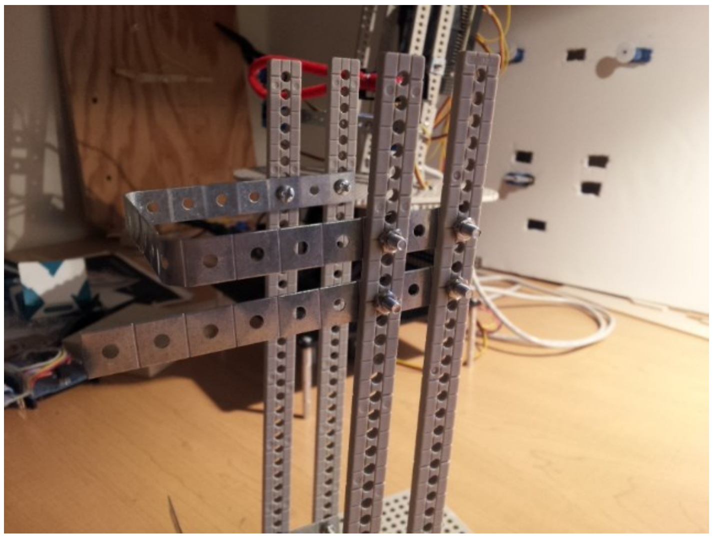
Next we attach them to the Long Universal Arms, one on each side. Note that they are attached 8 holes down from the top of each Arm.

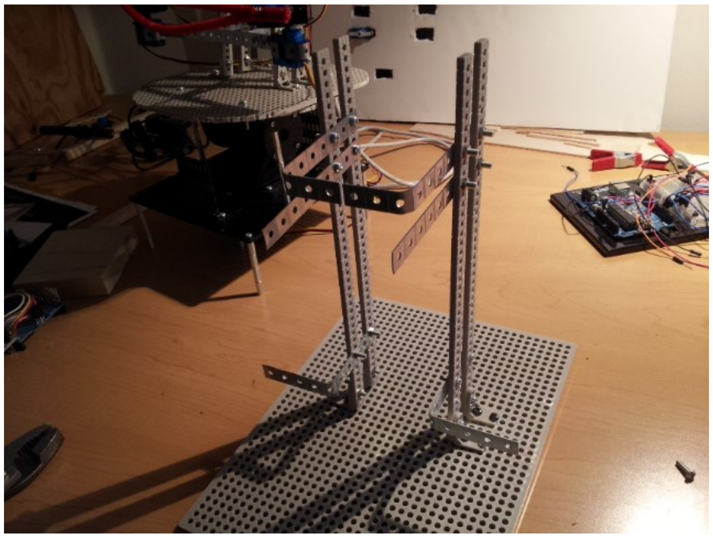



Next, we need to add the support structures for the eyes. For this one, we don't need to bend the Metal Joint, but we do need to cut it to make it a little shorter. You'll notice that the Metal Joint has indentations between each hole. Using some wire cutters, we want to cut off one of the holes, so that the resulting Metal Joint is 9-holes long. We need to do this to two Metal Joints (one for each eye), as shown below.



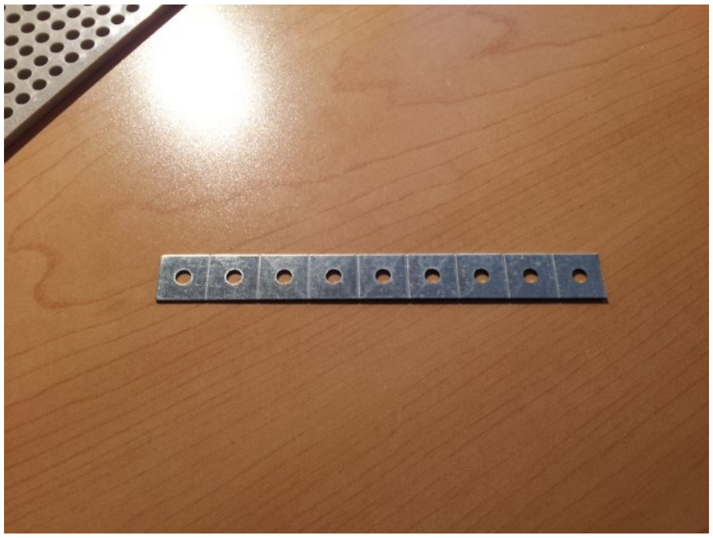
Now we need to attach these to the Long Universal Arms, one to each side. Note that we attach these 3 holes down each Arm from where we attached the eyebrow supports (i.e. 11 holes down from the top of each Arm).



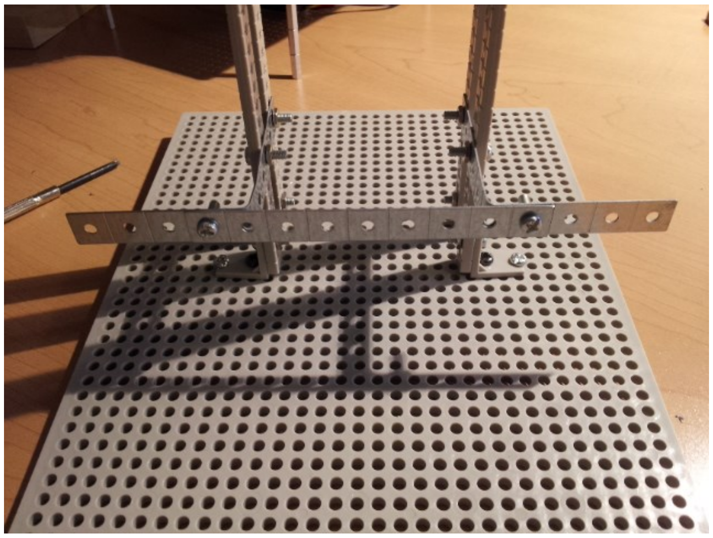


3) Additional Supports for the Mouth

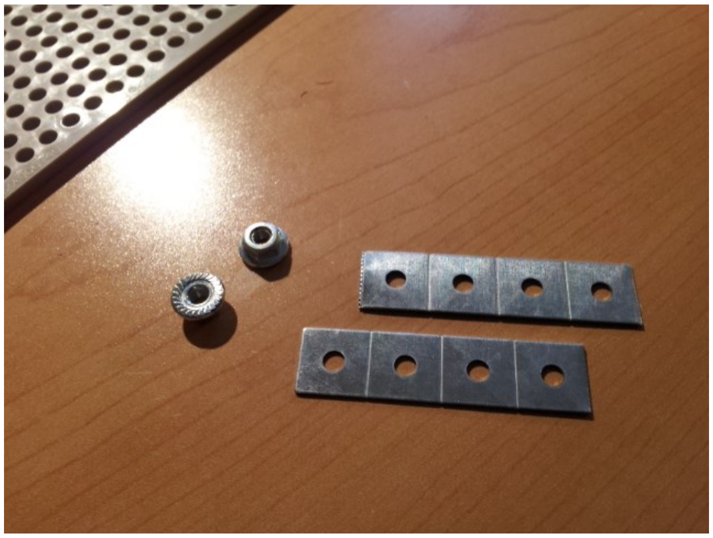
For the mouth, we need to add extensions to each side using Metal Joints, as well add a brace across the center to stabilize it for movement (otherwise servo motion at each mouth corner will cause the supports to move, rather than the mouth). First, let's add a brace support across the middle. To do this, we again use wire cutters to cut a Metal Joint down to 9-holes.



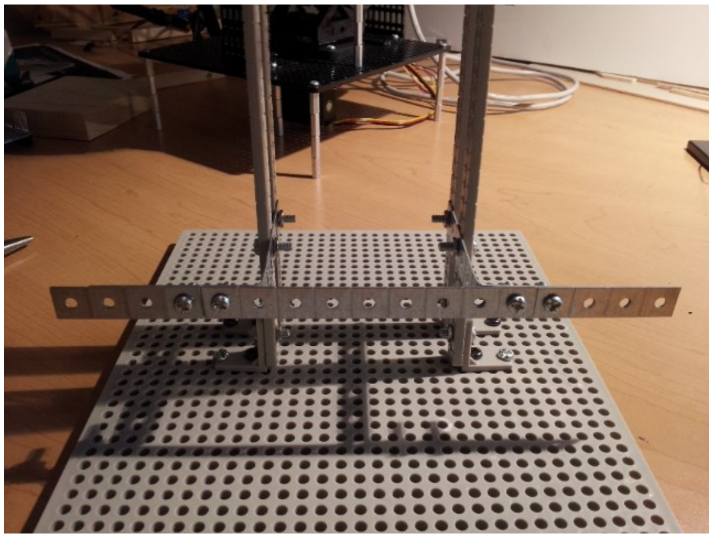
Then we attach it to both sides of the mouth supports (from step #2). Note that a 3mm screw at each end should line up exactly with the first hole of the front-facing part of the side supports.



Now we need to add 4-hole extensions to each side, so we cut two Metal Joints to size.



And finally we attach one of these to each side of the mouth supports. Screw these in one hole to the outside of where you attached the brace.

