Rochester Institute of Technology

Principle of Robotics Project – Fall 2017

Metroid

Dual Hexapod Ball

Mission Statement

The goal of this project was to build from start to finish a fully functioning dual hexapod robot, capable of turning into a ball and potentially rolling. The problem seen with conventional hexapods is their limited ranges of motion and its inability to move once tipped over. However, with a dual hexapod design the robot can now fall and keep moving on either end. It can protect itself from danger by turning into a ball, or moving more easily over objects while in ball form. A combination of acrylic and 3D printed parts were used to keep this build cheap and quick. A Raspberry Pi was used as the main processing unit. Two SSC32's were used to talk to 18 servos each. Both hexapods received servo positioning commands from the Raspberry Pi which when chained together created various walking and rolling motions. To create even more precise rolling motions the BNO055 9dof IMU sensor was used to relay angular data to the Raspberry Pi to know exactly which leg it should kick up to create maximum push when rolling.

People

FACULTY ADVISOR

Dr. Ferat Sahin

STUDENT

Felix Blanco, Xavier Tarr, Rodney Sanchez

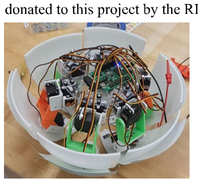
SPECIAL THANKS

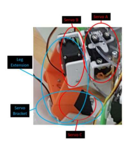
RIT MOSAIC Center

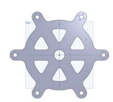
Hardware

Metroid – Mechanical Build

36 servos 4 acrylic plates and more than 50 3D printed parts were used to create Metroid. 4 laser cut acrylic base's were used as the center pieces of Metroid. 6 "shoulder" servos were placed on each pair of bases which created one hexapod. Each shoulder servo was attached to the base as well as to the "elbow" servo using 2 3D printed servo holders. Attached to the elbow joint were 2 more 3D printed pieces used to create a length which allowed Metroid's hexapods to create length between the ball and tripod forms. These two 3D printed lengths are more easily explained as a "forearm". The forearm then connected to the final wrist servo which was screwed onto each of 12 shells that when all placed together gave Metroid its Spherical look. The shells were hand cut from a 12 inch globe donated to this project by the RIT MOSAIC Center.







Metroid – Raspberry Pi

The Raspberry Pi is used to create and send both SSC32's where they should send each servo during each step of the current walking gate. The Raspberry Pi creates each walking gate and gathers data from the IMU sensor to computer which leg should be pushed out next during a roll.

Metroid – SSC32

This board was used as the servo driver for each hexapod of Metroid. It received UART commands as text from the raspberry pi to place each servo in its desired position.

Metroid – BNO055

This board was used to gather orientation, Gyroscope data, and accelerometer data. All relayed back to the raspberry pi for further processing to help determine Metroid's next motion.

Project Metroid

Software

Python Code

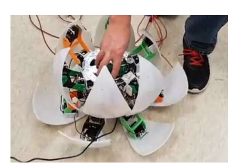
Three main controlling codes were created to allow Metroid to move. A custom class called SSC32.py was created to control both halves of Metroid (Both top and bottom hexapods). From here multiple hexapod gates were created including a dual hexapod tripod walk, single hexapod walk, single leg ripple roll gate, and tripod wave rolling gate. Finally, a custom class for the BNO055 IMU sensor was created to interface with gyroscope data, accelerometer data, and gather absolute orientation of the sensor.

Tripod Walk

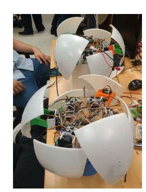
The single tripod gate was done by creating 6 arrays of servos. One array that encapsulated all the first tripods elbow. The next encapsulated all the first tripods shoulders, and the last all the first tripods wrists. The same three arrays were then created using the seconds tripods specific servos. Values were written to each tripods elbow, wrist, and shoulder servos. The motions defined in table 1 were then placed into a while loop and done as many times as the user would like. One iteration of table 1 would perform 1 tripod walk cycle.

