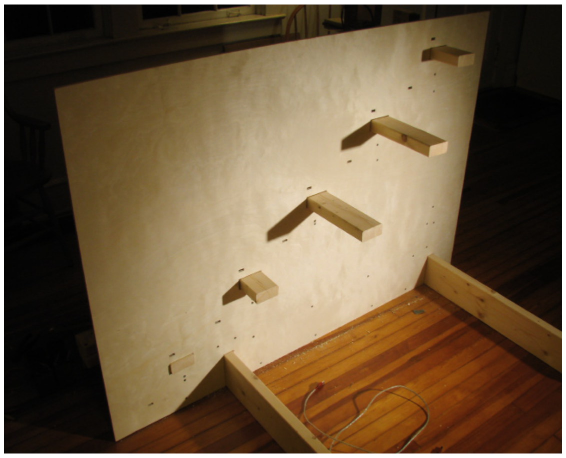
Electric Staircase
Prototype Design and Construction

Luke Kambic
MTI F09 Final Project

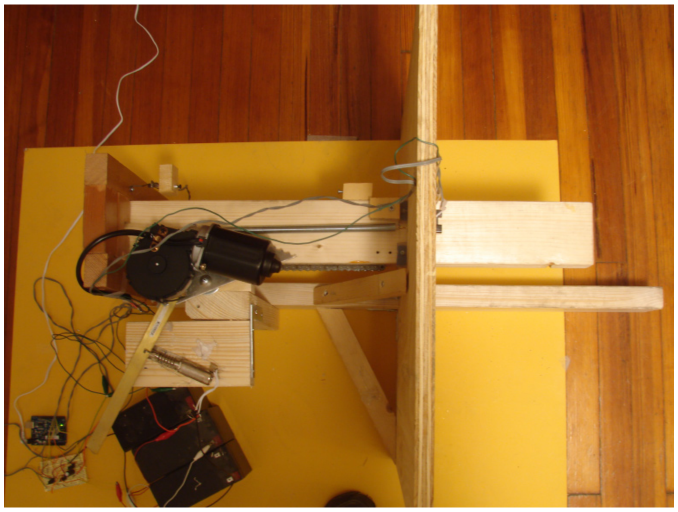
It's an experimental platform for tangible interaction that can be integrated into the walls of a structure. My main intention was a staircase that keeps its steps hidden in a wall and ejects them as users walk up, retracting them again as they pass. In certain situations this could free up a little floor space in a house, but it's mostly supposed to be cool and beautiful.

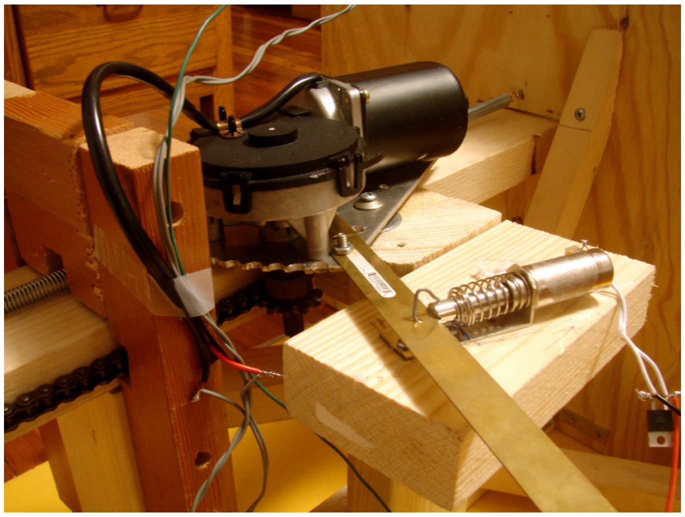


It was built on a recession budget. I wanted a functional prototype for testing at the absolute minimum cost using readily available parts at the expense of mechanical elegance and durability. Most of it is made of cheap pine.