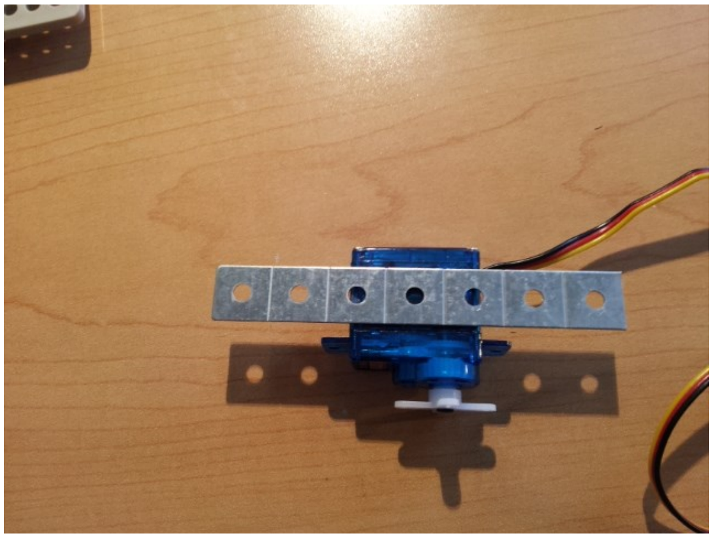


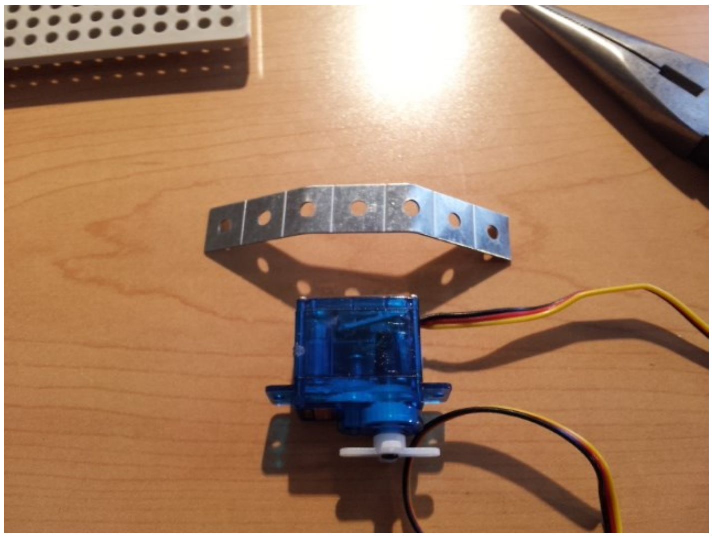
4) Make Some Servo Brackets

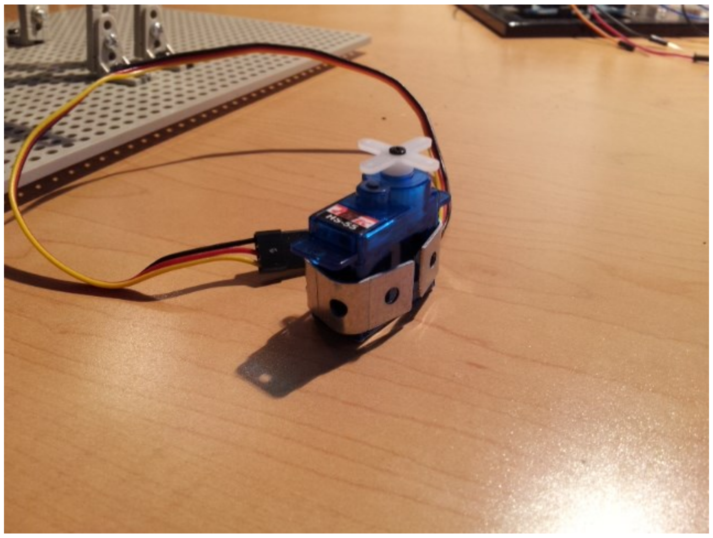
Generally, we use bought servo brackets. But for a few places in the face, it's just as easy to make some out of the Metal Joints. If you prefer to purchase some, you can of course do that and simply attach them in place of the ones we make below. Aversely, if you had trouble finding the low-profile servo brackets for the eyes (see the Parts List), you can make a couple extra ones exactly like we do here and use those instead to attach the eye servos (step #5).

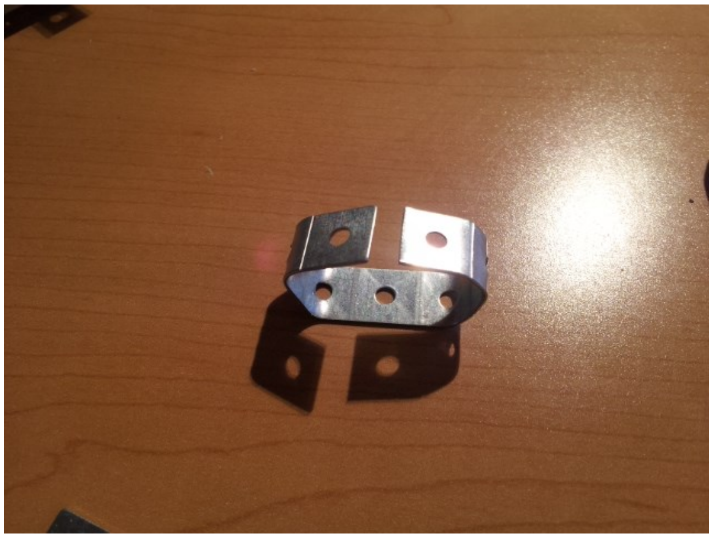
First, we need to cut some Metal Joints down to 7-holes, using wire cutters. We need four of these (two for the mouth, two for the eyebrows).

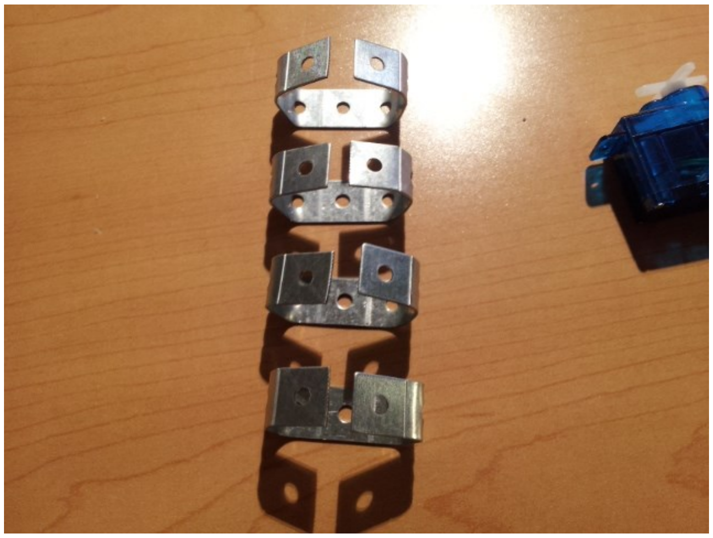
We are going to “wrap” these around each of the sub-micro servos, so they make snug-fitting holders for them. To do so, we line up the middle hole (the 4th from either end, out of 7) on one of the wider sides of a servo, and gently bend it around the servo. Don’t use pliers for this, it can be done with bare hands. Just like bending Metal Joints with pliers in step #2 though, this is a little bit of an art. It’s best done iteratively, i.e. bend a little on one side, then bend a little on the other, then bend the first a little more, etc. Once you get the hang of it though, it’s actually fairly easy to make consistently shaped brackets this way, as shown by the series of images below.



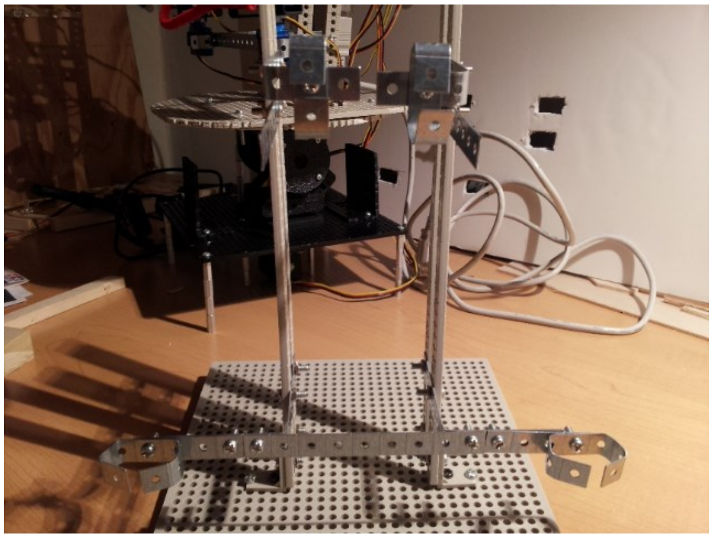








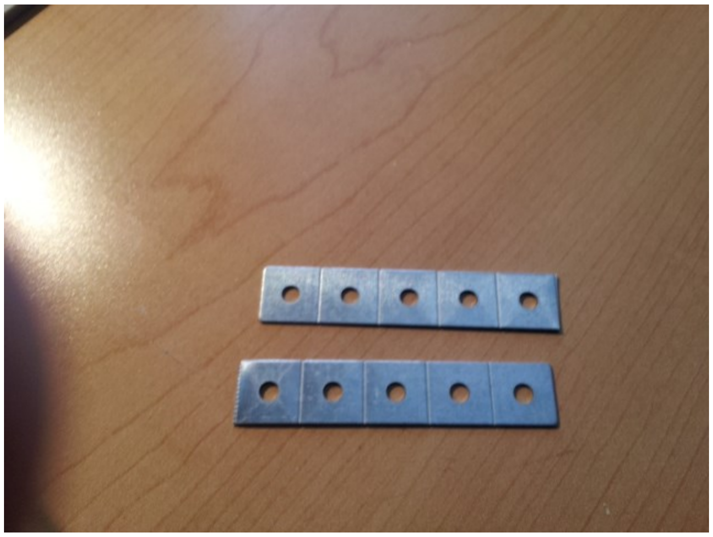
Now we need to attach each of these servo brackets to the face. We'll attach one to each eyebrow support (the middle hole on the front-facing side), and one to each end of the mouth (the very outside hole). This can be seen below. Note that with the screw in place, you will probably need to "loosen" up the bracket a little so the servo can slide in – however, *do not* put the servos in yet, we need to do a couple other things first.



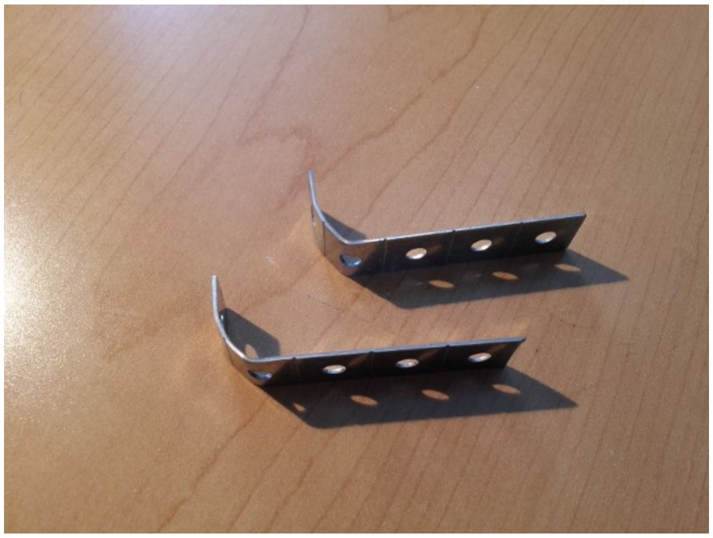
5) Attach the eye servos

The eyes need to be oriented forward, so first we need to add an extension of the eye supports to allow us to attach the servo bracket so that the servos can be oriented facing forward. If you bought and/or are using the low-profile servo brackets from the Parts List, you can follow these directions exactly. Otherwise, you can make some extra servo brackets out of Metal Joints as described in step #4, you'll just need to alter the orientation of the extension (instead of supporting from underneath, they will support from behind).

