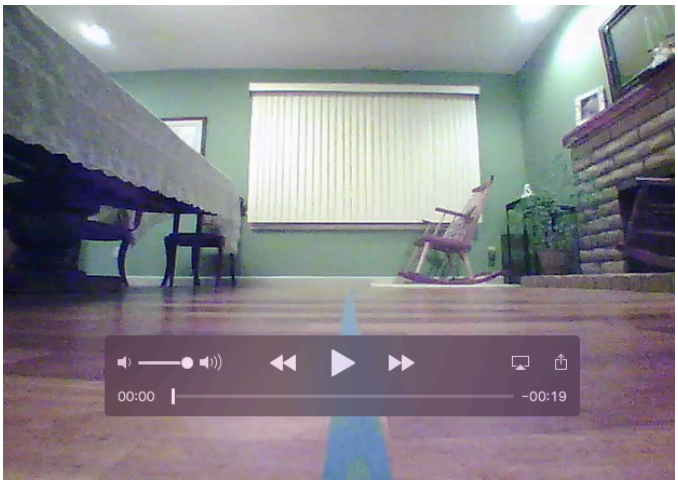
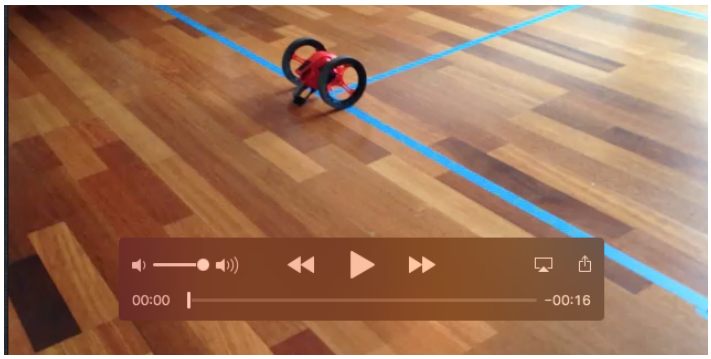
Coding the Parrot Jumping Night Drone



Drone Internal Camera

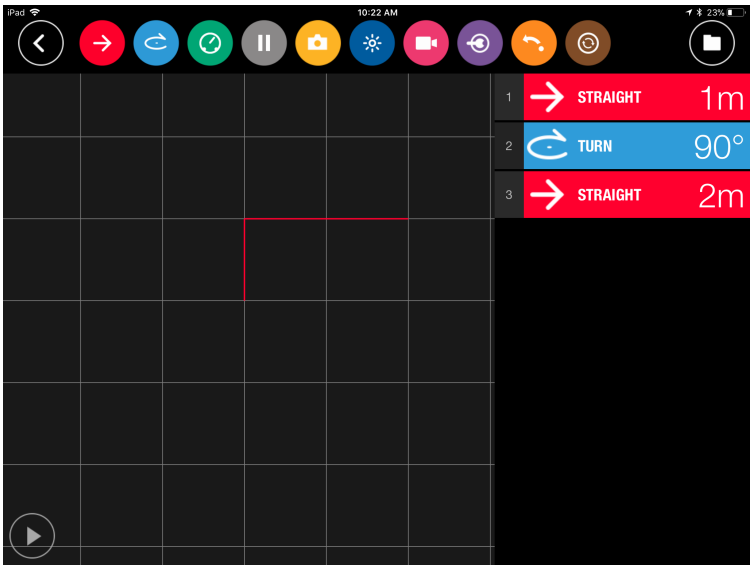


External view of drone

Over the last 2 weeks Rheannon and I have been coding a Parrot Jumping Night Drone. These drones have been discontinued and the remaining stock is being sold at a significant discount on Amazon ($37 - $50). The drone is controlled by code written on an iPad via a WiFi connection. The drone-to-iPad WiFi connection works but with some effort to make the handshake. Sometimes, I would have to turn the drone off, turn iPad WiFi off, and then turn both on to make a connection. Another issue was that the Parrot drone as received would not run until a firmware update was installed. The drone will connect to a computer using a USB cable and mounts as an external drive. This enables installation of the firmware. It took considerable time hunting on the Parrot website to locate the firmware update. It was not obvious where it was. After several Google searches I was finally successful in identifying the link to the firmware.

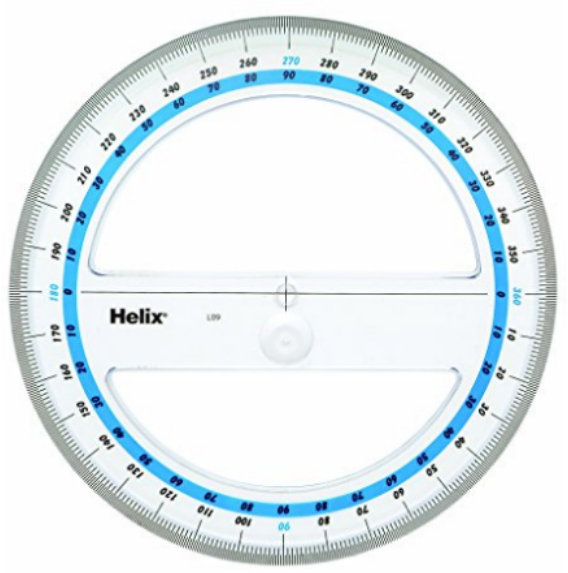
Parrot provides a proprietary app to control and program the drone. It works but is buggy. Many times we would have to delete the code, close the app and start over. So, I'm hesitant in recommending this drone. However, it has turned into a reasonable learning experience for Rheannon.

