Kai Chorazewicz
Michael McNabb
102B Group 7

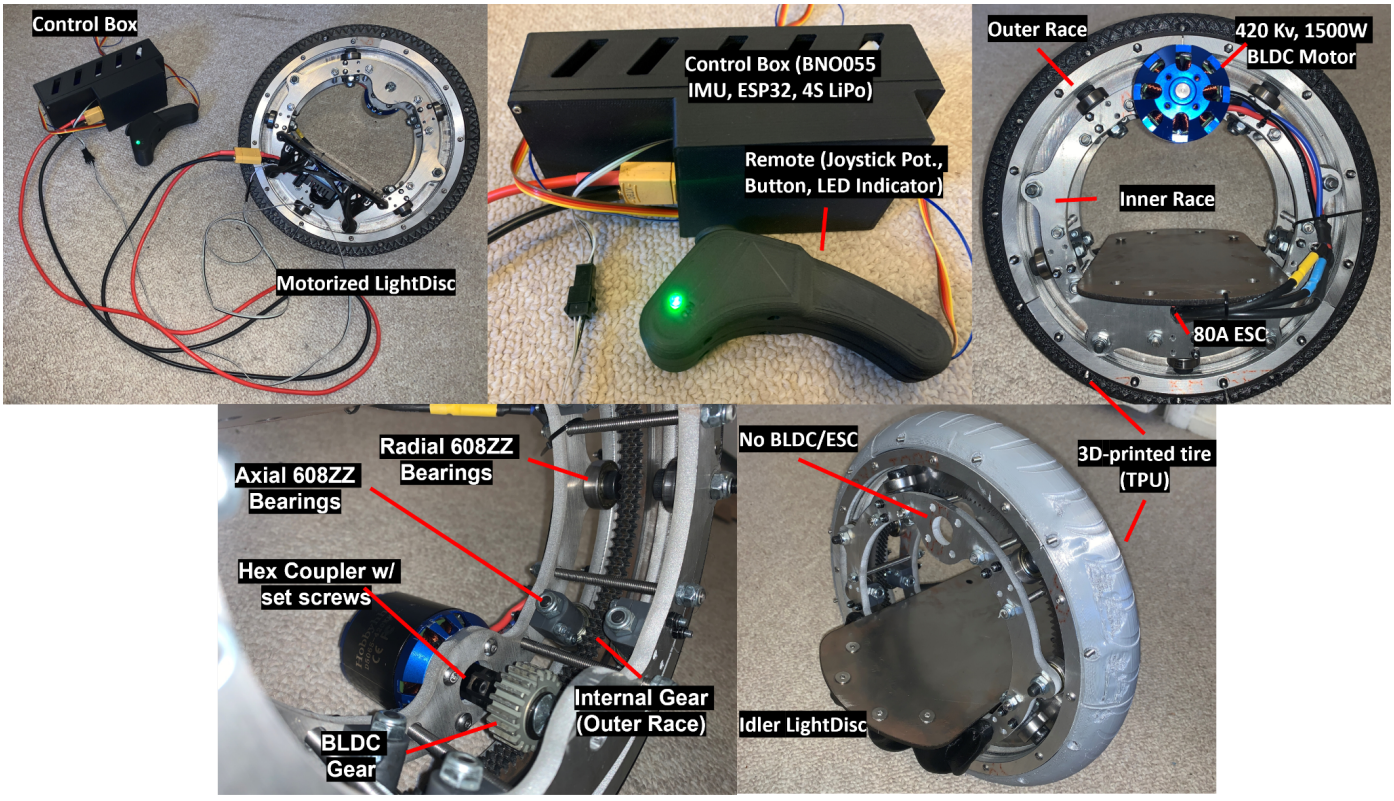
P4 LightDiscs Report

Summary

We chose to address the opportunity of creating a compact personal mobility platform to enable easier urban transportation for students, and achieved this with the LightDisc system. Our high-level strategy was to have two discs controlled by a control box and remote, and drive on various surfaces. We intended for the LightDisc system to have a 15 mph wheel speed, and we achieved a 17.2 mph wheel speed (calculations in subsequent sections).