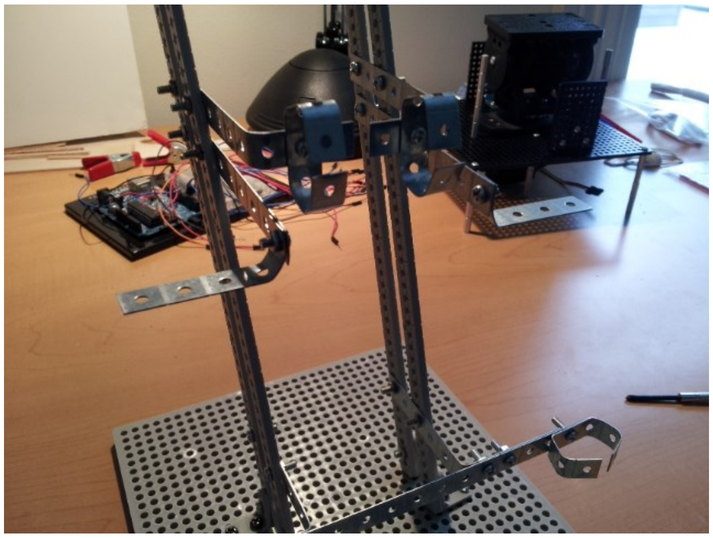
First, we need to make the extensions. We need to cut a couple metal joints to 5-hole length, using wire cutters.



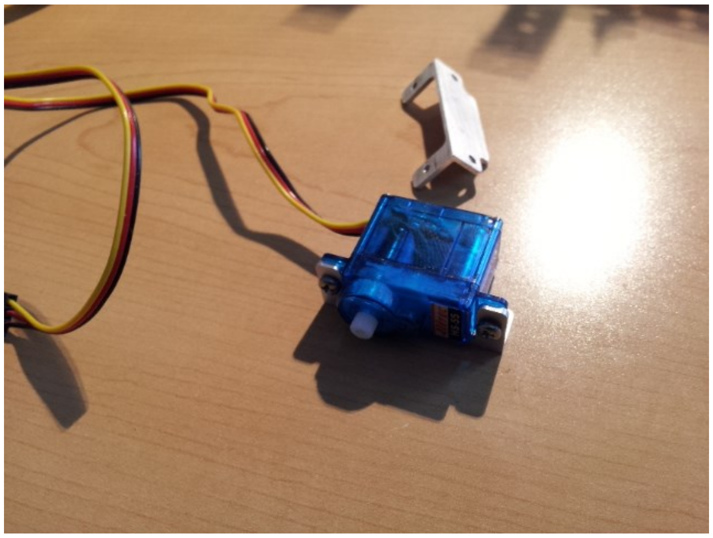
Then we need to bend them between the 4th and 5th hole. Make sure the 5th hole is flat, so it can be secured flush against the eye support.



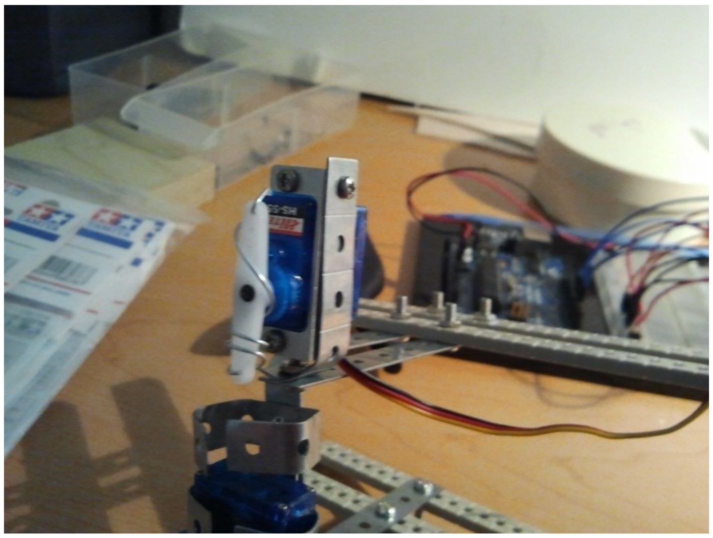
Now, we need to attach the extensions to the eye supports, one to each side.



Now, we mount one servo for each eye to the low-profile servo bracket.



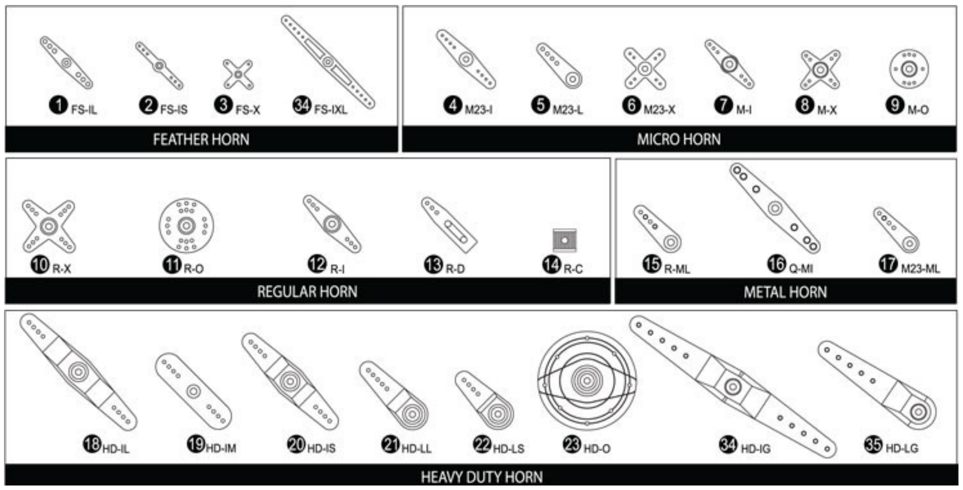
And then we attach the bracket to the extension (the extension supports it from underneath). Note the image below already has the servo horn (plus gauge wire) attached – we will discuss this in a later step.



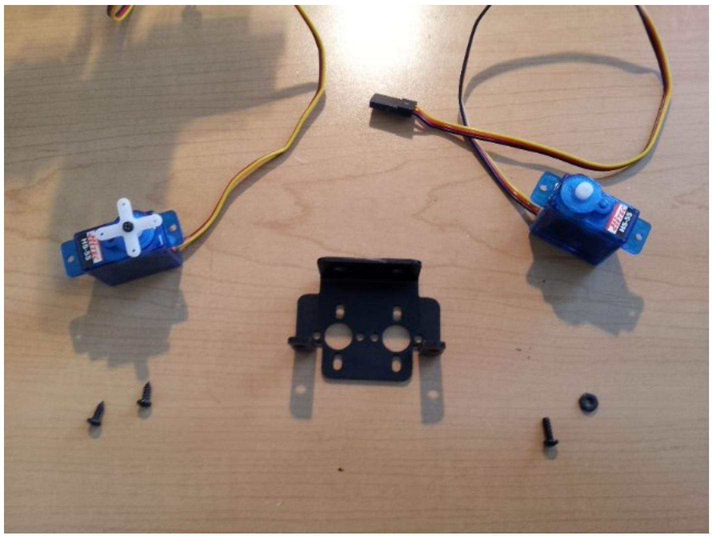
6) Attach the eyebrow/mouth servos

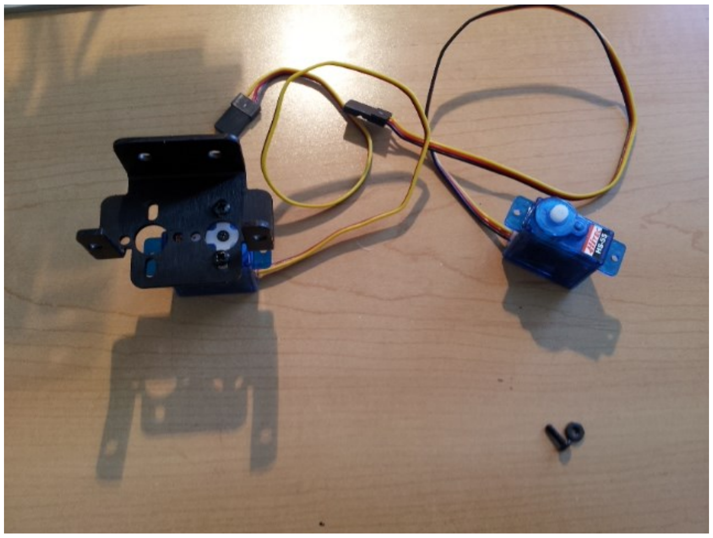
The eyebrows and mouth servos are attached exactly the same way. The only difference is how they are oriented. Both of them have 2 degrees of freedom, involving one servo mounted on another, via a multipurpose servo bracket. We'll refer to the first servo (the one which we mount the bracket and 2nd servo on to) as the *base servo*. We'll refer to the 2nd servo as the *mounting servo*.