Test Bed

My first idea for moving the steps was a linear gear. I wanted the step to come out of the wall as fast as possible, so rather than a reversible motor (which would need to be ridiculously powerful to move something the size of a step fast enough) I tried a tension spring and clutch. I built this test bed out of parts I had on hand:



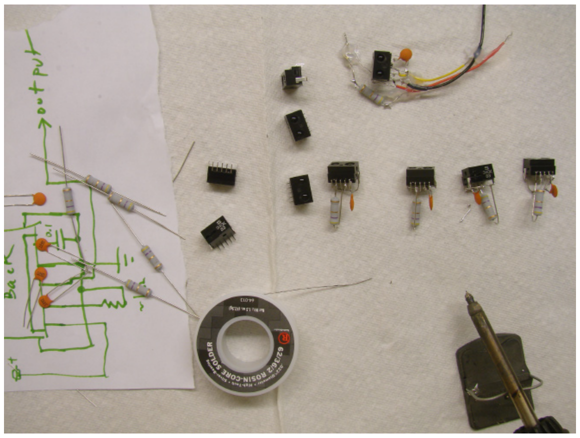


A sprocket on a power window motor engages with a length of #35 chain attached to a board at both ends by screws running through the links. The shapes of the mating parts make it easy to engage and disengage the sprocket without much force, but when it's engaged and delivering turning force it tends to stay there. Chains and sprockets are designed for some of this type of sliding so they probably won't wear too fast.

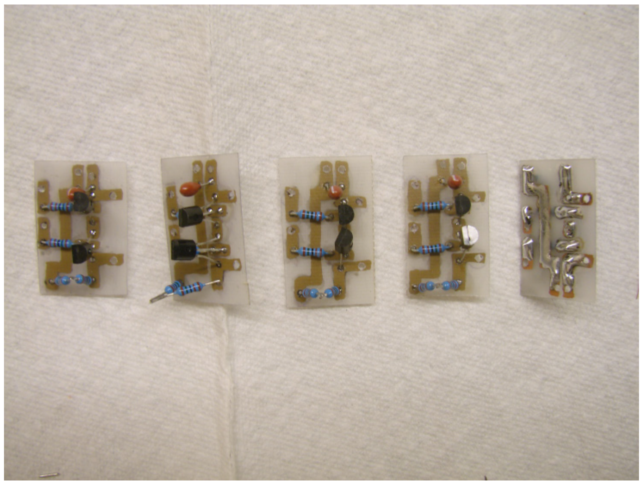
I stuck the motor on a plate with a single simple pivot point consisting of a screw through stacked washers. A cheap solenoid was attached to a lever arm to shift the motor and disengage the sprocket. The return spring of the solenoid provides enough force to re-engage it as long as the teeth are well aligned with the chain. The mechanism worked as a clutch and brake, because the motor uses a worm gear and its shaft won't budge when it's not under power.

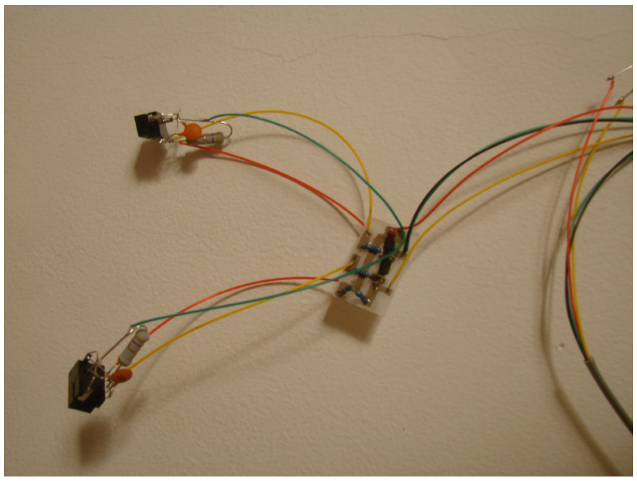
The motor drew the board back through a wooden guide (all the sliding parts were wood-on-wood, which worked fine) and pulled back the main spring. The spring I used was probably intended for screen doors. It drew the board back fast and hard when the sprocket disengaged, pulverizing the silicone stops I initially used.

Sensors



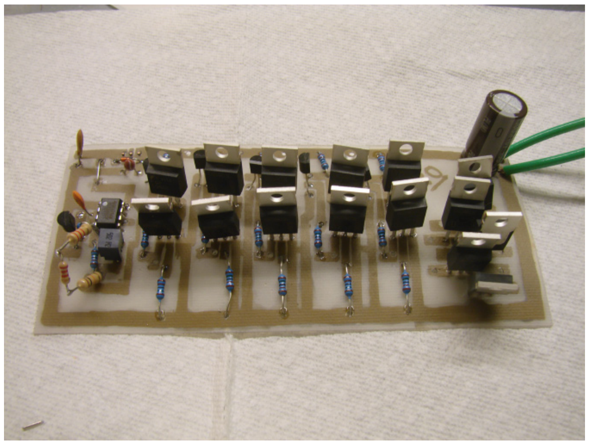
To keep costs and complexity down I used two sensors for each of 5 steps, one to detect a foot/leg as it swings up toward a step to trigger its ejection and another to hold the step out while someone stands on it. I chose the Sharp GP2Y0D340K infrared proximity sensor, which has a very narrow detection angle and detects from 40 cm to a cm or so. There aren't enough input pins on the Arduino Deumillanove to handle this and the other necessary stuff, but because both sensors would induce the same response I could feed both of them into one pin. Unfortunately, their two-state outputs are normally high and go low when they detect something, so I built NAND gates out of bipolar transistors to combine the signals of each pair.

