Basketball Return Optimizer

Brian Acker, CSE, Derek Foster, EE, Adam Paranay, EE, and Devon O'Rourke, CSE

Abstract—**In an era where automation is becoming a necessity in almost all facets of life, it is no surprise that the Basketball Return Optimizer's (BRO) main feature is automation via target tracking. Basketball return systems have been developed to help return the basketball to the player after they have made a shot. These return systems are meant to maximize the player's time shooting while limiting the time they have to retrieve the shot basketball. Unfortunately, the player still must manually adjust these systems to control where on the court the ball will be returned. This is an inefficiency that BRO addresses. As it stands, no recreational or professional system can track a player and return the ball to said player no matter where they stand on the court. BRO is a traditional return funnel system that is modified to maximize the player's shooting time by using automatic tracking. By taking the funnel, mechanizing it and integrating a webcam that tracks the player, our team has created a system that allows the player to freely move around the court and have the ball returned to them regardless of position.**

I. Introduction

TRADITIONAL basketball return systems, whether recreational, commercial, or professional do not maximize the player's time shooting because of the limitation of where the ball is returned on the court. The main problem that our team is addressing is that time spent practicing in basketball is often wasted by retrieving the basketball after shooting attempts. This problem does not need to be addressed however, we as a team feel that with our solution, the sport itself could reach new heights in terms of refining player's skill levels. Many existing products do this and we hope to build off of these designs to create an even more efficient system.

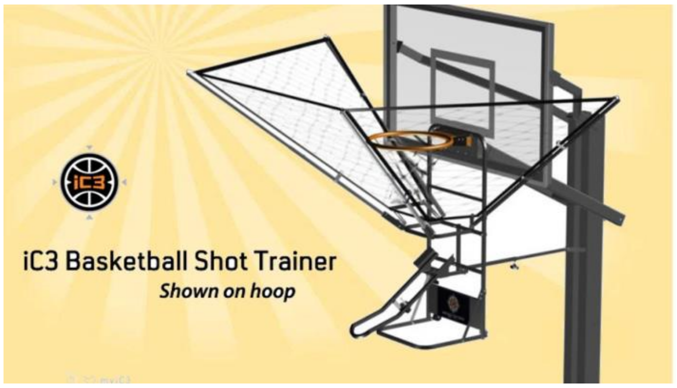


Figure A: iC3 Basketball Return System by Airborne Athletics

According to a study done by Airborne Athletics, their return system, the iC3 manages to triple the amount of shots possible within an hour [1]. Our system works on a recreational and professional level. Current return systems that professionals use only allow for pre-programmed return positions. With our system, professionals would have the freedom to shoot wherever they want while increasing their shots per hour. Our system is designed to better utilize a player's practice time so that they can take more shots per hour, allowing them to develop their skills faster and more efficiently.

As mentioned above, Airborne Athletics have created a system that effectively triples the shots a player can take per hour. Yet their system does not support free-form movement around the court because the iC3 only returns the basketball in one direction and requires manual adjustment to change that direction. The iC3 system has a retail price of $349.99. Another company, Goalrilla, has created a basketball return system using one large net that acts as a ramp. This ramp allows for the ball to be returned to anywhere on the free throw line. This system, like the iC3, does not accommodate free movement about the court because the ball is returned to a predefined location. This system retails at $79.50 [2]. The last system we have used for reference is the Dr. Dish Rebel. This system is a top of the line product meant for professional use. This system returns the ball using pre-programmed spots that the player or coach decides upon before starting up the machine. The Rebel not only returns the basketball but it does so in a chest-pass form, which represents how the shooter would realistically receive their passes. This system retails for $3,999.99 [3].

TABLE I
General Requirements

Specification	Value
Tracking Distance	5-25 feet from rim
Tracking Accuracy	100%
Operation Time	>1 hour
System Integrity	Withstands direct hit from basketball
Weight	<15lbs
Setup/Teardown	<5 minutes

By studying the current market, our team was able to identify a shortcoming common to existing products. No existing system can return the basketball to the player regardless of their position on the court in real-time. Our system implements the solution to this problem by tracking the player on the court via a camera and processing the image to find the direction in which the ball should be returned. Our team believes that this feature is desirable and marketable because it allows basketball players increased flexibility in where they shoot from on the court. Unlike existing systems, BRO ensures that the ball is always

D. Foster from Stoneham, MA (e-mail: derekf@umass.edu).
A. Paranay from Dracut, MA (e-mail: aparanay@umass.edu).
B. Acker from Norton, MA (e-mail: backer@umass.edu).
D. O'Rourke from Bourne, MA (e-mail: dporourk@umass.edu).

returned to the player without requiring the player to manually adjust the system.

II. Design

A. Overview

Our overall design revolves around the SKLZ funnel return system [4]. This funnel attaches to the rim using four hooks and straps. The straps connect to a disc around which the funnel rotates. The funnel is manually set to return the ball to a certain location on the court. By taking this $29.99 return system and modifying it, we made it possible to track a player in real-time while returning the ball to the player at any position in front of the basketball hoop.

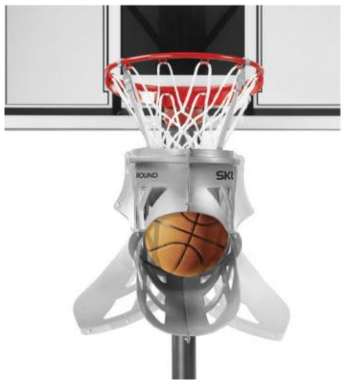


Figure B: SKLZ Funnel Return System

By replacing the funnel's disc with a 3D printed gear, we attached a motor and pinion gear that will rotate the funnel around the 3D printed gear. The motor is controlled by power signals regulated by the BeagleBone Black. The BeagleBone Black decides what signals to send to the motor with the help of the camera [5]. If the player is in the middle of the camera frame, then the funnel does not need to move so no power is supplied to the motor.