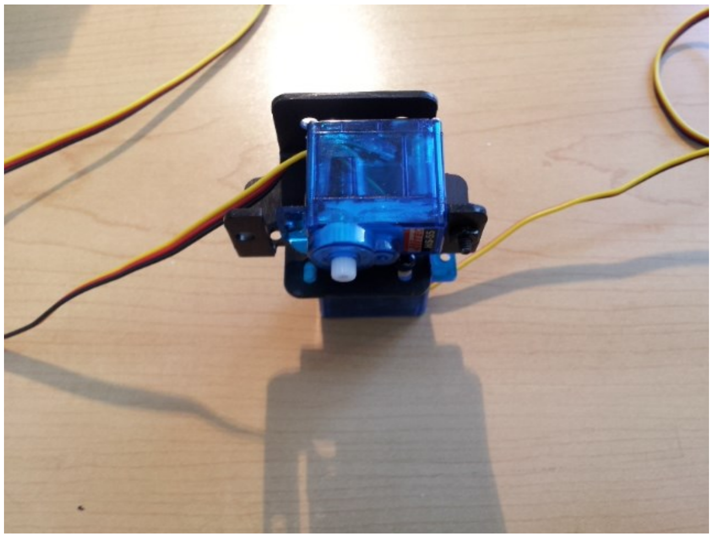
First, we need to attach the servo bracket to the base servo (or rather one of the servo horns attached to it). An image of Hitec servo horns is shown below. We typically use #1 horns for all the servos (especially for the facial components). For affixing a servo bracket to a base servo though, we often use a #3 horn. Other shape horns might work as well. Also note that the programming code is setup so that the horn on every servo should be parallel with the servo body when at 90° degrees (see Section III.b).



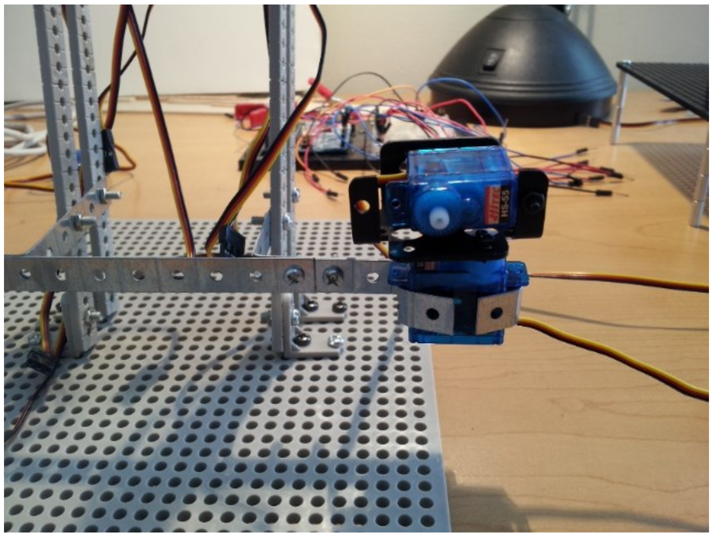
So, we attach one of the multipurpose servo brackets to the base servo (with a horn) and the mounting servo (without a horn) as shown in the steps below.



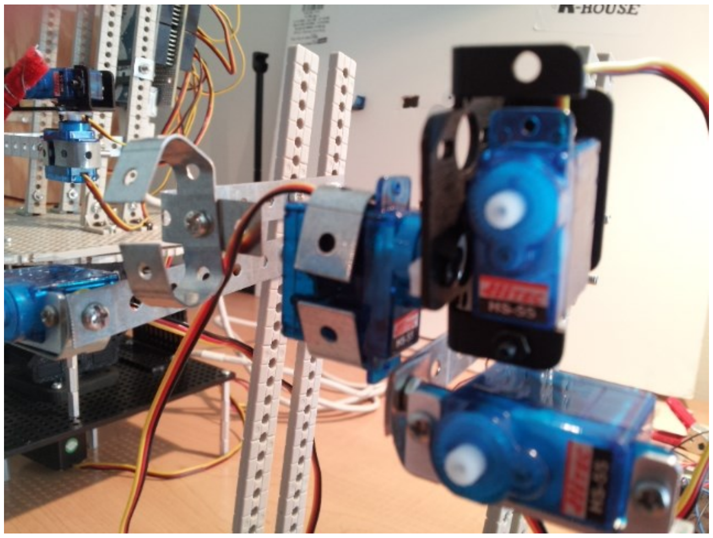




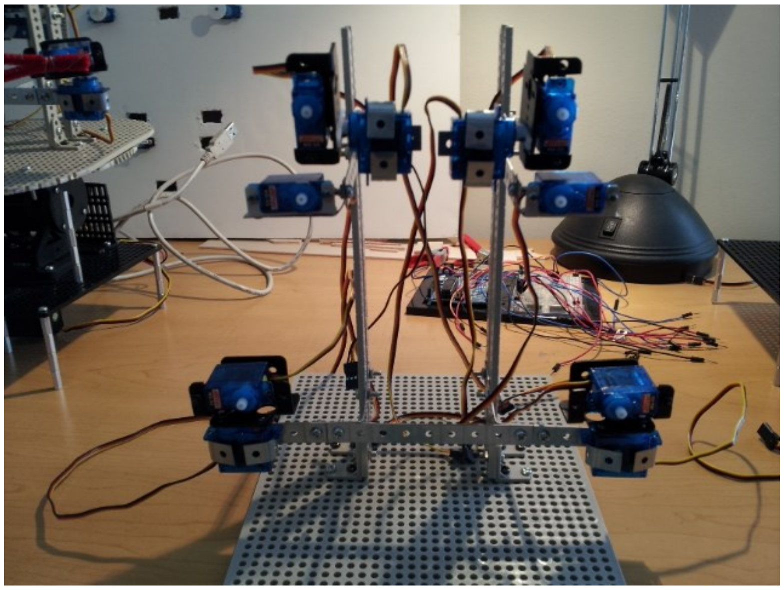
We need 4 of these – one for each mouth corner and one for each eyebrow. After we make them we can slide the base servo into the servo brackets we made in step #4. The mouth corners will slide in from above, pointing up (as shown below).



The eyebrows will slide in from the side, pointing out.

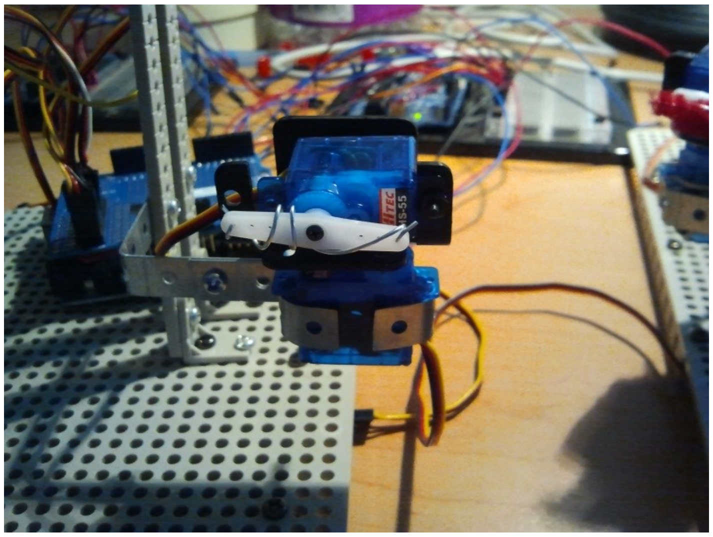


Once you get everything in place, it should look like the below.

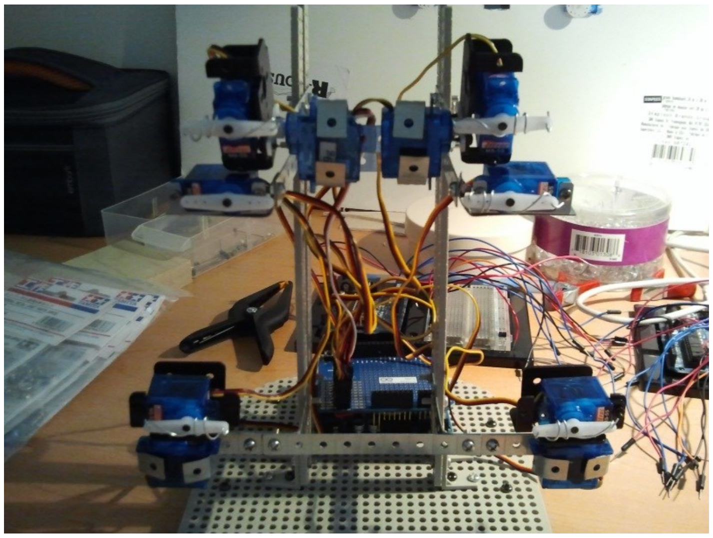


7) Attach the servo horns/wire

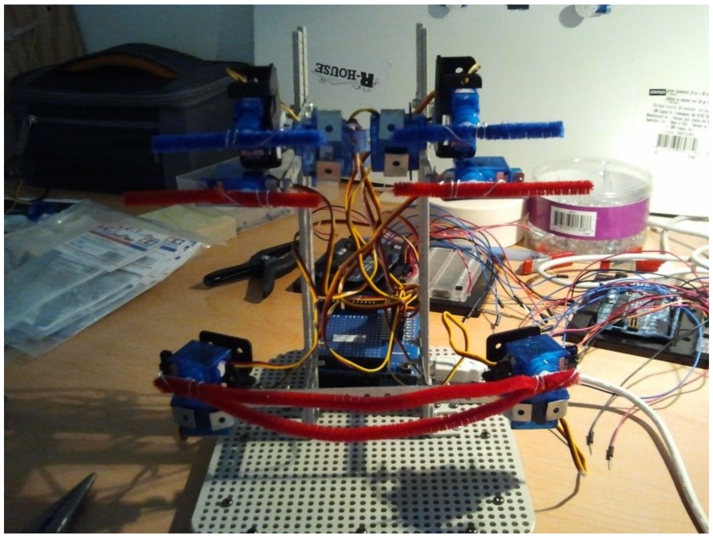
At this point, we can attach the servo horns to all the facial feature servos (eyes, eyebrows, mouth corners). If you just want something simple, you can attach steel gauge wire (as shown below) through the holes in the horn, making loops where you slide colored pipe cleaners into. If you are 3D printing facial components, or using other materials, you may have to modify this (e.g. use glue rather than gauge wire).



If you do use gauge wire, you'll end up with something that looks like the below.