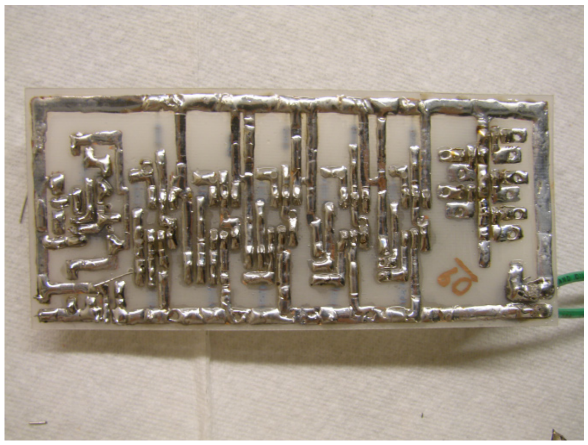

NAND boards


Paired sensors

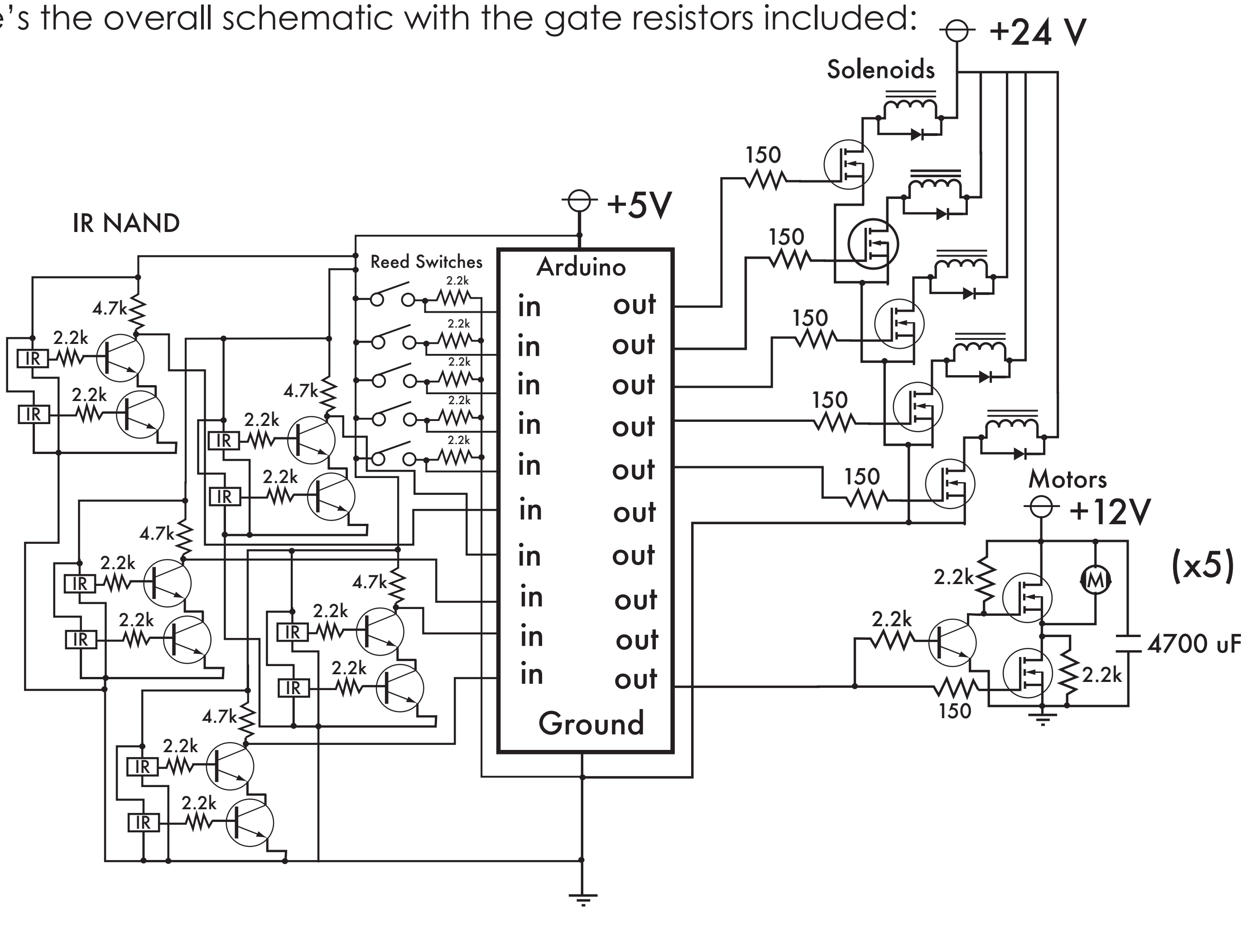
High-Current Switches for Actuators

The actuator driver board has 10 switching channels with 50 amp logic level MOSFETs (IPP065N03L). The 5 channels intended for the motors have a second MOSFET that turns on when the drive MOSFET turns off and connects the motor terminals. This causes it to decelerate faster when power is removed. I intended to drive the FETs directly from the Arduino with no resistors on the gates (to minimize the switching time), but this turned out to be a mistake, because the resulting current spikes were enough to kill the arduino outputs. I