Recommended Functions

ZERO ROBOTICS
SPHERES ISS CHALLENGE

Writing Functions

- If you haven't already, walk through the tutorial on writing your own functions. Writing functions makes for organized and easily-editable code, and is a skill all programmers should have.
- The ZR API is loaded with functions for your use, but there are some functions we want you to write yourself. We've made a list of functions you may want to add to your algorithms.
- All the functions in this tutorial are suggestions; you aren't required to write any of them, and we encourage you to write functions that aren't on this list.

ZERO ROBOTICS
SPHERES ISS CHALLENGE

Writing Functions

- There are many approaches to writing each function. The best solutions will be the most creative and precise. Collaborate with your team members.
- Test them, test them, and test them again. If your functions aren't reliable in every situation, things could go disastrously.
- Conditionals are your best friend. Use conditionals in loop() to decide when to call certain functions, and use conditionals within your functions to account for every possible scenario.

distanceBetween

- You have written the code for this before – it is very simple.
- Make distance an integral part of your code. At all times, keep track of your distance to certain items, your opponent's distance to certain items, and your opponent's state in general.
- You don't know how fast your opponent can move, so use distance as a conditional in your overall strategy to determine who has the upper hand.

moveFast

- setPositionTarget is reliable but ultimately inefficient. Use setVelocityTarget and/or setForces along with setPositionTarget to create a function that gets you to a target position as fast as possible, every time.
- Watch out for the second scenario in the setVelocityTarget tutorial. Having to narrow in on your target after several passes will end up wasting time.

isStopped

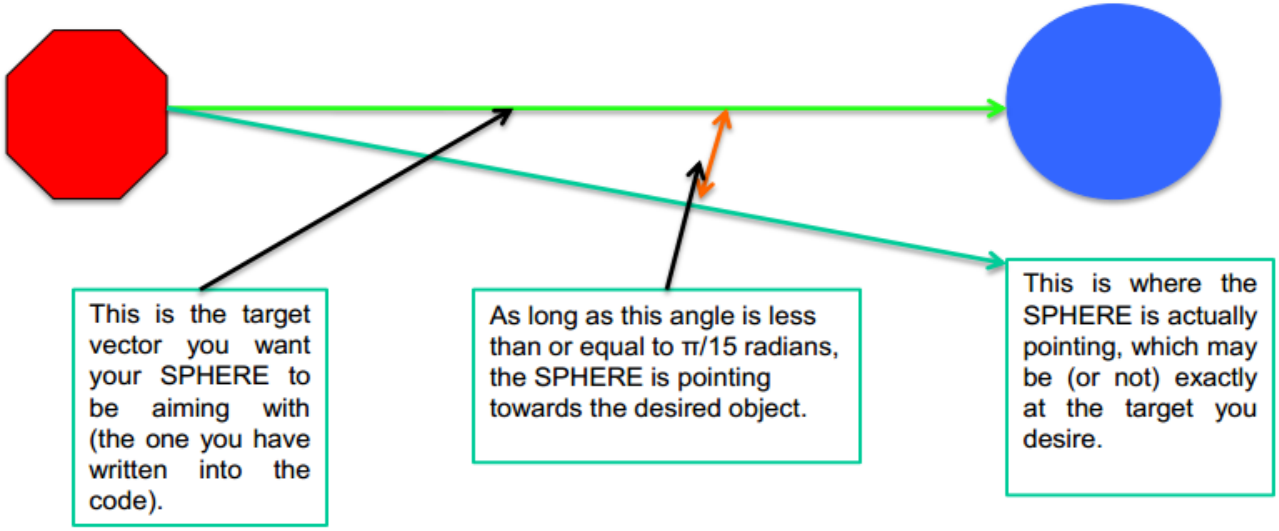
- Write a function that returns a boolean to determine whether or not your satellite is moving. Account for all types of motion (translational and rotational).

hasReachedTargetPos

- Because SPHERES drift, your current position may not exactly match your target position after you've reached a waypoint. If you use a simple == operator, your conditional will return false if you are even a hundredth of a meter off of your target position.
- Determine a range of values around a target position at which you would consider the satellite to have reached the target.

hasReachedTargetAtt

- You'll want to write a similar function for pointing. Zero Robotics allows some leeway in attitude, often a π/15 radian tolerance.
- Use mathVecInner to find the dot product of your attitude vector and the target attitude vector.



spin

- Having a solid spin algorithm will boost you up the leaderboard.
- Use the functions in the API with your math and physics skills. Controlling wobble while spinning is very difficult, but it can be done.
- You may want one spin function with several parameters or several independent spin functions to account for the desired axis of rotation or target angular velocity.

revolve

- Like spinning, a sound algorithm for revolving will be a huge advantage in this competition.
- Test this function extensively. Closely monitor your state and use the debug function for simulations. Use conditionals to ensure that you can steer your satellite back to the right path if something goes awry.
- Remember to aim for uniform circular motion.

avoidCollision

- Collision avoidance prevents you from crashing into your opponent's satellite, but has a penalty. Also, it's very hard to regain speed and get back on the right path after a near-collision.
- You'll be better off if you write a function to prevent collision avoidance from ever kicking in. Keep track of myState and otherState, and give your satellite enough time and space to maneuver out of the way.

ZERO ROBOTICS
SPHERES ISS CHALLENGE

Game-specific Functions

- The functions we've listed apply to all Zero Robotics competitions. You should also write functions specific to each year's game.
- For instance, if the game involves navigating through debris, write a function that finds the fastest route.
- Writing functions will make your code more versatile and efficient – two of the core goals of programming. Ensure accuracy, but never hesitate to take an unconventional approach. The most creative functions are often the most successful.