However, we pivoted to a 3D-printed polyurethane tire over a pressurized tube tire to enable custom sizing and rigidity of the tire. Also, we pivoted to only one motor in the “leader” LightDisc, which would pull the rider forward, while the “follower” is not motorized. We did this for controls and electronics simplicity, but also safety and stability, as having a pushing disc or both discs motorized could cause the rider to fall much more. Additionally, while we intended to calculate velocity by integrating an IMU, this proved to be extremely difficult due to sensor drift creating unacceptable error in velocity, and we instead used an open-loop controls system with the rider’s remote “closing” the loop.

Device Overview



Device Decisions

For actuation of the LightDisc, as mentioned before, we chose to motorize one of the discs for controls simplicity and improved safety. We chose a brushless DC motor due to their high power output and high efficiency with large loads, which would be necessary to transport a person. The wheel speed measured with a laser tachometer, at no load with full remote throttle (which was scaled to around ⅔ of the actual throttle of the motor for safety), was 550 rpm. With a 10.5 inch diameter wheel, this means: $v_{disc} = 550\ \frac{rot}{min} * \frac{60\ min}{1\ hr} * \frac{\pi * 10.5\ in}{1\ rot} * \frac{1\ mile}{63360\ in}$ = 17.2 mph. Meanwhile, the ideal max wheel speed given battery voltage of 14.4 V and Kv of 420, was:

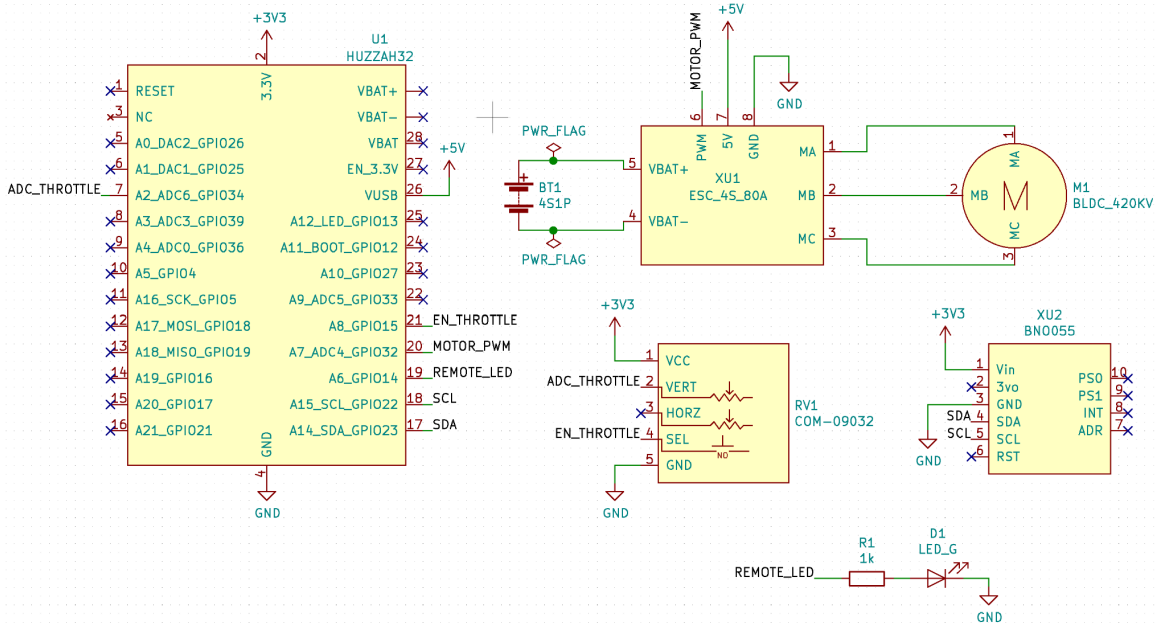
$v_{ideal} = 420 \frac{rpm_{gear}}{V} * \frac{1\ rot\ (PD_{internal\ gear})}{6.7\ rot(PD_{motor\ gear})} * 14.4\ V * \frac{60\ min}{1\ hr} * \frac{\pi * 10.5\ in}{1\ rot} * \frac{1\ mile}{63360\ in} * \frac{2}{3}$ = 18.8 mph. So although the target speed of 15 mph was reached, there was some discrepancy due to losses in the bearings and transmission, but the 420 Kv motor we chose was sufficient for our speed and power requirements.

For the bearings loads, we assumed 4 bearings to be in contact at any time, due to deflections within the races. This meant that with a max load of 100kg, the max load a bearing would see: $F_{bearing} = \frac{100\ kg * 9.81\ m/s^2}{4\ bearings}$ = 245.3 N. Given a 608ZZ bearing is rated for 1334.47N (300 lbf) as per McMaster-Carr, the safety factor here is 5.44. Considering that there are many more than 4 bearings per disc resolving forces at any given time in both axial and radial directions, this safety factor is likely even greater.