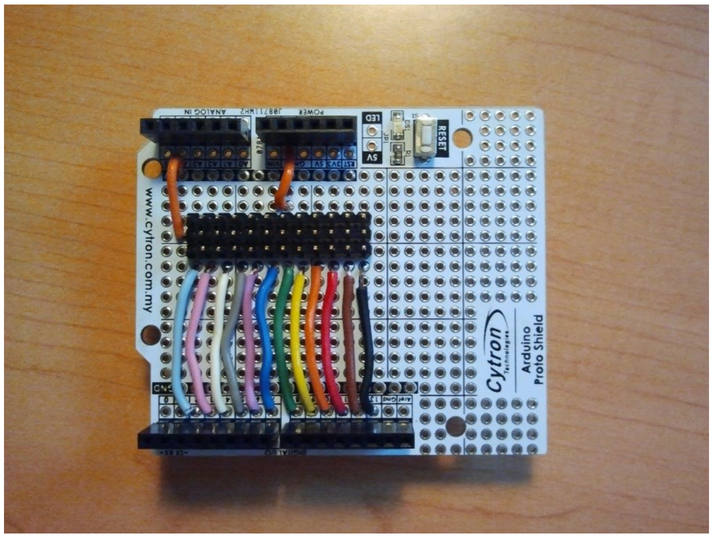


If you further attach some pipe cleaners, you'll get something that looks like the next image. The pipe cleaners are cut to: mouth (~6.9 inches), eyes (2.75 inches), eyebrows (2.5 inches).

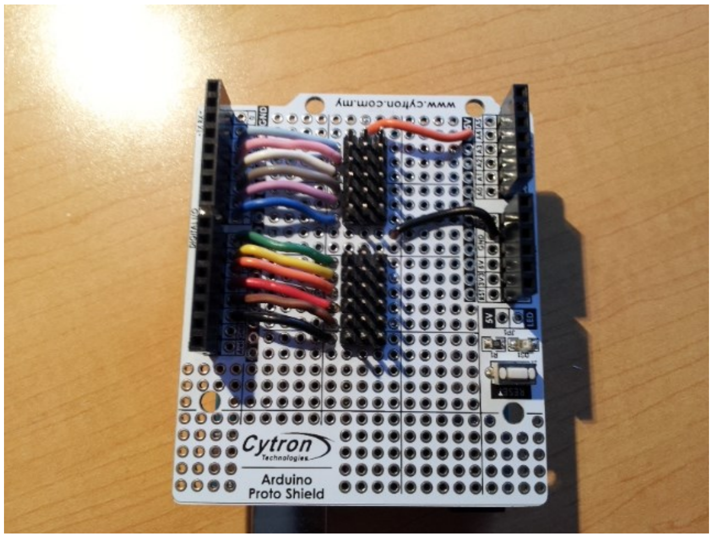


8) Add Arduino board

Next, we need to add an Arduino microcontroller board to function as "the brain" of the robot face. We already have an Arduino board, but we need to solder the prototyping board so that we can just plug the servos directly into it (i.e. so we don't need a breadboard). Really, the prototyping board can be soldered in any way you want, so long as there are 12 connections for the servos. The way we prefer to do it is shown below.

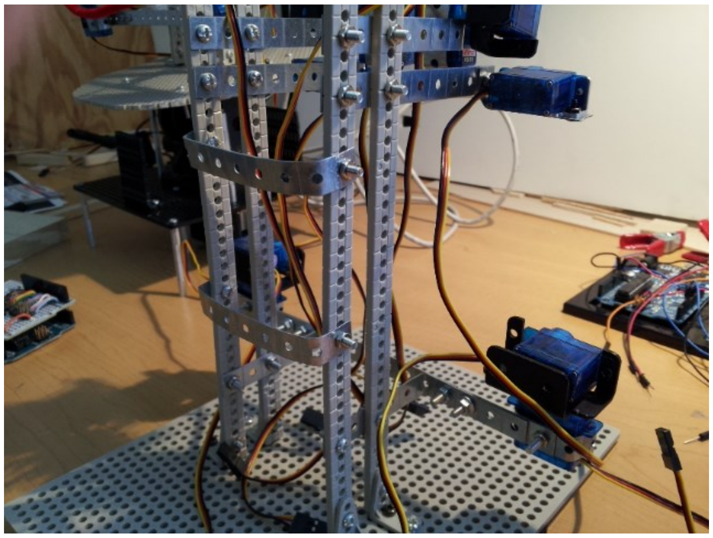


Basically, you have a series of 3-prong connectors across the middle of the prototyping board, with one pin being the positive, one the ground, and one the control pin. One thing that makes it easier to connect all the servos, is to leave a gap between the 3-prong connectors, so that instead of one long series of 12, you have two sets of 6.

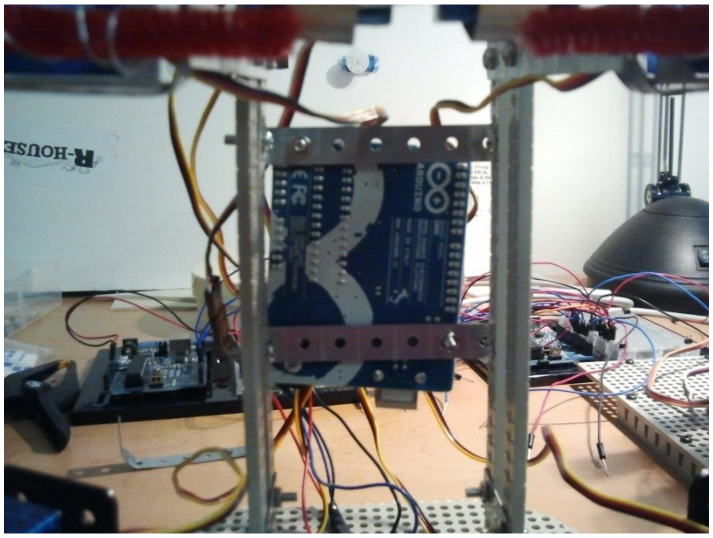


Either way will work though. This prototyping shield will attach on top of the Arduino. Next, we need a place to mount the Arduino on the robot face. We will do this on the back-side of the face/head (sort of similar to where the brain resides in humans). For this, we need to bend a couple more Universal Metal Joints. We bend them on both ends, on the 2nd hole from the end (i.e. the 2nd and 9th holes).

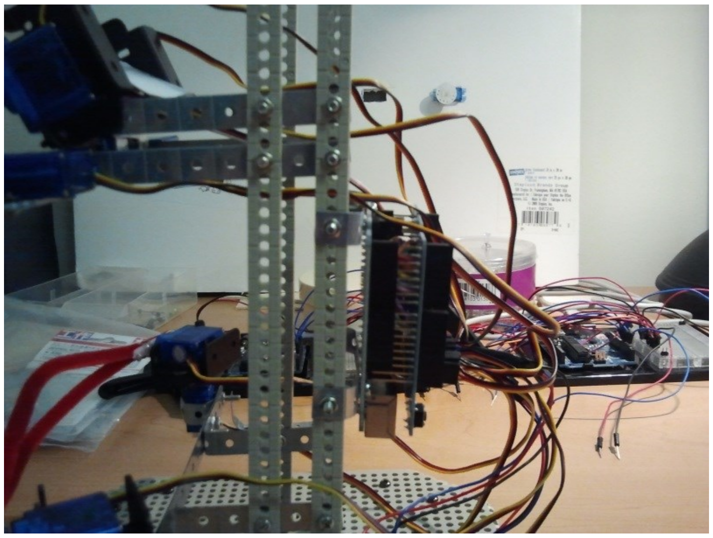
We then attach these to the back side of the face/head. Note, that the top Arduino support goes 4 holes down the arm from where the eye supports attached, and the bottom Arduino support goes 8 holes up the arm from where the mouth supports attached. This spacing is important for the holes on the Arduino to line up correctly so that you can attach it with screws.



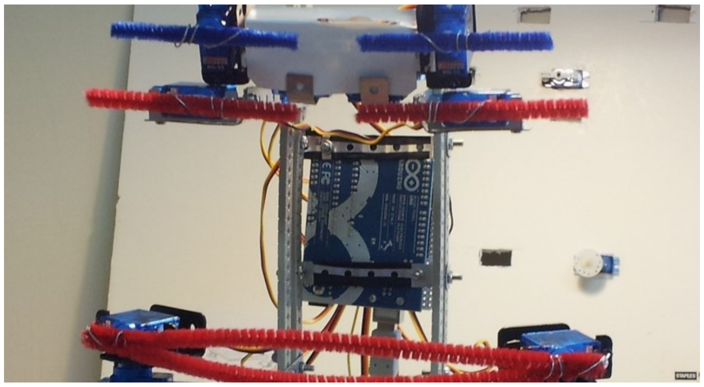
The Arduino board then can be mounted to the supports as shown below.



And the prototyping board can be mounted on it, and the servos plugged in. The usable pins are numbers 2-13 on an Arduino Uno (we don't use pins 0 and 1 because those are for serial tx/rx), which equate to the twelve 3-prong connectors we made on the prototyping board earlier. You can plug the servos in any of the connectors, or use the configuration we use (pin numbers can be seen in the example Arduino sketch distributed with the library).