drew the connections on PCB stock with a sharpie and etched them with ferric chloride from Radio Shack, then beefed up the current-carrying capacity with solder. The IC on the left of the board is part of an oscillator for an auxiliary circuit I didn't end up needing.



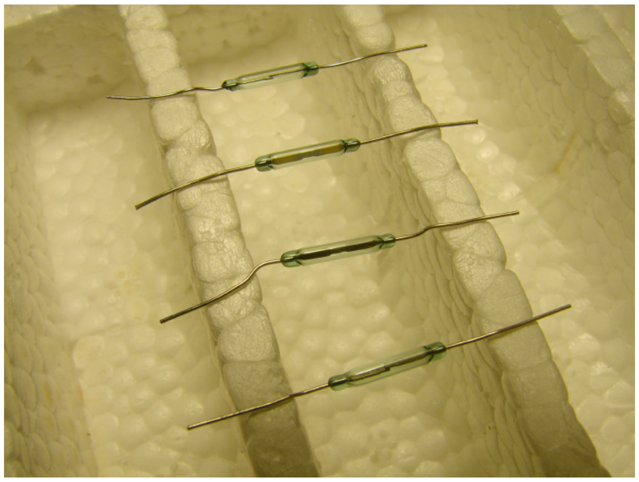


Here's the overall schematic with the gate resistors included:

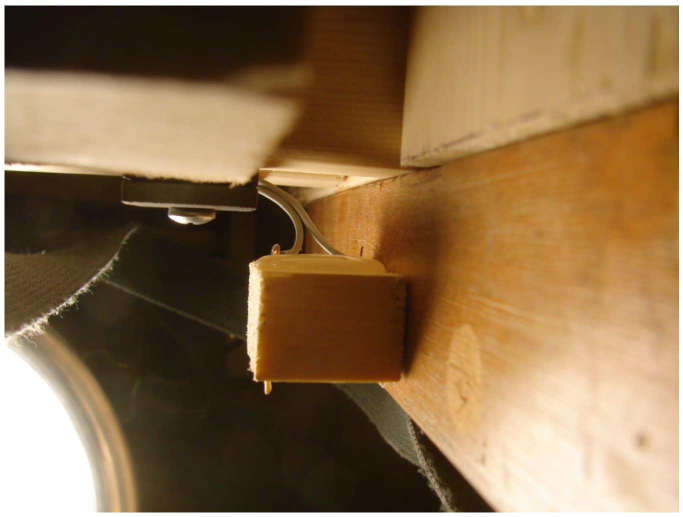


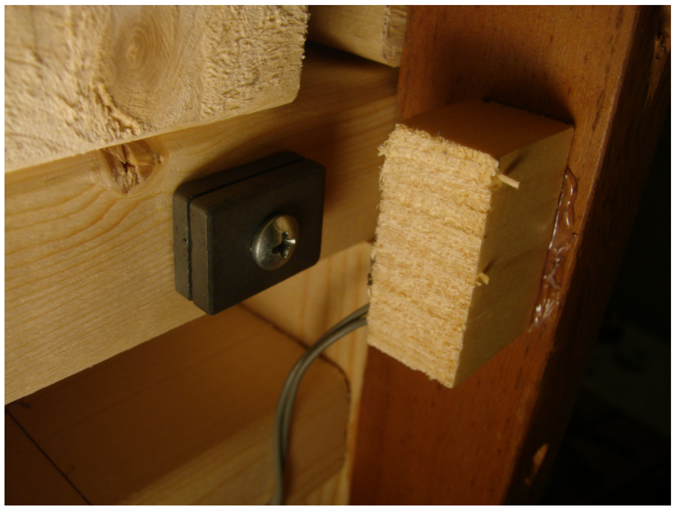
End-of-Retraction Sensors

The motors need to know to stop when they fully retract a step into the wall. I decided on magnetic reed switches attached to the stationary supports with magnets attached to the steps.