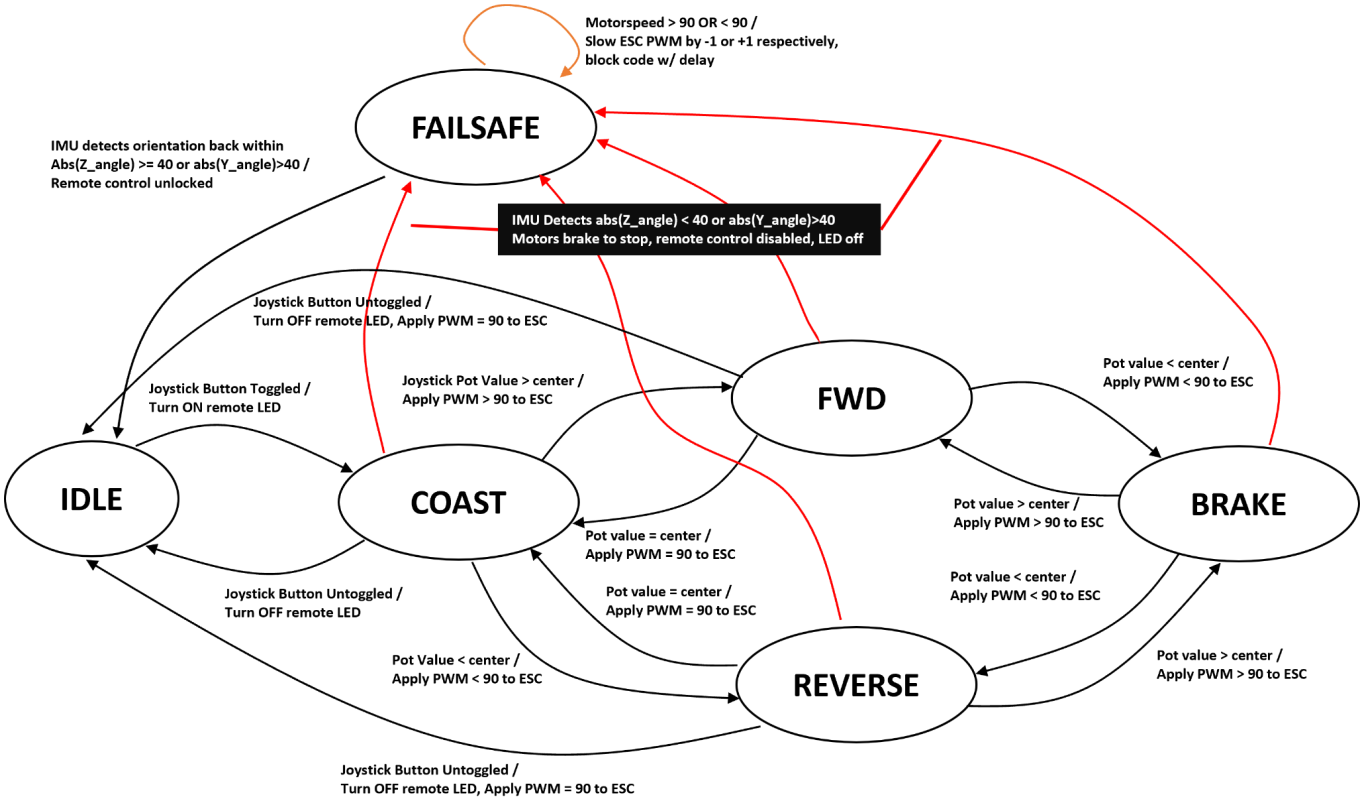
Mechanically, the device is assembled with threaded rods clamping the axial thrust bearings of the assembly against the internal bearing surface, guaranteeing solid contact even at extreme turning angles. These threaded rods function as both locating features and structural members, to reduce costs while still meeting the tolerances required within this assembly.

Software & Hardware

Circuit diagram



State transition diagram



Reflections

In hindsight, what worked well for us was pivoting to one motorized disc, as this was plenty to move a person. Also, the TPU 3D-printed tires instead of tubes were a good decision, as they allowed tunability of flexure and size, instead of having a size constraint of a purchased tire. The aluminum parts also worked great as they were extremely strong and light, similarly to the generative design for the foot supports, which allowed for 3D-printed foot supports that were optimized for the footplate. Also, designating tasks and subsystems to work on based on each team members' strengths worked very well, and allowed us to fabricate and prototype the final design in less than a month.