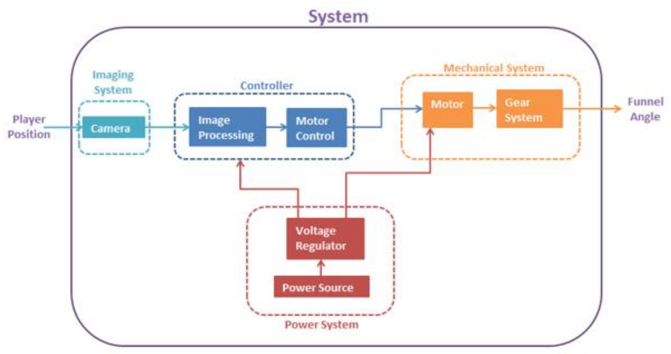


Figure C: Block Diagram

If the player is to the left or right of the camera's center, then either positive or negative power will be sent to the DC motor resulting in either a clockwise or counterclockwise rotation of the funnel.

The BRO consists of four sub-systems: imaging, controller, power, and mechanical. We will further break down each of the sub-systems. The imaging sub-system consists of the webcam, which takes the pictures at a rate of 30 frames per second. The controller sub-system processes the images and sends a signal that tells the motor what voltage to pull. The power system regulates the different powers and voltages needed to run the BeagleBone and the motor. The mechanical system turns the funnel using the motor and 3D printed gears.

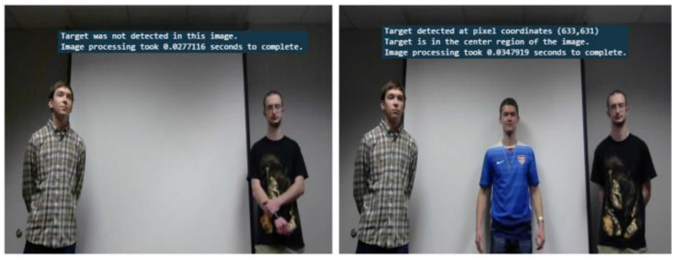
B. Block 1: Imaging System

The imaging system is the simplest block of our overall system and only consists of a camera for capturing images of the shooting area of the basketball court. These images are then sent to the controller for processing. The camera is able to take images with enough clarity to analyze pixel colors of objects at distances up to sixty-five feet, and it sends images to the controller at a rate of thirty frames per second.

The camera we chose to use is the Logitech c270 720p HD Webcam [6]. This camera is perfect for our project because it is a small, lightweight and inexpensive webcam. Thus, it does not weigh down our system and it left us with plenty of money left to spend on other components of our project. The 720p resolution is more than clear enough for us to see color details of objects at far distances and having fewer pixels than a 1080p camera allows us to process images significantly faster than we could with a HD webcam that captures more pixels. The webcam plugs into our BeagleBone Black controller via USB which allows for easy integration with the controller and easy connection even when mounted on our funnel.

TABLE II
Imaging System Requirements

Specification	Value
Image capture distance	5-25ft
Capture rate	>=5 frames/sec
Imaging processing time	<=200ms
Resolution	720p



```
Target was not detected in this image.
Image processing took 0.0277116 seconds to complete.
```

```
Target detected at pixel coordinates (633,631)
Target is in the center region of the image.
Image processing took 0.0347919 seconds to complete.
```

Figure D: Demo of Imaging System Basic Concepts

In Figure D, we show the early stages of the color filtering algorithm and what the camera was seeing. In Figure F, we show an example of what the camera would see in a single frame captured. The jersey has a specific blue, green, red pattern that the color filtering code is looking for and this can be seen in Figure E.

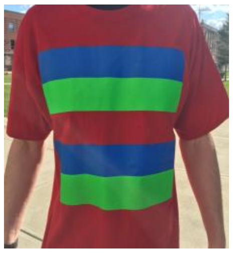


Figure E: Jersey containing specific pattern

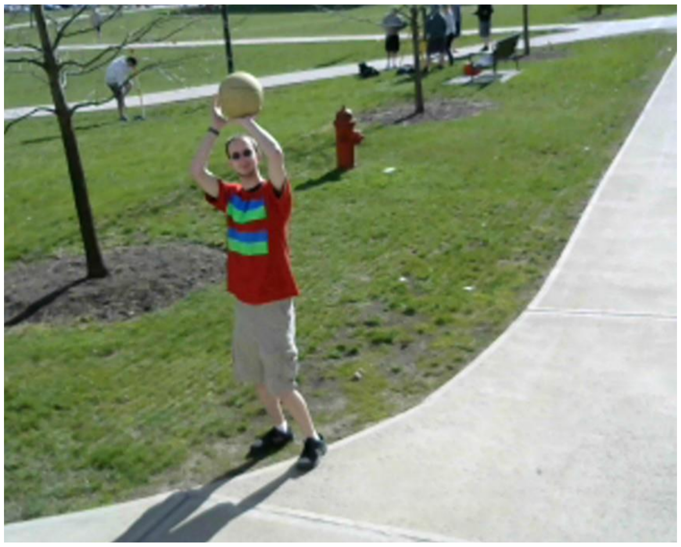


Figure F: Image of shooter in frame as seen by the camera

C. Block 2: Controller

The controller block of our system is a very significant part as it is responsible for processing the images taken by the camera and sending signals to the mechanical system indicating how it should move. The technology that we chose to use for this block is the BeagleBone Black Rev C microcontroller. We chose this controller over other options, such as Arduino and Raspberry Pi, primarily for its processor speed of 1 GHz and its ability to run a Debian Linux operating system.