The Parrot App UI displays 3 windows. The main central window displays a square grid with 1m edge lengths. The top window contains code icons which are dragged-and-dropped into the right hand window to build the code. The



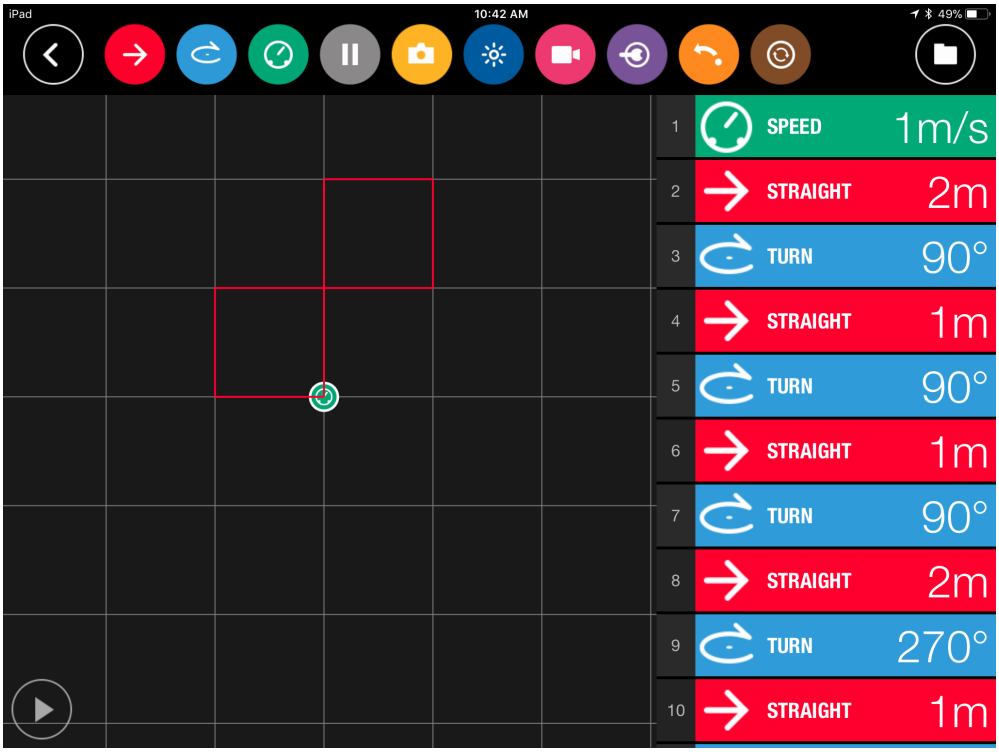
drone motion defined by the code is displayed in red on the grid. This code directs the drone to move forward 1m, turn right, and move forward 2m. The drone motion is very accurate (accepting some slippage due to lack of friction between tires and wood floor) in hitting node points as shown in the above movies.

Because the drone motion was precise, I turned this into a lesson on grids, linear motion and angular motion. The first step was moving the dining room furniture to the wall and laying out a 1m x 1m square grid on the floor with tape. Next was buying a meter stick (remember I'm in the US and we have yard sticks) and a 360 degree protractor. The turn icon only allows clockwise rotation so a 90 degree left turn is defined as a clockwise 270 degree rotation.



Following is a list of activities we did.

1. The meter stick is graduated in cm and mm. Rhea was impressed that the last number was 1000. This is a big number to her. She also liked that she could count by 10s to get to 100 and then by 100s to get to 1000.
2. We used the meter stick to lay out the grid. I showed her that 500 was half the length. We added marks on the tape to indicate this location.
3. We played human robot. I gave her verbal directions, such as move forward on the grid 1m. Look at the protractor in your hand and turn clockwise 90 degrees. Then it was my turn to be the robot and she gave me voice commands to move on the grid.
4. We wrote code to direct the drone's motion.
5. We experimented with fractional distances (e.g., move forward 0.5m).
6. We experimented with an angular turn of 45 degrees and noted that if the drone traveled 1m it didn't end up at the grid point. I showed her we were traveling on the hypotenuse of a triangle and the distance is more than the grid edge length and can be calculated. Entering a travel distance of 1.4 gets the drone to the grid point.



Using the Parrot app, this is the code to direct the drone's motion shown in the movies. Note that commands 11 to 13 to bring the drone back to the starting position are scrolled off the display. Loops are not available. Also, note the command "turn 270" to initiate a left hand turn.

Using the Tickle app, this is the code to direct the drone's motion shown in the movies. Two loops were used to decrease the total number of repetitive code needed by the Parrot app. The "turn" command allowed both a right and left turn as an inline menu option. While the Parrot app allowed specifying a translational distance, the Tickle app required entering a speed and travel time. Several experimental runs were required to zero in on the correct (% speed, time) data pairs to travel 1m. However, once determined, the motion was consistent. The drone traveled 1m at 30% speed for 1.4sec. The drone traveled 2m by doubling the time to 2.8 sec.