What we could have done better is using a better system for mounting feet to the plate, with something like straps or velcro, to ensure greater stability. Also, a better coupler between the BLDC and the spur gear could have been used, such as a spring pin instead of the set screws we used. Additionally, we should have avoided using captive nuts, and instead used threaded inserts, which are much more convenient and easy to install. Also, the mild steel used for the internal gear made the design fairly hefty and not very portable, so we could have used aluminum instead. We had chosen steel to minimize noise and wear from the steel bearings, but this turned out to be overkill and aluminum would have been sufficient.

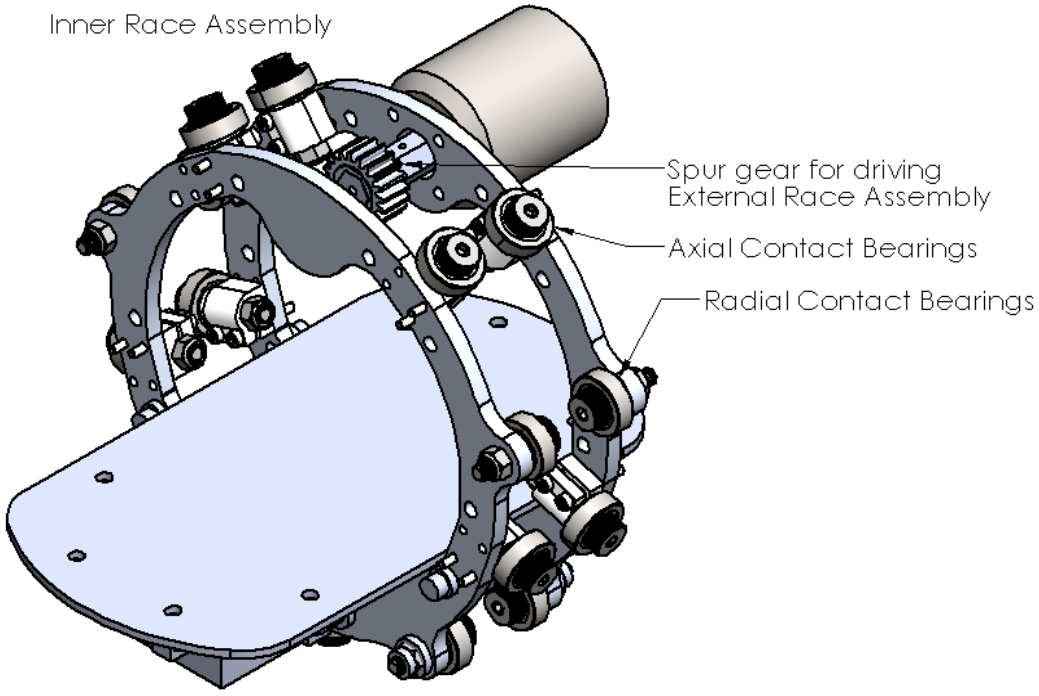
APPENDIX

Bill of Materials

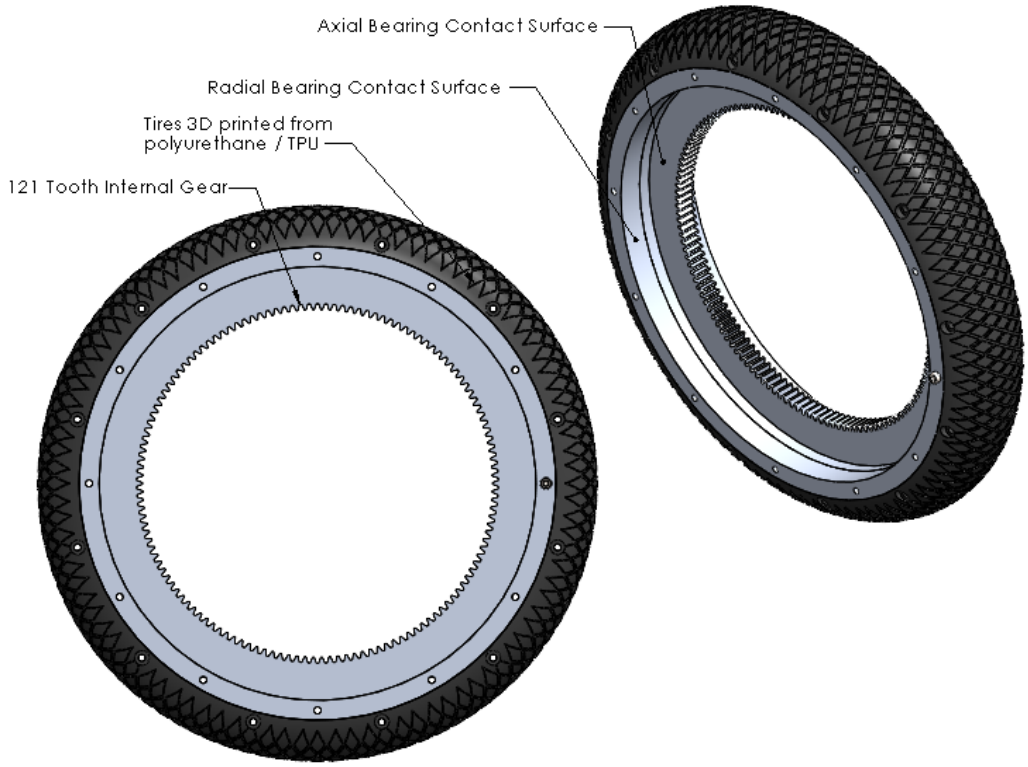
Note: Purchase Links omitted for formatting reasons. Full BOM located at https://docs.google.com/spreadsheets/d/1YPpNyvbBt36CzO6ezv6KbZ6A3B11mRnaP4VYTzj7Mlg/edit?usp=sharing

Item Name	Description	SKU	Quantity	Vendor
Shoulder Bolts	M6 thread 8mm OD x 20mm lg. Shoulder Screws	92981A203	20	McMaster
BLDC Motor	400KV, 1500W D5065 Brushless DC Motor	-	1	Amazon
80A ESC	80A Electronic Speed Controller	-	1	Amazon