Constructing this block was primarily composed of writing software in C++ capable of performing quick image processing and integrating a controller with a mechanical system. As such, techniques from ECE 373 (Software Intensive Engineering) and ECE 354 (Computer Systems Lab II) were used to build this block. Most of our knowledge and experience writing C++ code and running scripts on Linux-based systems was gained from our studies in ECE 373, and ECE 354 taught us all about embedded systems and specifically writing programs capable of simple image processing techniques. All of these skills were essential to the completion of this block of our project and have been extremely helpful.

TABLE III
CONTROLLER SYSTEM REQUIREMENTS

Specification	Value
Capture Distance	5 – 25 ft.
Capture Rate	>=5 frames per second
Processing Time	<=200ms

The purpose of the image processing in our system is to determine the position of the shooter on the basketball court. While we considered many different image processing techniques, we found that the most efficient method for our target detection is to have the shooter wear a jersey with a specific color pattern on it and have the camera look for that pattern. Thus, the image processing technique that we chose for our target detection is color filtering. The color filtering approach is exactly what it sounds like-the camera captures an image, and then the controller filters through the different color values in that image until it finds the value or range that it is looking for and then performs some action.

Our color filtering code is written in C++ and it is run on our BeagleBone Black’s Debian Linux 7 operating system using a startup script in the board’s auto run folder that allows the program to run on boot. We decided to use the OpenCV [7] and Video4Linux2 [8] libraries to complete our image processing. The V4L2 functions allow us to access the camera and its information from the BeagleBone, while the OpenCV functions provide us with data structures, methods, and API’s specifically for accessing the pixels of captured images and performing image processing.

Our color filtering approach looks for a jersey with horizontal red, blue, and green stripes. The decision to use this pattern primarily resulted from the effect of illumination on other more obscure colors. In different lightings and at different distances, colors that are more sensitive mixtures of red, blue, and green can have their pixel values changed so much that the image processing perceives them as different colors. However, clear reds, blues, and greens are almost always recognized by the program in all lightings. Thus, we decided to use these three colors in a horizontal striped pattern that is almost impossible to find naturally in a regular basketball environment. While it is technically possible to confuse the image processing by inserting a second instance of this pattern into a frame, our image processing code has never been tricked and has never picked up a background pattern thinking it was the player, and the chances of finding this pattern anywhere other than the jersey are extremely small.

The way our code finds the target in each frame is by using a process thread to go through the pixels of the image column by column looking for clear reds, blues, and greens (which is determined by extreme differences in respective RGB values of each pixel) and determining whether the frequency of each color is high enough in the column to constitute pattern detection. If the pattern is detected in enough consecutive columns, then the program calculates the center of the detected target and determines whether it is in the left, center, or right region of the image. If the target is in the left or right region, the code calculates the difference in pixels between the center of the image and the center of the target. Using this difference, a timed motor pulse is calculated by either using a linear function for targets closer to the center or a quadratic function for targets that are farther away. The signals to move in the corresponding direction are set high for that calculated amount of time to allow for the funnel to center on the target before making another move. The code processes only a fraction of the 720p resolution of the webcam as the fewer pixels allow for quicker processing and are enough to detect the target at distances as far as sixty-five feet away. Each iteration of this image processing thread takes under thirty milliseconds.

As stated in the imaging system, the camera sends images to the controller at a rate of thirty frames per second. While the image processing is quick enough for this rate, the OpenCV function to retrieve an image from the camera buffer takes longer. There is a five frame buffer on the Linux operating system and due to the slower image retrieval times, processing every frame in order would lead to a lag in response time by the system. To work around this issue, there is also a grabber thread running at all times that constantly grabs frames from the image buffer at all times to flush it out and ensure that any frames that the image processing thread processes are the most recent frames possible, preventing any unnecessary movements by the funnel. Even with this thread, error checking is included in the process thread to prevent reacting to a player's position more than once if an accidental lag occurs. The synchronization of the image processing and grabber threads is essential to the proper functionality of our system.

The result of our multi-threaded image processing code is a controller subsystem that quickly and accurately determines the basketball player's position on the court without interference from background and sends proper directional signals with the correct timing to the motor system in order to allow the overall system to track the player in real time.

D. *Block 3: Power System*

The power system component of the system must regulate our input voltage to power the motor and BeagleBone. On the high end, we estimated that the motor would draw 2A and the BeagleBone Black Controller would draw 1A for a total current draw of 3A. Also note that the 2A estimate for the motor is during motor startup. When DC motors start moving they instantaneously pull a large amount of current called "inrush current." Because the motor in our system constantly changes direction, it is crucial that our electronics can provide this amount of current on a fairly regular basis. We estimated the continuous running current of the motor to be closer to 1A, but designed our hardware around the 2A inrush current estimate. Thus, we used 3A (2A DC motor inrush current + 1A BeagleBone current) as our worst-case current draw estimate during the design of the power system.