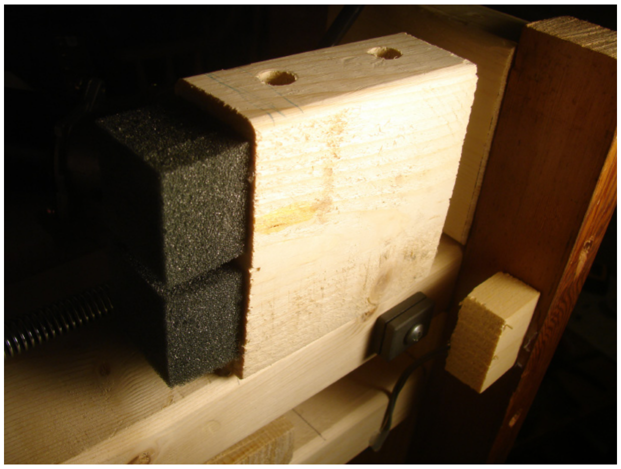


Reed Switches





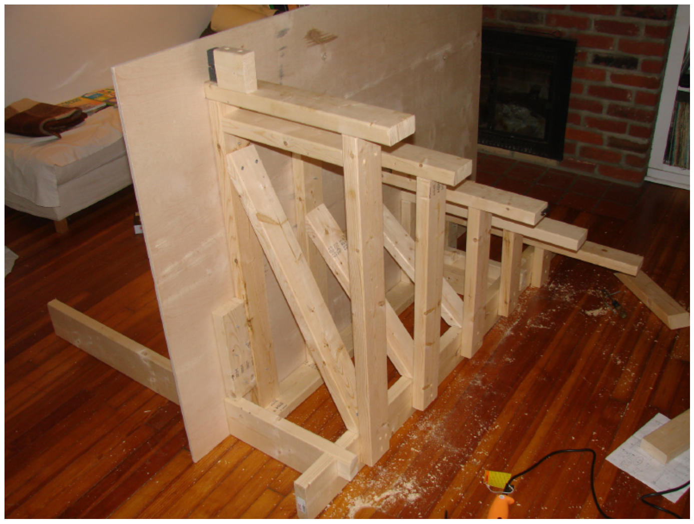
I encapsulated them in blobs of hot glue on pieces of wood to make them more rugged.

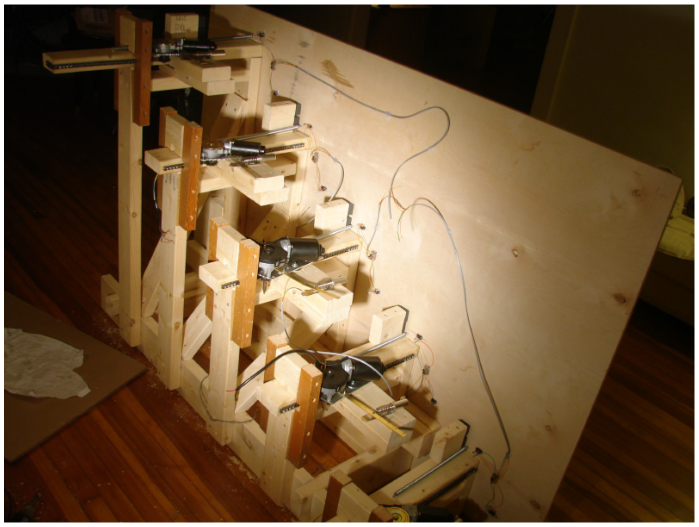


This photo also shows the stops, cubes of polyurethane foam glued to a sturdy chunk of wood attached to each step.