Aluminum Plate 1/4"	Used for making all frame components	-	1	Jacobs Hall Material Store
Shoulder Bolts	M6 thread 8mm OD x 15mm lg. Sholder Screws	92981A752	16	McMaster
608ZZ Bearings	22mm OD 8mm ID Skateboard Bearings	-	36	Amazon
M5 Assembly Allthreads	M5 x 170mm to be cut into 10x 85mm for each disc	-	20	Amazon
Motor Mounting Bolts	M4 x 15mm SHCS Bolts	-	4	On-hand
M6 Locknuts	M6 lock nuts	90576A115	36	McMaster
M5 Jam Nuts	M5 Jam nuts	90695A037	20	McMaster
M3x55mm Bolts	M3 x 55mm bolts, button head cap screw to distribute forces	92095A120	48	McMaster
M3 Locknuts	M3 Nylock nuts	-	36	Amazon
M3 x 25mm Bolts	M3 socket-head cap screw bolts for mounting support bearings	90128A205	40	McMaster
M5 Locknuts	M5 Nylock nuts	90576A104	36	McMaster
Steel Plate 1/8"	24" x 24" x 1/8" Mild Steel Plate	-	1	Jacobs Hall Material Store
Axial Bearing Blocks	Bearing blocks for axial shoulder bolts + bearings, 3D printed (Tough 2000)	-	20	3D-printed
Belleville Washers	Washers for 608ZZ bearings, 3D printed (Tough 2000)	-	20	3D-printed
Shaft Coupler	Hex Stock motor shaft coupler	-	1	On-hand
Internal Spur Gear	20 tooth Hex stock spur gear, 1" PD	-	1	On-hand
Tire Tread	3D-printed TPU tire tread, 10.5" OD	-	6	3D-printed
Set Screws	#4-40 Set Screws	-	2	On-hand