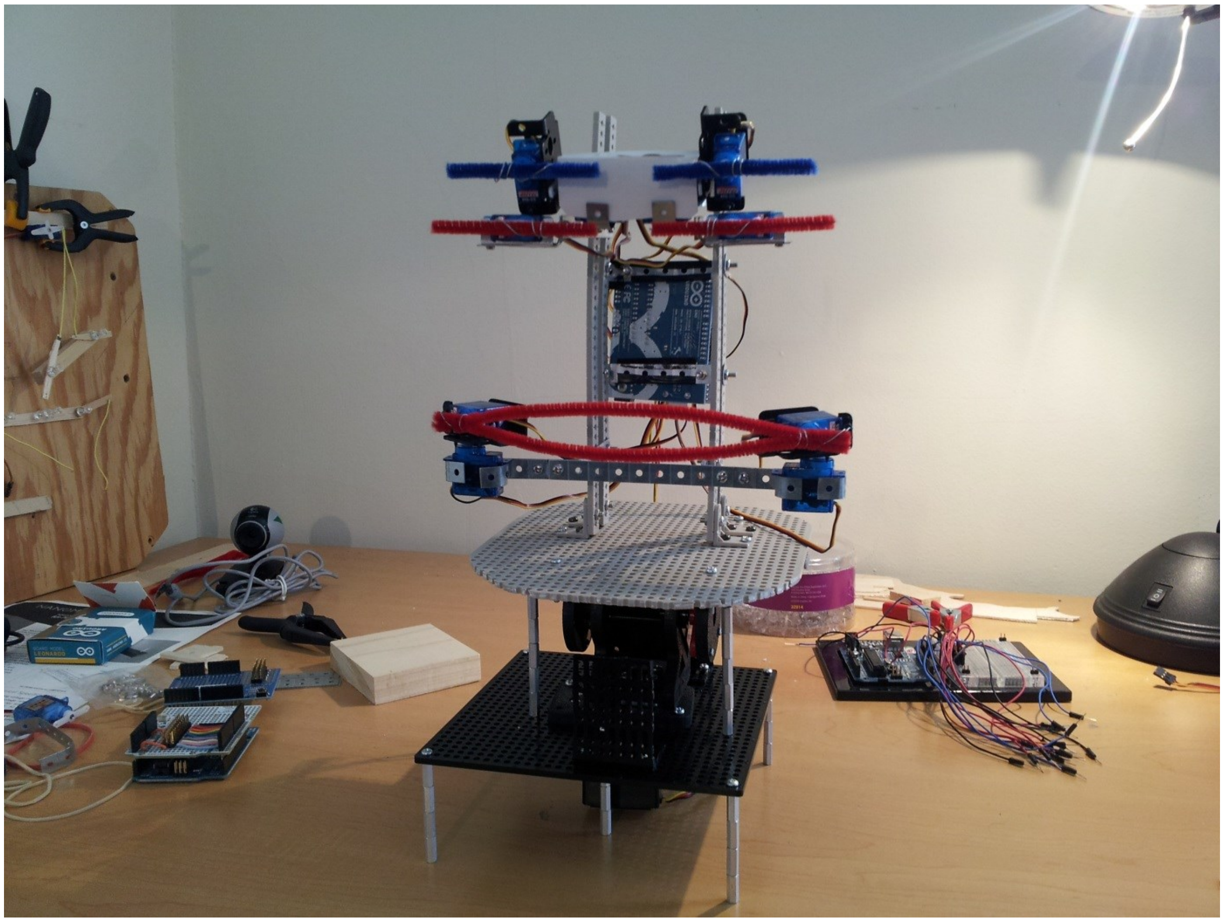


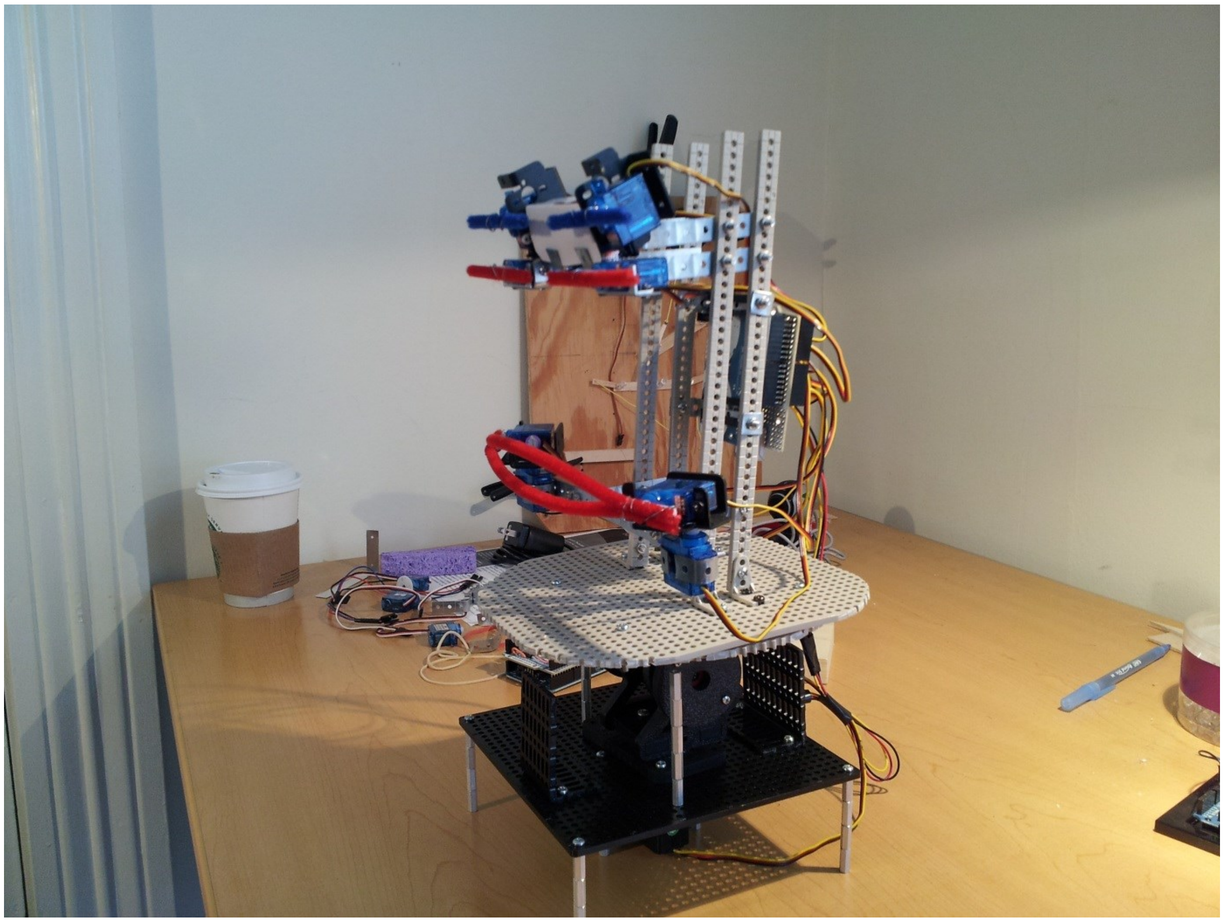
**Note: one critical thing we are not showing here (so as not to obscure how we attach the Arduino), is that the Arduino supports need to be wrapped in electrical tape. Otherwise, some of the metal on the Arduino may touch the metal supports and short out parts of the board. If you are randomly having servos not move correctly, or the Arduino fail to power on when connected to a power source, this is likely why. You can see from the image below (from a later stage of construction) the black electrical tape on the Arduino supports.



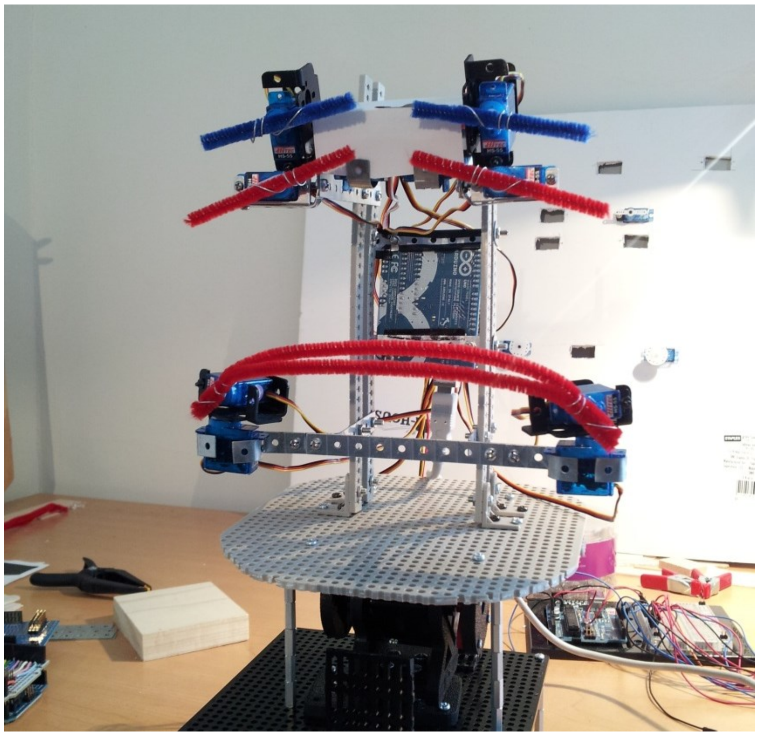
9) Adding Neck Motion (optional)

At this point, we are essentially done building the basic face. It can of course be enhanced in various ways as desired. We should have something that looks like the following series of images (not including the neck mechanism, i.e. the black sections). Note that we typically trim (using snips) the gray Universal plate that forms the base of the robot-face at this point, so that it is rounded and less boxy looking. That is aesthetic choice though.





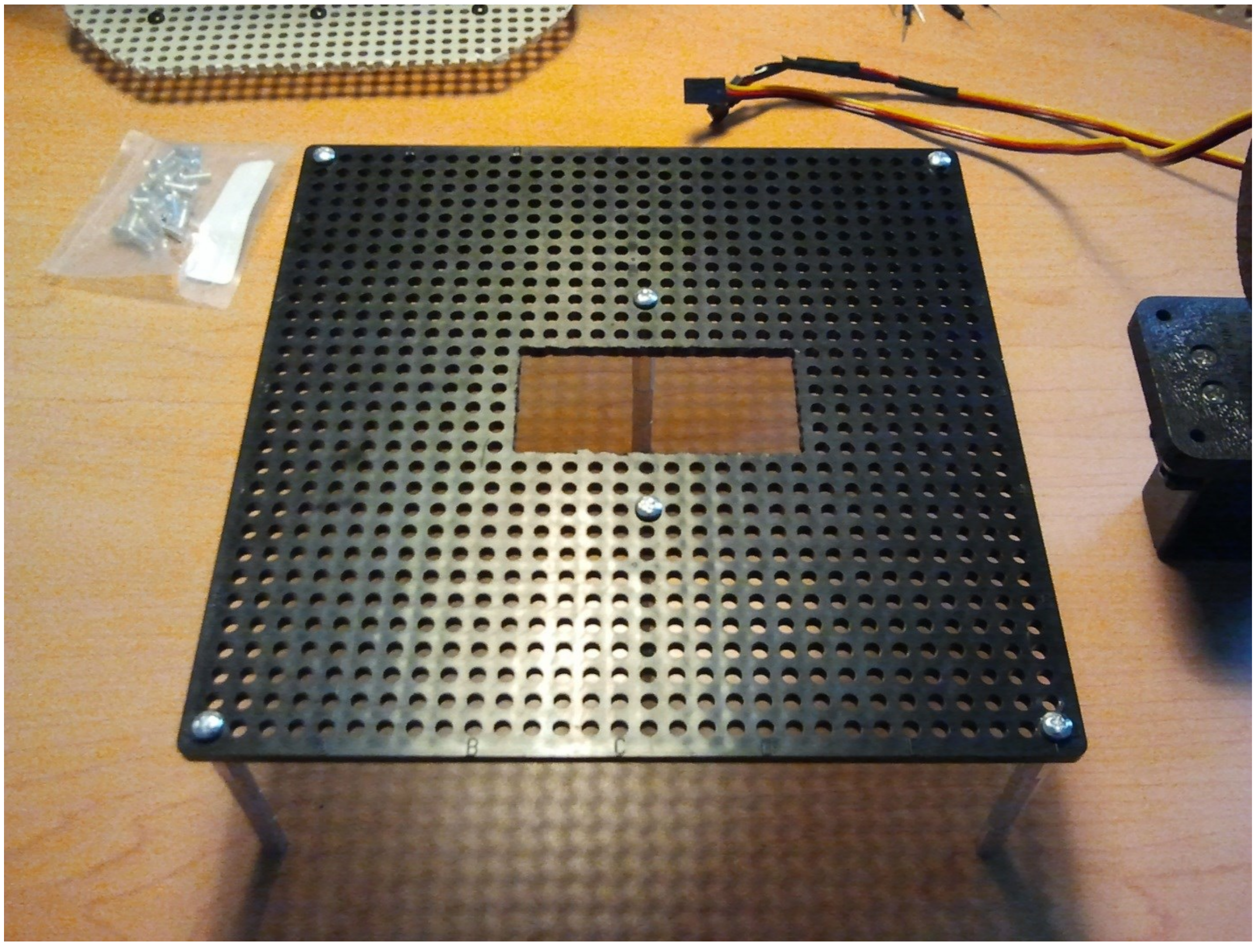
Or making a sad face:



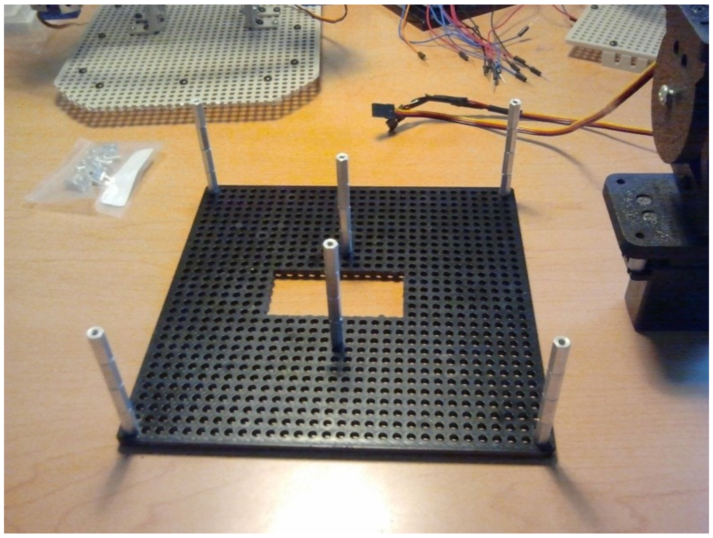
So the final thing we want to do here is add a neck mechanism so that the robot face can pan and tilt its neck, which can make its facial expressions more recognizable, as well as allow it to follow/track people or objects as they move around in its environment (assuming a camera is attached, see step #10 below). This step is optional.

We do this using the Pan&Tilt system, two heavy-duty servos, a Tamiya Universal Plate, and the aluminum standoffs (see Parts List).

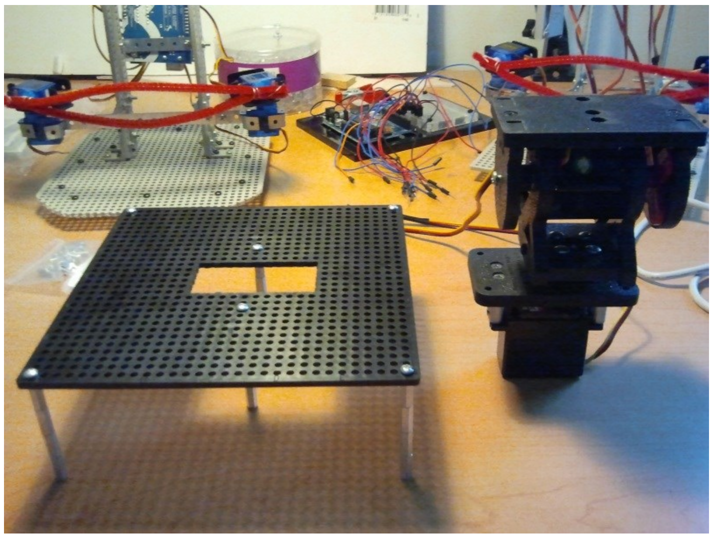
The first thing we need to do is cut a hole (where we can insert the Pan&Tilt system) in one of the square Tamiya plates. Again we like to use the black ones so they stand out from the rest of the face, but it's not necessary. Do note the number of holes from each side where we make the hole. The hole itself is 12 holes x 6 holes. An easy/simple way to cut this out is by using a pair of snips, making multiple cuts to the connectors between the holes on both the front and back side of the plate.



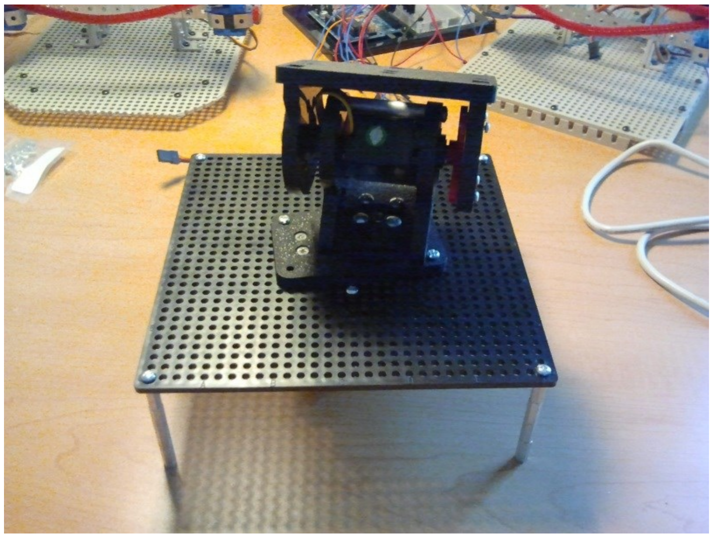
Next we attach the "legs" to the plate, so that it is high enough off the ground for the Pan&Tilt system to fit (you can see where these are attached above). We make these by connecting 3 M-F standoffs to each other, and then capping it off with 1 F-F standoff. Then we screw these into the plate.

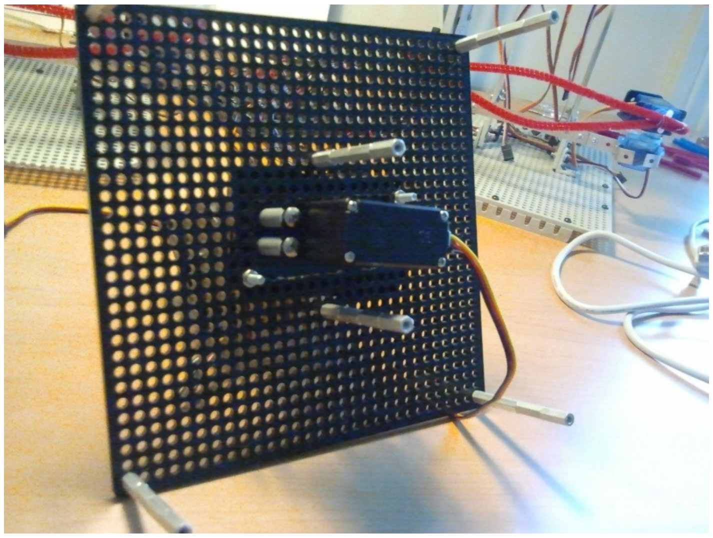


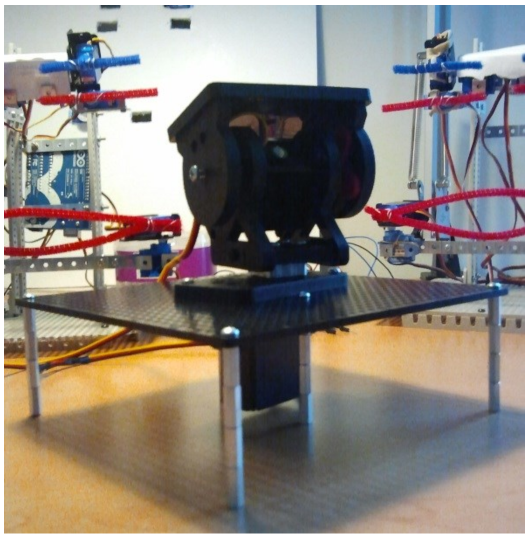
The Pan&Tilt system can be built by the following the directions that come with it. The heavy-duty servos will be integrated into it. You could also custom-build one, of course, but it's easier/more convenient to buy one. Plus, purchased ones (like the SPT200 one we use, see Parts List) includes ball bearings, which help with weight distribution/balance and, thus, performance. The constructed plate and Pan&Tilt system should look something like the below (different pan & tilt systems may requires slight modifications).



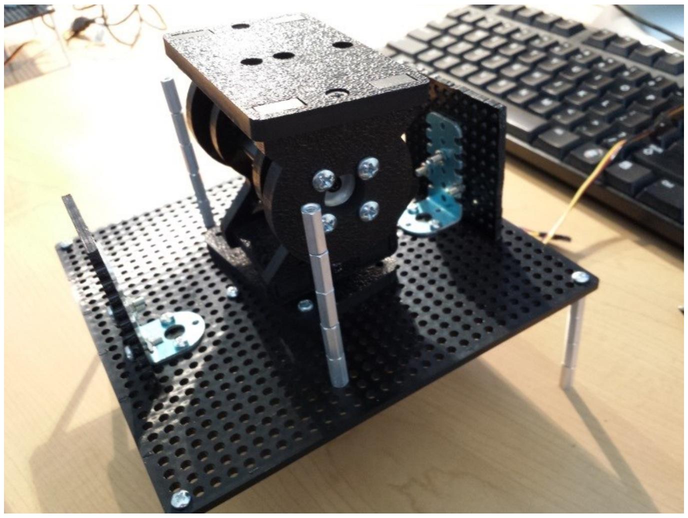
We then attach the Pan&Tilt system to the plate, using a couple screw/nuts.