Support Frame

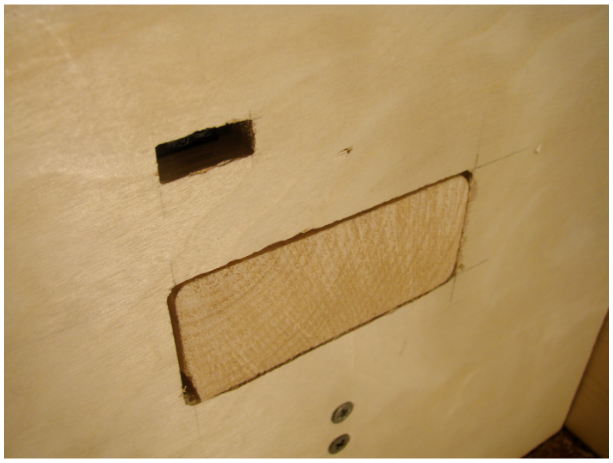
It's mostly pine 2x4s cut with a hand saw. The prongs that stick out of the front to keep it from tipping over slide into groves in the back so they can be removed easily to make the thing less cumbersome. The stops slam into the 1/2" thick plywood, which flexes it and also makes the whole structure jump forward about half an inch. The stairs feel pretty solid underfoot.





The downward force from the steps goes on the vertical 2x4s rather than the edges of the holes in the wall. The back ends are caged in sturdy slots.

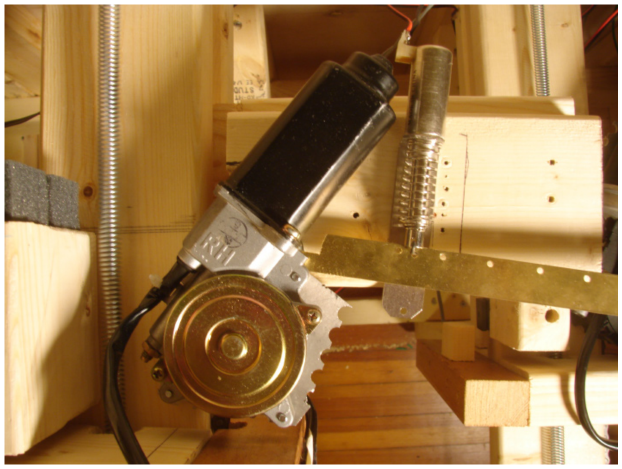
Sensor and Step Holes

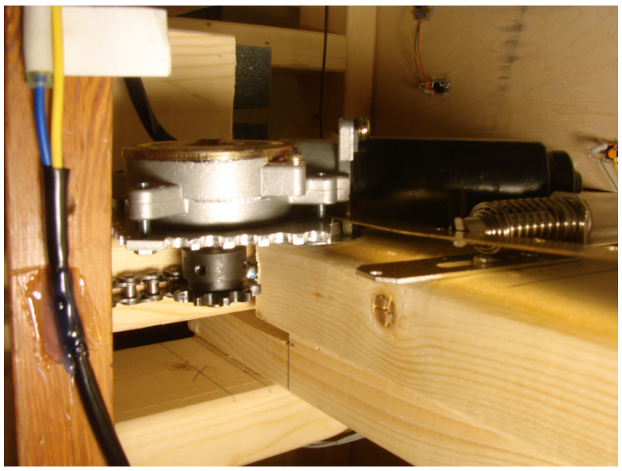


I cut all the holes in the plywood "wall" with a hand drill, recipracating saw and hand file. The sensors are recessed almost a full half inch, which protects them without affecting their operation.

Actuators

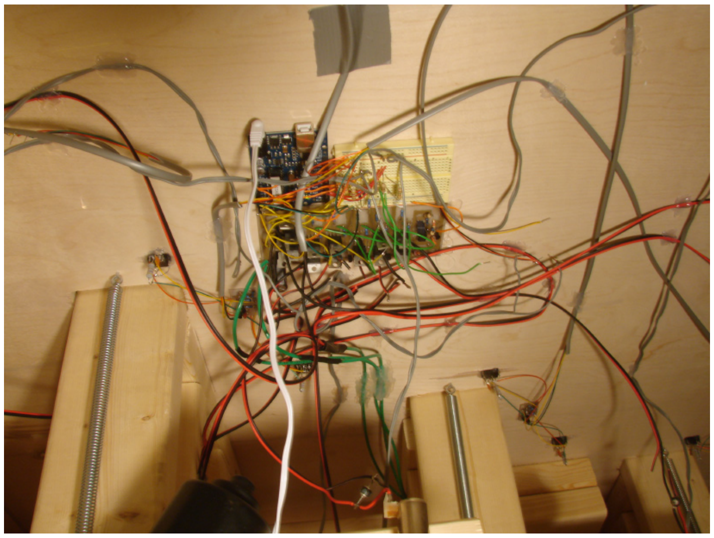
They're configured like the one in the test bed, but with sturdier supports.