TABLE IV
POWER SYSTEM REQUIREMENTS

Specification	Value
Motor Voltage	12V
BeagleBone Voltage	5V
Operation Time	>=1 hour
Current	>=3A

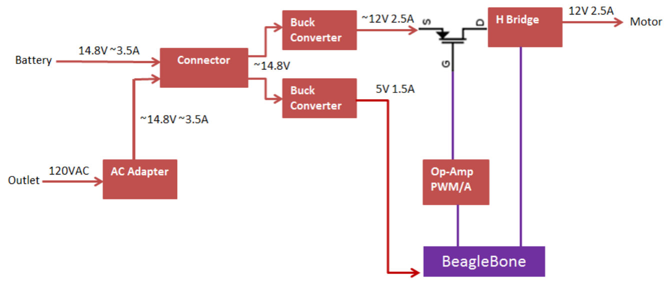


Figure G: Further Breakdown of Power Block

The block diagram for this subsystem is shown in Figure G. Included with our system are both a LIPO battery and an AC adapter so the system can be adjusted for maximum portability or maximum longevity respectively. With a battery, the user finds no restriction on where he can place the system but is limited by the battery's lifecycle. If the user has access to a power outlet then he can plug the system in using the AC adapter, in turn eliminating the dependence on the battery's lifecycle. Both power sources output 15VDC so we only needed to design circuit for a single input voltage.

The battery is a 14.8V lithium polymer (LIPO) rechargeable battery pack. We selected a LIPO battery because it has the highest energy density of rechargeable batteries on the market today [9]. This allows us to run the system for long periods without costing us much in terms of system weight. Moreover, even small LIPO batteries are rated to supply the necessary current of 3A to power the system for at least an hour. We selected a 5000mAh LIPO battery to ensure that we meet this requirement [10].

The AC adapter must have a 15V output and must be able to source at least 3A of current. We selected the AC adapter in [11] for use in our project.

We used efficient buck converter switching supplies for voltage regulation in order to maximize battery life. From our 15V input we had to regulate the voltage down to 12V and 5V to power the motor and BeagleBone respectively. In the subsequent discussion we split the circuit into two paths: the motor path and the BeagleBone path. We will look at the design of each path independently.

For the motor path, we designed the buck converter around the Texas Instruments TPS54331 [12]. This chip is a 3A, 28V input step down DC-DC converter. Using this chip, we designed our buck converter according to the design procedure laid out in the datasheet. We only had to make slight changes to the input and output capacitor values to achieve a working design.

Unfortunately, the buck converter suffered under high current loads. When the current-hungry DC motor was attached to the output, the chip would enter its over-current protection mode and shutdown. We were able to conclude that the high inrush current of the DC motor was overloading the device. When running the motor off a bench power supply we measured the startup current to be about 2.8A. Although the specifications of the TPS54331 deemed this acceptable, it was clear that we had to reduce this current. To do so, we inserted a current-limiting P-channel MOSFET as shown in Figure G. We initially planned to apply a ramp function to the gate of the FET so that the FET would slowly open up, thus limiting the initial current draw but ultimately not wasting much power since the FET turns on fully. We were able to design a simpler solution however. We experimentally discovered that, by setting the gate voltage to a certain constant value (about 6-7V), we were able to remedy the current inrush problem without having to ramp the gate voltage. In contrast to the originally measured 2.8A startup current, the startup current with the FET never exceeds 1.7A! This solved our problem while simultaneously conserving power. Because we are limiting the inrush current the motor starts up a little slower, though this is hardly noticeable and perfectly acceptable in this application.

In order to generate the desired software-adjustable gate voltage we needed an analog signal. Unfortunately, the BeagleBone does not offer any analog outputs. Thus, we designed a crude kind of DAC that takes a 0-3.3V PWM output from the BeagleBone, averages it via a low-pass filter to produce an analog voltage based on duty cycle, and adds the necessary gain with an op-amp network. By adjusting the duty cycle of the PWM signal, our circuit effectively produces an analog voltage that we can adjust between 0-15V. This is the “Op-Amp PWM/A” block in Figure G. The ideal value for the gate voltage was determined experimentally to be about 6.5V.

After the FET we inserted an H-Bridge circuit to control the motor direction. We used the Toshiba TB6549PG integrated Full-Bridge Driver chip [13]. This IC is capable of outputting 3.5A and has built-in digital control circuitry so we can control the motor direction with digital pins from the BeagleBone. It also has built-in shoot-through protection so that the H-Bridge never short-circuits. The pins of the DC motor are then hooked up directly to the output of the H-Bridge circuit. We designed the circuit according to the design procedure laid out in the datasheet. We added substantial output capacitance to reduce noise on the DC motor pins. We also found that we could further reduce the noisy motor pins by putting a small 1uF ceramic capacitor between the two motor pins.

We attempted to design the BeagleBone voltage regulator using the same TI DC-DC converter chip but were unsuccessful even after working with a TI applications engineer. We were able to design a functioning 15-5V regulator but the regulator was unable to source any current. Due to time constraints, we were forced to leave this as an unresolved problem and began to investigate other avenues.

We found an off-the-shelf solution in the form of a $4.30 1.5A, 7-28V input 5V output buck converter from Murata Solutions [14]. Not only was this an inexpensive solution, but it also helped reduce our PCB footprint and offered a high power efficiency of 95%. Thus, we deemed the purchase of this unit an acceptable solution given our time constraints and given that the unit itself was ideally suited for the application.

We prototyped our design on a breadboard and, once we verified that it was working, proceeded to translate the design to a PCB using the PCB Artist software. We designed the PCB as a cape for the BeagleBone Black so that we could stack the two boards and conserve space. We did a two-layer board where one layer serves as a ground plane. We ensured that the power traces were wide but minimized the width of signal traces to conserve space. We also ensured that the ground plane did not run beneath our inductor (see Figure H); this removes the possibility of eddy currents being produced in the ground plane as a result of the inductor coils. We were also careful to ensure that no ground loops were created in our design by connecting the respective grounds of the two major power paths only at the power source itself. This also helped ensure that the noisy motor path (due to motor inductance) does not interfere with the BeagleBone path. We ordered the board and components separately, and populated the board ourselves. Our initial test of the board went smoothly, all components were working as we expected, and the PCB implementation fixed some of the issues we were having with our breadboard version of the circuit.