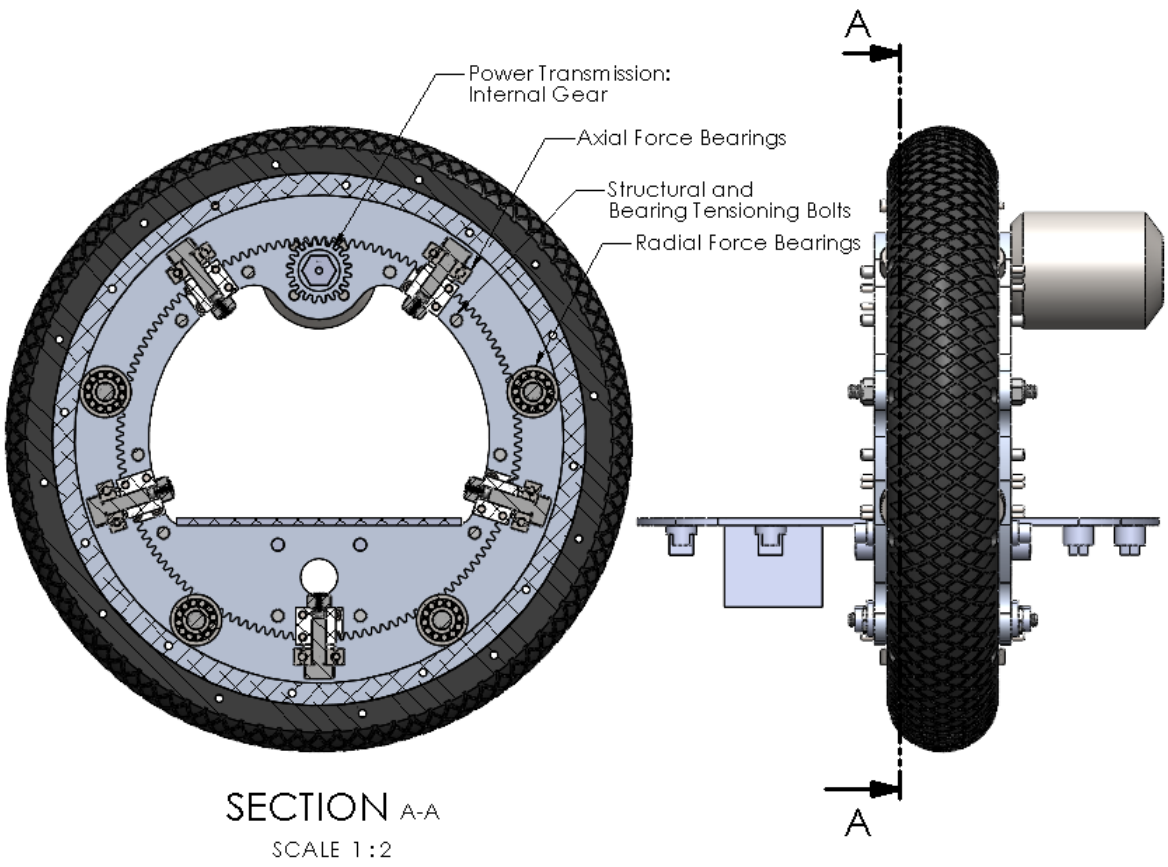
CAD



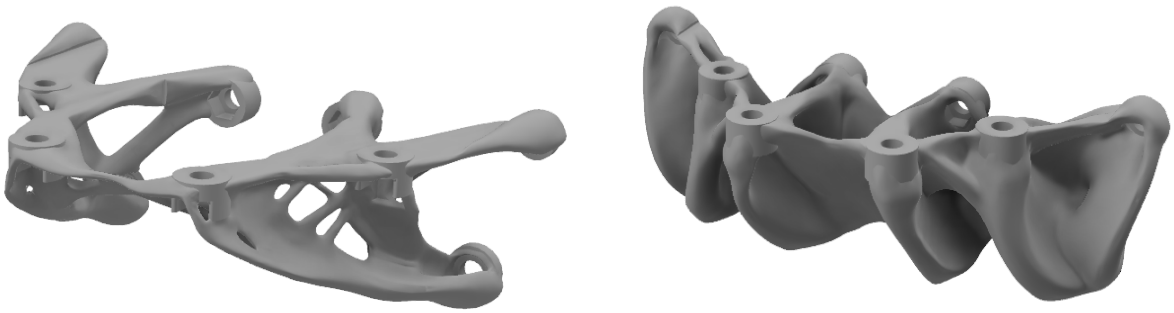
Inner Race Assembly CAD.



Outer Race Assembly.



Assembly CAD and cross sectional view. Generative design models not shown due to modeling complexity.



Generative Design models used for front and back brackets to support foot plate to inner race, respectively.