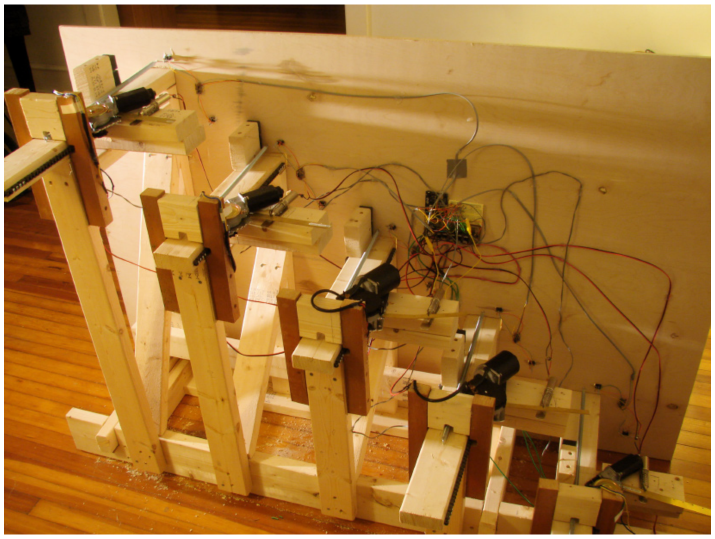




Wiring and Power

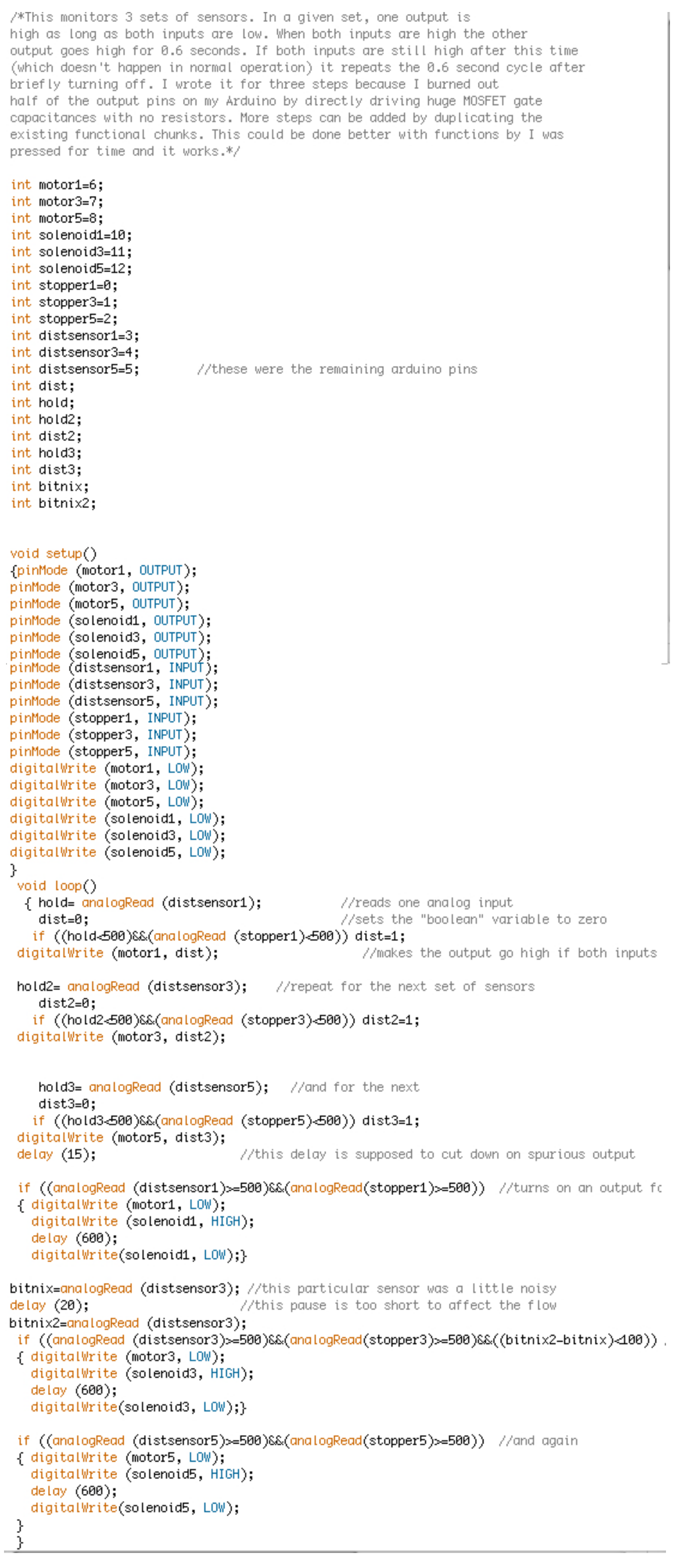
The actuators are connected to the switches with 20 AWG stranded zip cable. Power gets to the switchboard through 16 AWG cables. Currently, the power source is a pair of 12V 7Ah sealed lead acid batteries. The sensor signals get to the Arduino through 4-conductor telephone cable.