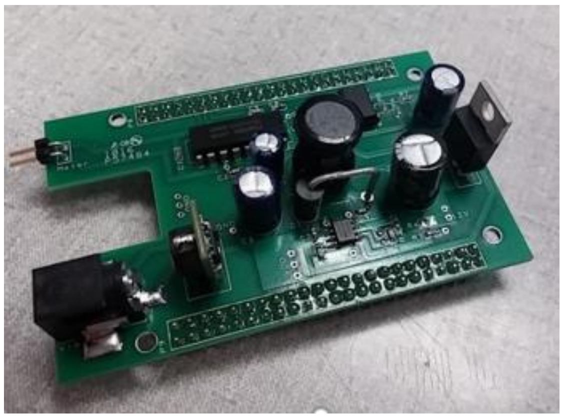


Figure H: PCB

We were very methodical in the design of this subsystem. We designed and tested each component individually before testing the subsystem as a whole. We did extensive power tests to ensure that our components were robust in the face of large current swings (as the motor turns on and off) and that every component was capable of sourcing the required current for extended periods of time.

We successfully achieved all of our goals and met all of our specifications for this subsystem. We exceeded our lifecycle requirement with a battery that lasts for at least 3 hours. At the end of the 3-hour testing period (demo day minus some system repair time), the battery still had ample charge left so we estimate that the lifecycle far exceeds 3 hours. We were able to achieve this with efficient buck converters that operate at 90% and 95% efficiency for the motor and BeagleBone paths respectively. We were able to minimize current draw with a current-limiting MOSFET. Our final measured worst-case current draw was 2.2A (1.7A DC motor inrush + 500mA BeagleBone at full CPU usage) and our average current was 1.2A (700mA DC motor steady state + 500mA BeagleBone at full CPU usage). These values are well within the estimated 3A worst-case current draw estimate that we designed around. We also verified that our design outputs a steady 12V for the motor and 5V for the BeagleBone with minimal ripple (the motor voltage does have some ripple, but this is to be expected from a DC motor).

The design of this subsystem was led by Adam and Derek. As the EEs in the group, they used their knowledge of electronics and buck converters from ECE 324 to design the circuitry around our selected ICs. We had to learn a lot about battery types, DC motor inrush current, buck converter chips, and H Bridges to design this block. We also learned about good PCB design practices while designing our PCB.

E. Block 4: Mechanical System

Although being the senior design project for an ECE program, this project relies heavily on the mechanical aspects in order for it to function properly. The mechanical system is composed of the funnel, the motor, and the gears. It also consists of mounting and casing designs.

One of the main focuses of the first semester was to design the gears used to rotate the funnel and determine which motor we would use to rotate the gears. The decision on the motor was led by Devon O'Rourke while the gear design was led by Adam Paranay

As Electrical and Computer Systems Engineers, we rarely work with motors during our undergraduate career. When we did, we didn’t have to choose one from scratch. We were given one as well as the specific function in which it would work correctly. In this project we needed to find a motor that would work with the requirements we set. This meant we needed to learn specific things about motors that we were not taught in our classes.

There are many different variations of motors that are sold commercially and we needed to calculate specific needs of the desired motor before shopping around. The requirements from Table V guided this decision significantly. Of the many characteristics of motors, the most important was the maximum RPM and the stall torque. We needed to make sure that our desired RPM and required torque matched well with that of the motor. Our desired RPM was set with the help of Computer Science Professor Rod Grupen in the Computer Science Department here at UMass. He suggested we use a RPM of around 50deg/s. We decided we would use a range that would be able to track the player quickly enough, seen in Table V. The torque needed to rotate the funnel was determined with the help of Francis Caron and the MIE Department. By using a force gauge from the MIE Department, Adam and Derek were able to measure the force needed to rotate the funnel while under a simulated weight. They measured ~8lbs. By multiplying this by the radius of Gear 1 we received our minimum torque needed to spin the funnel. With the help of mechanical engineering student Joe Howard and some research on gear properties, we decided upon a radius of 1in for our pinion gear, Gear 2 [15].

TABLE V
MECHANICAL SYSTEM REQUIREMENTS

Specification	Value
Weight	<15lbs
RPM	45deg/s – 55deg/s
Gear 1 diameter	13in
Gear 2 teeth	>=20

Using this radius and the minimum required teeth on a pinion gear, we chose 20 teeth for Gear 2. This gives us all of our knowns for the motor.

The next step of the process was to put all of our knowns in an Excel sheet, plug them into formulas (see formulas below), and generate the expected values needed for minimum stall torque and maximum RPM (without load). The main formulas we used took in stall torque and maximum RPM without load and outputted a RPM with our estimated load. We used stall torques and RPMs of motors from a well-known motor hobbyist website [16]. After thorough discussion as a team, we decided upon a 60RPM HD Planetary Gear Motor because it offered the highest stall torque with the closet RPM to what we wanted when under load. This meant we could run the motor at its nominal voltage of 12V.