Tripod Being Moved	Servo Set being Moved	New Position
Tripod 1	(Elbows/Shoulders)	(Up/Forward)
Tripod 1	Elbows	Down
Tripod 1 - Tripod 2	Shoulders - (Elbows/Shoulders)	Back - (Up/Forward)
Tripod 2	Elbows	Down
Tripod 2	Shoulders	Back
Loop back to the top however no delay between last motion and first motion.		

Table 1: Tripod Walk Break Down



Tripod Walk



Dual Tripod Walk

Software Continued

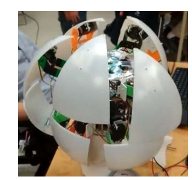
Single Legged Wave Roll

The rolling script used both hexapods simultaneously to create more surface area for the robot to roll on. For the wave rolling algorithm one important piece of information had to be realized. For maximum push not only does the elbow have to pull out but the wrist has to stay parallel to the floor as much as possible to maintain lots of surface area. This was finally done in our code and a table for the simple one legged ripple roll can be seen below as Table 2.

Legs Being Moved	Servo Set being Moved	New Position
Hex 1 Leg 1 - Hex 2 Leg 1	(Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out)
Hex 1 Leg 2 - Hex 2 Leg 2 Hex 1 Leg 1 - Hex 2 Leg 1	(Elbow/Wrist) - (Elbow/Wrist) (Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out) (Down/In) - (Down/In)
Hex 1 Leg 3 - Hex 2 Leg 3 Hex 1 Leg 2 - Hex 2 Leg 2	(Elbow/Wrist) - (Elbow/Wrist) (Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out) (Down/In) - (Down/In)
Hex 1 Leg 4 - Hex 2 Leg 4 Hex 1 Leg 3 - Hex 2 Leg 3	(Elbow/Wrist) - (Elbow/Wrist) (Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out) (Down/In) - (Down/In)
Hex 1 Leg 5 - Hex 2 Leg 5 Hex 1 Leg 4 - Hex 2 Leg 4	(Elbow/Wrist) - (Elbow/Wrist) (Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out) (Down/In) - (Down/In)
Hex 1 Leg 6 - Hex 2 Leg 6 Hex 1 Leg 5 - Hex 2 Leg 5	(Elbow/Wrist) - (Elbow/Wrist) (Elbow/Wrist) - (Elbow/Wrist)	(Up/Out) - (Up/Out) (Down/In) - (Down/In)
Hex 1 Leg 6 - Hex 2 Leg 6	(Elbow/Wrist) - (Elbow/Wrist)	(Down/In) - (Down/In)
Loop back to the top however no delay between last motion and first motion.		

Table 2: Single Leg Wave Roll

Single Leg Roll



Tripod Roll

Results

All motion gates were created and implemented on Metroid successfully. Communication between the SSC32's and the Raspberry Pi was flawless. However, it was found after countless tests that our prediction of making most of Metroid's joints out of 3D printed material would be an issue. After about 8 rigorous tests involving trying to roll and performing full tripod walks, the 3D printed servo holders began to crack. Due to time constraints and lack of funding it was beneficial to make these key components out of 3D printed material. From the results seen after this experiment it is extremely important for these parts to be metal so that the required weight of Metroid can be adequately supported.

It was also found that the servos used could not provide the torque required to actually get Metroid rolling. This can easily be mitigated by replacing the current servos with metal geared servos to provide the required torque. However, the current servos were once again chosen due to funding constraints.

Conclusion

From this project an extremely flexible robot was created. Some feature upgrades that will be done will definitely include stronger servo holders and stronger linkage parts. This will make Metroid extremely rugged. Servos will be updated to allow for higher torques to really get Metroid rolling.

Even though Metroid was unable to perform all initial tasks due to physical constraints, it was an extremely rewarding experience to bring our idea to completion.