Central rat's nest



Programming

The code is trivially simple for the moment. It turns on a motor output if its associated reed switch is low and if its sensors detect nothing. If the step is parked with the spring fully extended and the reed switch high, a signal from the sensors triggers the solenoid to activate for 0.6 seconds, long enough for the step to fully eject.



```
/*This monitors 3 sets of sensors. In a given set, one output is
high as long as both inputs are low. When both inputs are high the other
output goes high for 0.6 seconds. If both inputs are still high after this time
(which doesn't happen in normal operation) it repeats the 0.6 second cycle after
briefly turning off. I wrote it for three steps because I burned out
half of the output pins on my Arduino by directly driving huge MOSFET gate
capacitances with no resistors. More steps can be added by duplicating the
existing functional chunks. This could be done better with functions by I was
pressed for time and it works.*/

int motor1=6;
int motor3=7;
int motor5=8;
int solenoid1=10;
int solenoid3=11;
int solenoid5=12;
int stopper1=0;
int stopper3=1;
int stopper5=2;
int distsensor1=3;
int distsensor3=4;
int distsensor5=5;        //these were the remaining arduino pins
int dist;
int hold;
int hold2;
int dist2;
int hold3;
int dist3;
int bitnix;
int bitnix2;


void setup()
{pinMode (motor1, OUTPUT);
pinMode (motor3, OUTPUT);
pinMode (motor5, OUTPUT);
pinMode (solenoid1, OUTPUT);
pinMode (solenoid3, OUTPUT);
pinMode (solenoid5, OUTPUT);
pinMode (distsensor1, INPUT);
pinMode (distsensor3, INPUT);
pinMode (distsensor5, INPUT);
pinMode (stopper1, INPUT);
pinMode (stopper3, INPUT);
pinMode (stopper5, INPUT);
digitalWrite (motor1, LOW);
digitalWrite (motor3, LOW);
digitalWrite (motor5, LOW);
digitalWrite (solenoid1, LOW);
digitalWrite (solenoid3, LOW);
digitalWrite (solenoid5, LOW);
}
 void loop()
  { hold= analogRead (distsensor1);            //reads one analog input
    dist=0;                                    //sets the "boolean" variable to zero
   if ((hold<500)&&(analogRead (stopper1)<500)) dist=1;
 digitalWrite (motor1, dist);                     //makes the output go high if both inputs

 hold2= analogRead (distsensor3);    //repeat for the next set of sensors
    dist2=0;
   if ((hold2<500)&&(analogRead (stopper3)<500)) dist2=1;
 digitalWrite (motor3, dist2);


    hold3= analogRead (distsensor5);   //and for the next
    dist3=0;
   if ((hold3<500)&&(analogRead (stopper5)<500)) dist3=1;
 digitalWrite (motor5, dist3);
 delay (15);                      //this delay is supposed to cut down on spurious output

 if ((analogRead (distsensor1)>=500)&&(analogRead(stopper1)>=500))  //turns on an output fc
 { digitalWrite (motor1, LOW);
   digitalWrite (solenoid1, HIGH);
   delay (600);
   digitalWrite(solenoid1, LOW);}

bitnix=analogRead (distsensor3); //this particular sensor was a little noisy
delay (20);                      //this pause is too short to affect the flow
bitnix2=analogRead (distsensor3);
 if ((analogRead (distsensor3)>=500)&&(analogRead(stopper3)>=500)&&((bitnix2-bitnix)<100))
 { digitalWrite (motor3, LOW);
   digitalWrite (solenoid3, HIGH);
   delay (600);
   digitalWrite(solenoid3, LOW);}

 if ((analogRead (distsensor5)>=500)&&(analogRead(stopper5)>=500))  //and again
 { digitalWrite (motor5, LOW);
   digitalWrite (solenoid5, HIGH);
   delay (600);
   digitalWrite(solenoid5, LOW);
 }
 }
```