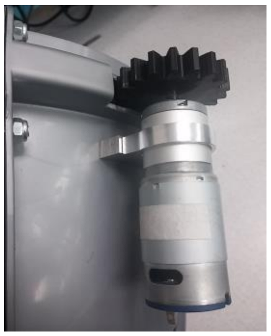


Figure I: Motor and Pinion Gear Mounted to SKLZ Unit

Formulas

RPM_a * Teeth_a = RPM_b * Teeth_b

RPM_b = (RPM_a) * (Teeth_a/Teeth_b)

Torque Ratio → (Teeth_a/Teeth_b) = (RPM_a/RPM_b)

We needed to design a mechanical system that could rotate the outer shell of the SKLZ system around its inner ring that attaches to the rim of a basketball hoop. Being a team of electrical and computer systems engineers, we didn’t have too much experience with designing gears or mounting systems, and we couldn’t look to any of our course material for guidance either. Still, we were confident in our ability to come up with a solution to our problem. Our initial idea of creating a system of two gears, driven by the smaller motor mounted pinion gear, was what we decided to implement in our project. We knew it was important to look to other possible approaches before moving ahead with an idea, so we considered several other ideas before moving ahead with the gear design.

We decided that a gear system would be perfect for our project because it would be simple implement and very reliable. We replaced the inner ring that came with the system (that the straps attach to) with a near identical ring with gear teeth along its outside edge. In the back of the SKLZ system, we cut a two-inch window in the ring track where the small gear (attached to the motor shaft) would interface with the redesigned inner ring gear. Both of our gears would be designed by us and 3D printed here on campus, since you can’t find a 13-inch diameter ring gear online for purchase. The motor is mounted to the SKLZ system underneath the gear window to allow the two gears to mesh fully. Using one of the motor mounts we purchased along with our motor, we created a mounting bracket and attached the motor and pinion gear to the system.

The actual design of the gears was done on Autodesk Inventor by Adam. After learning the basics of the program, we learned how to use the Inventor gear creation tool. Using this tool, you can specify several known parameters and have the software calculate the last unknown. We were able to input the number of teeth, gear ratio, and center distance (which indirectly specified the diameter of each gear), and have the creation tool calculate the module of each gear (the tooth size and pitch).

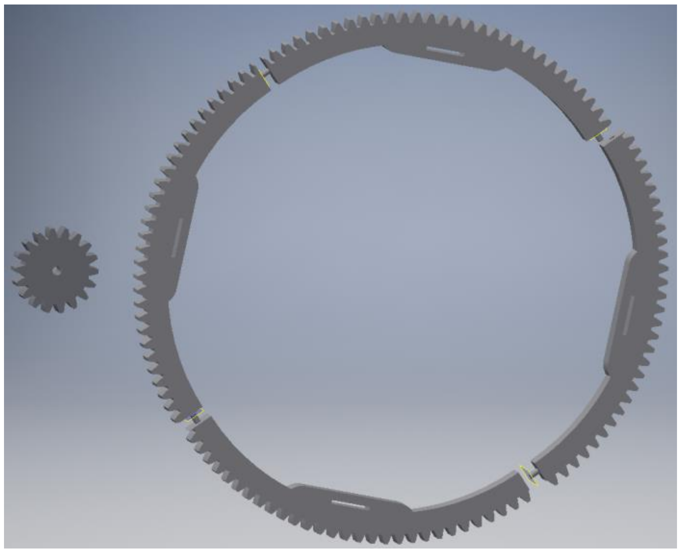


Figure J: 3D Model of Pinion Gear and Inner Gear

After creating each gear using the tool, it was important to check that the gears would mesh properly before moving ahead with further modifications and eventually printing. By using physical constraints, we were able to constrain faces, axis, and planes of the two gears to achieve rotation within the program. By doing this, we verified that the gears mated well and prevented the need for additional 3D printing in the future due to poorly designed gears. We then moved ahead and created a D shaped motor shaft hole in the two-inch diameter pinion gear. Also, we made the large gear into a ring by removing the center area of the gear. We were then able to split the large gear into four quarters, since the singular gear was too big to 3D print as one piece. We added connection pegs and holes to the faces of the split gear so they would be easy to piece together and then acetone to make a solid finished product. Finally, we created the strap mounts and added one to each quarter of the gear ring so the system could be hung and tested.

We were skeptical of the strength of the 3D printed gears and figured that there may be problems with our first print anyways. Therefore, we were expecting to have to print another round of gears. We were extremely happy to find that the finished product, after using acetone to melt the four quarters together into one large gear, was near perfect! The gears meshed very smoothly and there was no way we were going to have a gear tooth break off. In addition, the pinion gear fit onto the motor shaft well, and we secured it with Guerrilla glue to make sure the metal motor shaft would not eat into the plastic gear. After completing the gears and getting the motor/pinion gear mounted to the body of the SKLZ system, we were eager to test it. We used a ladder to hang the system with completed mechanical components, and powered the system with a ±12V powered breadboard. By adding a switch, we could test the effect of rapidly changing directions by switching between +12V and -12V. All in all, we found that the design we implemented had no problems rotating. We tested the system under normal operation, under a simulated load (by adding weight where we are planning on mounting the battery, camera, etc.), and under the effect of a basketball being shot at and through the system. The integrity of the gears has been maintained throughout most of the project. One of the sections that we melted together with acetone broke but was fixed promptly. In the future, we would ideally 3D print a gear in full in order to avoid putting too much pressure on the connection points. Alternatively, we could add more straps to the system to take some of the pressure off of the four straps we have now.

The mounting of the components on the system consisted of 3D printed designs and parts we were able to gather from M5. Using Autodesk Inventor, we designed all of our 3D prints and printed them in M5.