LightDisc Software

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <ESP32Servo.h>

#define PWM_PIN 32
#define POT 34 //A2
#define EN_T 15
#define LED 14
#define BNO055_SAMPLERATE_DELAY_MS (100)

Adafruit_BNO055 bno = Adafruit_BNO055(-1, 0x28);
Servo M1;
int motorSpeed = 0;
bool failsafeTriggered = 0;
bool remoteEn = 0;
unsigned long prevMillis = 0;
enum STATE{
  IDL = 0,
  COAST = 1,
  FWD = 2,
  REV = 3,
  BRAKE = 4,
  FAILSAFE = 5
};
STATE state = IDL;

void checkOrientation(){ //check IMU orientation to see whether rider fell off
  imu::Vector<3> euler = bno.getVector(Adafruit_BNO055::VECTOR_EULER);
  Serial.println("y:"+String(euler.y())+",z:"+String(euler.z()));

  if(abs(euler.z())<40 || abs(euler.y())>40){ //orientation off z or y indicates rider fell off
    state = FAILSAFE;
    Serial.println("Failsafe enabled!");
  }

  else if(failsafeTriggered && abs(euler.z())>40 && abs(euler.y())<40){
    state = IDL;
    Serial.println("Failsafe deactivated!");
    failsafeTriggered = 0;
  }
}

void IRAM_ATTR isr() {  // remote button interrupt
  static unsigned long last_interrupt_millis = 0;

  if(millis() - last_interrupt_millis > 500){
    last_interrupt_millis = millis();
    remoteEn = !remoteEn;
    Serial.print("Button pressed, ");

    if(remoteEn){
      Serial.println("enabling!");
```

```
      digitalWrite(LED,HIGH);
      state = COAST;
    }
    else{
      Serial.println("disabling!");
      digitalWrite(LED,LOW);
      state = IDL;
    }
  }
}

void setup() {
  digitalWrite(LED,LOW);
  M1.attach(PWM_PIN);
  pinMode(LED,OUTPUT);
  pinMode(POT, INPUT);
  pinMode(EN_T, INPUT_PULLUP);
  attachInterrupt(EN_T, isr, FALLING); //remote button pulls to ground when pressed
  Serial.begin(115200);

  if(!bno.begin()){
    Serial.print("No BNO055 detected!");
    while(1){
      Serial.print(".");
      delay(500);
    }
  }
  bno.setExtCrystalUse(true);
}

void loop(){
  float val = analogRead(POT); //check analog value of potentiometer

  switch(state){
    case COAST:
      if(val < 1710) state = FWD;
      else if(val > 1760) state = REV;
      else{
        motorSpeed = 90;
        M1.write(motorSpeed);
      }
      break;

    case FWD:
      if(val > 1760) state = BRAKE;
      else motorSpeed = -1*(int)(pow(val-1720,1) / ((pow(4095-1720,1)/50))) + 90; //throttle fwd
mapping
      Serial.println("FWD,servo:"+String(motorSpeed));
      M1.write(motorSpeed);
      if(motorSpeed = 90) state = COAST;
      break;

    case REV:
      if(val < 1710) state = BRAKE;
      else motorSpeed = (int)(pow(1720-val,1) / (pow(1720,1)/20)) + 90; //throttle reverse
```

```
mapping
      Serial.println("REV,servo:"+String(motorSpeed));
      M1.write(motorSpeed);
      if(motorSpeed = 90) state = COAST;
      break;

    case BRAKE:
      M1.write(motorSpeed);
      delay(500);
      break;

    case FAILSAFE:
      if(!failsafeTriggered){ //code-blocking for-loops to ensure lightdisc slows down
        digitalWrite(LED,LOW);
        if(motorSpeed > 90){
          for(int i = motorSpeed; i >= 90; i--){
            M1.write(i);
            delay(25);
          }
        }else{
          for(int i = motorSpeed; i <= 90; i++){
            M1.write(i);
            delay(25);
          }
        }
        failsafeTriggered = 1;
      }
      else{ //simple flashing indicator on remote once disc has slowed down
        digitalWrite(LED,HIGH);
        delay(500);
        digitalWrite(LED,LOW);
        delay(500);
      }
      break;

    default:
      motorSpeed = 90; //central PWM value for "coast" is 90 on ESC
      M1.write(motorSpeed);
  }

  if(millis() - prevMillis > BNO055_SAMPLERATE_DELAY_MS){ //non-blocking sampling of BNO055 IMU
    checkOrientation();
    prevMillis = millis();
  }
}
```