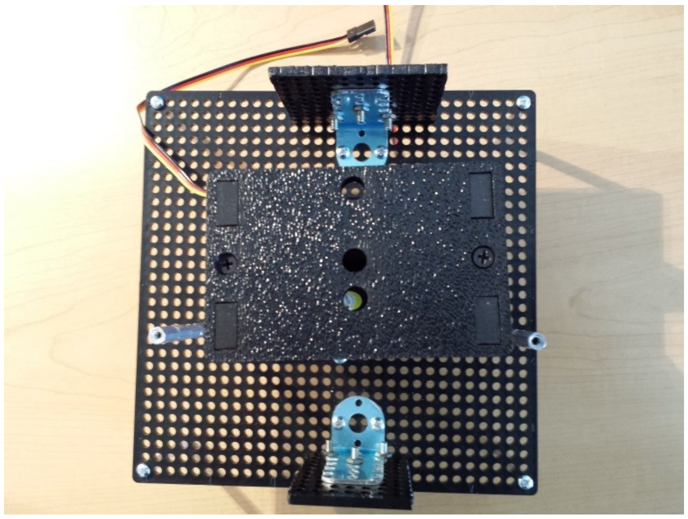




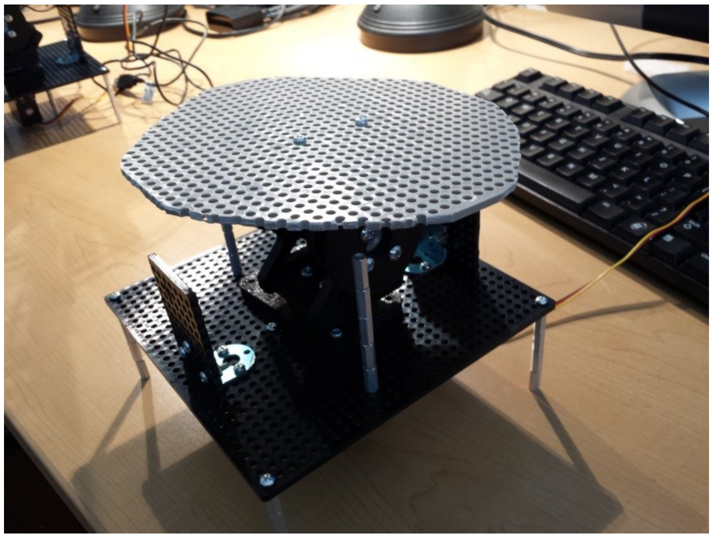


We usually add some “guards” to the plate, to constrain the movement of the face, and to keep it from tipping over. This is similar to the human neck, in that it only has a certain range of motion (e.g. you can’t spin your head around 360° degrees, you can only tilt your head so far up or down).





The robot-face can then attach to the top of the Pan&Tilt system. We typically don't attach the robot-face directly to it though, rather we create a *secondary plate*. This secondary plate is first trimmed down to a similar shape as the gray base plate of the robot-face, and then attached using a couple screws to the top of the Pan&Tilt system, as shown below. The base plate of the robot-face then "sits" on the secondary plate, attached by four screws placed towards each rounded corner (you can see this if you look closely at the 3 images at the beginning of this step). This makes it significantly easier to attach/remove the robot-face to the neck mechanism, and also helps with balance.



We have also created a 3D-printed "base" upon which the neck mechanism can attach, schematics for which we will be releasing in the near future. This "base" stabilizes the robot-face during movement (otherwise it tends to slide around a little bit, since it is top-heavy), and could also serve as the basis for the creation of shoulders or an upper torso, if desired.

<u>10) Adding Camera (optional)</u>

If you want to do any sort of vision stuff, e.g. have the robot face track/follow people or objects in its environment, you'll need a camera. It doesn't need to have a fancy camera, we've found that even a relatively cheap $20 USB web camera works fine. The RobotFace library includes functions for sending the location of objects of interest to the Arduino/robot, via serial communication (e.g. using a USB cable), so that the computer vision (CV) parts can be handled offboard (e.g. OpenCV) and the results simply sent to the Arduino. The RobotFace library also includes functions for calculating "where to look" using those detected locations, and for actual movement, as well random saccade movements when nothing of interest is detected. It works with either a fixed camera placed nearby the robot-face, or a camera mounted to the head itself (accounting for self-motion).

We won’t say much more about this part at the moment, but we will probably release more information about integrating CV with the robot-face (MiRAE) in the near future. We have a couple ongoing research projects related to this, including placing interactive robotic faces into public art installations, i.e. “robots in the wild”.

III. Programming Approach

We will provide a basic overview of the programming code and philosophy in this section. Note that more extensive details about individual pieces of code can typically be found in readme's associated with that code (e.g. the readme in the RobotFace library).

III.a Philosophy

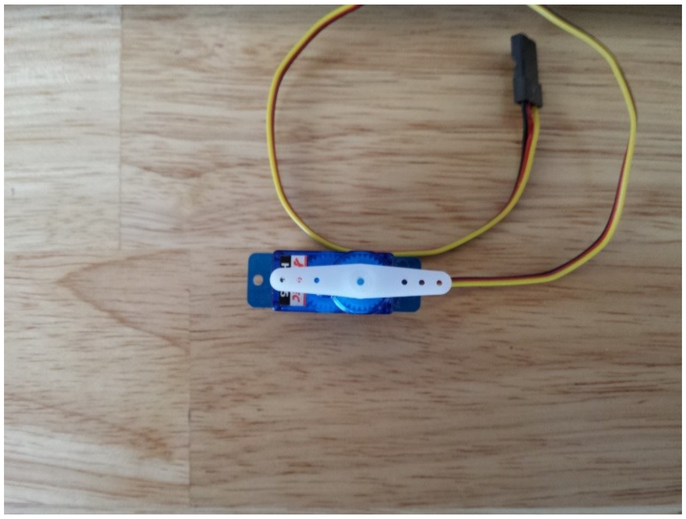
So there are few key principles that underlie the way we approached creating programming code for the robot-face (MiRAE). First, we wanted to develop it using an encapsulated function approach, so that everything was essentially self-contained functions to perform critical tasks. The goal was to be able to have the main program be nothing more than a collection of function calls and input parameters. This can be seen by the example RobotFace Arduino sketch included in the library. We initialize a RobotFace instance, set some parameters (e.g. the init values for the servos), and then call some functions. More complicated programs (such as those incorporating computer vision/visual tracking of people) work exactly the same way.

Another key principle behind our approach was that the programming code should be flexible and extensible. For example, expressions of different intensity can be made just by changing the degree values in a RobotFace instance. More broadly, the code is written so that new expressions or behaviors could be added just by using the current expressions as a template. And this can done without concern for breaking any of the existing code. We note that the RobotFace library is being released under the Lesser GPL (www.gnu.org), which means that the library (and any modifications thereof) must remain open-source and available to the public, although derivative work is permitted (please see ReadMe and license.txt in the library folder).

Mainly, we just wanted to produce a useful but replicable robotic face so that people could explore the space of robotic faces from both a functional and design standpoint, to facilitate research/scientific endeavors, and (long-term) to *hopefully make it easy for everyone, regardless of background or resources, to push the boundaries of human knowledge*. The more people who can contribute to that, the better off we all are.

III.b Orienting Servos/Horns

One tricky thing about using mass-produced servos for something like this, in terms of programming, is ensuring that the servos are oriented in a way so that they all have the same start position (e.g. neutral face) using the same programming code. To facilitate this, every servo horn should be attached so that when the servo position is 90 degrees, the horn is parallel with the servo body, as shown below.



Different servos may actually have different starting (init) values, e.g. the servos controlling the eyebrow raise height actually use 60° and 105° degrees (depending on whether it's right or left) as the init value. However, all servo horns should look like the above image when at 90° degrees. This greatly simplifies the production and consistent reproduction of the robot-face (MiRAE).

To facilitate this, we have written a short Arduino script for testing servo position, which is included on our website. It basically rotates the servo between three positions – 0°, 90°, 180° – with a slightly longer pause at 90 degrees. That way you can determine where the 90° position is, and affix the servo horn oriented as shown in the image above.

One last thing of note, since we are using mass-produced hobby servos, they are not all exactly the same, because of how the grooves align between the servo gear and the horn. This means that some servos/horns will actually be parallel at, e.g. 92° or 87°, rather than exactly 90°. This doesn't cause too many problems, but we do occasionally have to tweak the init values (listed in RobotFace Library ReadMe) for some of the servos up or down a couple degrees for each individual robot-face if we want it to be really precise (see the RobotFace Library ReadMe).

III.c RobotFace Library

The library has its own ReadMe, which contains a lot of information about usage and input parameters and other details. Here, we will just briefly discuss the library at a high-level, so that you can have a sense of what it is and what it's for.

The RobotFace library (as described in Section III.a) was originally built so as to encapsulate much of the common functionality for a robotic face that would be needed across applications. In that sense, its main purpose is to facilitate derivative work. The input parameter values (e.g. neck movement limits) that can be seen in the example Arduino sketch distributed with the library are currently tuned to work specifically with MiRAE, but they could be tweaked to work with any robotic face (given a similar set of facial features). This in keeping with an open source robotics philosophy.

The library includes functions for movement, facial expressions, communication with offboard computers/processors, and computer vision integration. The RobotFace library is written as a C++ library, so it canbe used with in any C++ program. We currently use it with open-source Arduino microprocessors, which have their own IDE based on C++. The RobotFace library can simply be imported at the start of any Arduino sketch, assuming it is installed in the Arduino libraries folder. There are additional details in Section III.a and the library's ReadMe file.

As mentioned in Section II.10, we also utilize computer vision (CV) with MiRAE, which this library supports. This typically entails use of an offboard computer to process the CV sensor input (e.g. OpenCV, written variably in C++ or Python) which is then communicated to the Arduino via the communication functions. In a loose sense (in mammalian brain terms), the offboard computer then can be thought of as the cortex, with the Arduino as the cerebellum. As discussed in Section II.10, we will likely be releasing more information and programming code related to this in the near future. We also have several ongoing research projects related to this, including placing interactive robotic faces into public art installations.