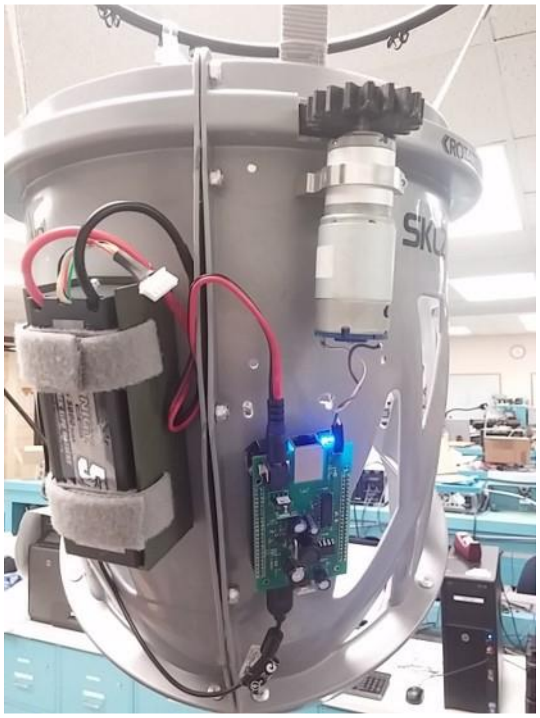


Figure K: Mounting without 3D casings

The battery and AC adapter used the same 3D housing. We designed a small module that would fit both the adapter and the battery. The components were secured to the module using Velcro straps. The PCB was as a BeagleBone Black cape, that is, it form-fit the BeagleBone and connected directly into the headers. The pair were mounted directly to the funnel and then covered in a 3D-printed cover. The motor was also mounted to the funnel and enclosed with a 3D-printed cover. Lastly, the webcam was mounted to the front of the funnel.

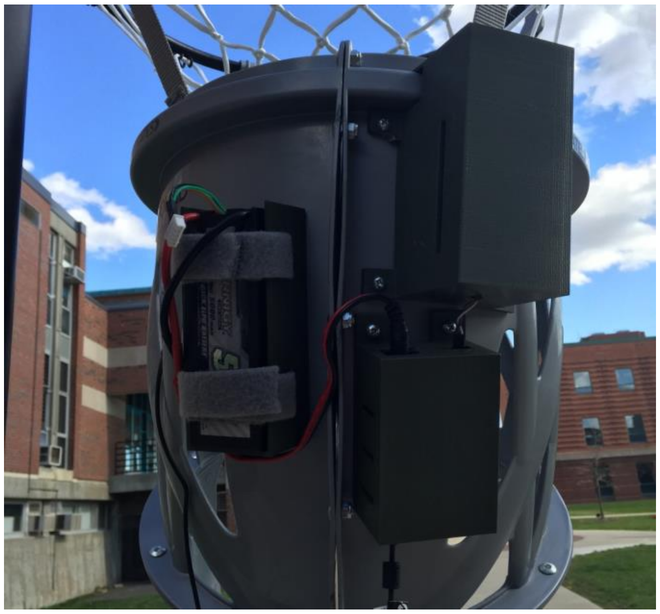


Figure L: Mounting with 3D casings attached

We cut out a small portion of the front so that the 3D-printed enclosure we designed would fit flush with the mold of the funnel. We found the angle that the webcam needed to be by measuring with a smartphone level application and used that to provide the arc needed in the webcam enclosure. We also put some soft foam under the webcam enclosure to absorb some of the impact if hit with the ball. We needed to make sure that the webcam could survive the impact of the ball if hit and this case allows the webcam to be unaffected by the ball on a direct hit while keeping the vision of the camera uninhibited. We are very happy with the 3D-printed cases. They provide much needed protection and we were able to prototype them very quickly.

III. Project Management

As a team, we were able to meet all of our goals throughout our time working on the project. Early on we created a Gantt chart that would be a guide and a resource for when we needed to have certain goals met. Following this diligently the whole time kept us on track to complete the project. For Cumulative Design Review, we delivered three things: full integration of the webcam, BeagleBone Black, and the motor, a completed power system breadboard design, and a mounting design for the integrated sub-units. For the Final Product Review, we delivered three things: All components mounted to the system, a printed and working PCB, and a fully functional BRO system.

Team 10 worked well together. We constantly met the deadlines we established for ourselves as well as the deadlines of our advisor/faculty evaluators. Derek Foster and Adam Paranay are EEs while Brian Acker and Devon O'Rourke are CSEs. This gives us a solid background in both hardware and software. Each of us has worked in a professional engineering setting giving us a leg up in terms of working responsibly within a team.

We worked to set goals for the semester early on. After setting the goals and assigning leads for each one, we individually worked on our parts. While we achieved all the goals we set for ourselves, the roles of each member in each sub-unit changed slightly from the projected roles assigned. We knew that this project would require adaptation and a dynamic outlook. Everyone helped out with every aspect in the project. The communication we had as a team contributed heavily to our success. Our weekly meetings allowed us to check in with each other's progress and provided an avenue to help each other and offer insight on every part of the project.

IV. Conclusion

We are happy to report that the BRO meets all the specifications we set for ourselves in September of 2015 and it functions just as well. Not only did we meet all of our specifications, we exceeded all expectations. This last half of the project focused mainly on system integrations and improvements on the image processing algorithm. We had a working color filtering algorithm for MDR as well as a power system design. In the time between MDR and CDR, we managed to integrate the mechanical system with the color filtering algorithm, design the power system on a breadboard, and design a mounting plan. After CDR, the focus was to print the PCB, fully integrate the system, and perfect the image processing. As a team, we worked very well together from day one. We were always very communicative and our cohesion yielded great results. Come FPR, we were able to fully demo our project to our evaluators and they were impressed with our demo. SDP Demo Day went very well. Our project hit a few snags when one of the straps broke and a solder point detached but we quickly fixed these issues and continued with the demo. Overall, we were very happy with how the BRO turned out. We had a successful project and enjoyed the time spent learning as well as developing as individuals and as a team.

Specification	Goal	Actual
Weight	< 15lbs	< 10lbs
Range	5-25ft	8-65ft
Capture Rate	>=5 frames/sec	>5 frames/sec
Processing Time	<=200ms	Avg. <200ms
Battery Life	>1 hr.	>3 hr.
Motor Voltage	12V	12V
BeagleBone Voltage	5V	5V
Current Draw	<=3A	~1A
RPM	45deg/s – 55deg/s	~50deg/s

Figure M: Specifications for the entire project

Acknowledgment

We want to thank Professor Hossein Pishro-Nik for his tireless support and the professional and personal insight he has provided our team. Acknowledgements also go out to: Francis Caron, Walmart, Jeremy Landry, Joe Howard, Professors Kelly, Holcomb, Duarte, Tessier, Soules, Hollot and Grupen.

References

[1] 2016. [Online]. Available: [1] https://drdishbasketball.com/ic3/. [Accessed: 19- Jan- 2016].

[2] Amazon, 2016. [Online]. Available: http://www.amazon.com/Goalrilla-B2608W-Goaliath-Basketball-Return/dp/B002UPWBUK/ref=sr_1_34?ie=UTF8&qid=1452970711&sr=8-34&keywords=goalrilla. [Accessed: 19- Jan- 2016].

[3] D. Rebel, "Basketball Shooting Machine | Dr. Dish Rebel", *Dr. Dish*, 2016. [Online]. Available: https://drdishbasketball.com/product/dr-dish-rebel/. [Accessed: 20- Jan- 2016].

[4] Amazon.com, "Amazon.com : SKLZ Shoot Around 180° Ball Return : Sports & Outdoors", 2016. [Online]. Available: http://www.amazon.com/SKLZ-Shoot-Around-180%C2%B0-Return/dp/B013B8ICNY/ref=sr_1_78?srs=2600676011&ie=UTF8&qid=1453175005&sr=8-78&keywords=basketball. [Accessed: 19- Jan- 2016].

[5] Sparkfun.com, "Beaglebone Black - Rev C - DEV-12857 - SparkFun Electronics", 2016. [Online]. Available: https://www.sparkfun.com/products/12857. [Accessed: 19- Jan- 2016].

[6] Amazon.com, "Amazon.com: Logitech HD Webcam C270, 720p Widescreen Video Calling and Recording: Computers & Accessories", 2016. [Online]. Available: http://www.amazon.com/Logitech-Webcam-Widescreen-Calling-Recording/dp/B004FHO5Y6. [Accessed: 20- Jan- 2016].

[7] Opencv.org, "OpenCV | OpenCV", 2016. [Online]. Available: http://opencv.org/. [Accessed: 19- Jan- 2016].

[8] Wikipedia, "Video4Linux", 2016. [Online]. Available: https://en.wikipedia.org/wiki/Video4Linux. [Accessed: 19- Jan- 2016].

[9] L. Keith Arauo - Epec, 'Primary and Rechargeable Battery Chemistries with Energy Density', *Epectec.com*, 2015. [Online]. Available: http://www.epectec.com/batteries/chemistry/. [Accessed: 11- Dec- 2015].

[10] Turnigy. (2016) "*Turnigy 5000mAh 4S1P 14.8v 20C Hardcase Pack*" [Online]. Available http://www.hobbyking.com/hobbyking/store/__15521_Turnigy_5000mAh_4S1P_14_8v_20C_Hardcase_Pack.html. [Accessed:1-May-2016].

[11] Amazon. (2016). "*Universal AC Adapter 15V 16V 18V 18.5V 19V 19.5V 20V 22V 24V 70W*" [Online]. Available: http://www.amazon.com/Universal-Adapter-15V-18-5V-19-5V/dp/B004I5ERUW. [Accessed:1-May-2016].

[12] Texas Instruments. (2014). "*TPS54331 3-A, 28-V Input, Step Down DC-DC Converter With Eco-mode™*" [Online]. Available: http://www.ti.com/lit/ds/symlink/tps54331.pdf. [Accessed:1-May-2016].

[13] Toshiba. (2010). "*TB6549FG, TB6549PG,TB6549HQ Full-Bridge Driver IC for DC Motors*" [Online]. Available: http://www.mouser.com/ds/2/408/toshiba%20america%20electronic%20components, %20inc._dst_tb-552224.pdf. [Accessed:1-May-2016].

[14] Murata Solutions. (2014). "*OKI-78SR Series Fixed Output 1.5 Amp SIP DC/DC Converter*" [Online]. Available: http://www.mouser.com/ds/2/281/oki-78sr-56393.pdf. [Accessed:1-May-2016].

[15] Understanding Gear Ratios, 2016. [Online]. Available: https://www.teamassociated.com/pdf/cars_and_trucks/shared/gear_ratios.pdf. [Accessed: 20- Jan- 2016].

[16] Servocity.com, "60 RPM HD Precision Planetary Gear Motor", 2016. [Online]. Available: https://www.servocity.com/html/60_rpm_hd_precision_planetary_.html. [Accessed: 20- Jan- 2016].

Figures

[A] I.ytimg.com, 2016. [Online]. Available: https://i.ytimg.com/vi/QuUIJmdrvsM/maxresdefault.jpg. [Accessed: 19- Jan- 2016].

[B]Amazon.com, "Amazon.com : SKLZ Shoot Around 180Â° Ball Return : Sports & Outdoors", 2016. [Online]. Available: http://www.amazon.com/SKLZ-Shoot-Around-180%C2%B0-Return/dp/B013B8ICNY/ref=sr_1_78?srs=2600676011&ie=UTF8&qid=1453175005&sr=8-78&keywords=basketball. [Accessed: 19- Jan- 2016].