ZERO ROBOTICS
HS TOURNAMENT 2014

Zero Robotics ISS Programming Challenge High-School Tournament 2014:

Conducting Optical Research on Nearby Asteroids (CORONA)

GAME MANUAL V2.2

2014-Sep-06

To: Zero Robotics Teams
Re: CORONA program

Attention to all teams:

NASA has observed a recent spike in the number of asteroids knocked loose from the asteroid belt and into close proximity to Earth. Using highly sophisticated algorithms, they have used the mass of these space rocks to predict the composition of these asteroids with great accuracy. However, there exists amongst these rocks a small percentage that are of comparable mass percentages but do not fit the model. Eager geologists believe scientists may have discovered a new element and immediately rally for further study.

NASA calls upon its fellow scientists at MIT for a plan of action. With the help of these researchers, the joint CORONA program was born. CORONA (Conducting Optical Research On Nearby Asteroids) requires the use of MIT's most recent project, SPHERES, to capture visuals of the closest asteroids to Earth's atmosphere.

MIT researchers are fairly confident that they have stumbled upon something novel in these asteroids. However, some believe that there may actually be ice on these rocks and not a new element at all. The contrasting opinions lead to the creation of two separate SPHERES teams. Both teams will be using their respective satellites to take up-close photos of the asteroids at specific points of interest, as determined by NASA.

Soon after launch, however, NASA catches sight of incoming solar flares at the exact location where the SPHERES are intending to intercept the asteroids. The satellites risk imminent mechanical issues and loss of their stored photos if they impact with the solar flare while turned on. The MIT teams have to decide whether they want to use the shadow zone of the asteroid for protection or temporarily power down when a solar flare is near.

The choice is a tough one, but one that must be made. Each team is counting on its satellite to find visual proof of why either water or a new element exists on these space rocks. Proof of either would be invaluable, but conclusive evidence is essential.

As a SPHERES expert, your skills will be in high demand. GOOD LUCK!

[illegible]

Alvar Saenz-Otero

MIT SPHERES Lead Scientist

Contents

1 Game Overview

Matches will be played between two SPHERES satellites, controlled by code written by two different teams/alliances. Satellites start the game facing the asteroid. Each team will attempt to take pictures of special points of interest (POI). These points of interest change location once every minute. The SPHERES can hold up to two pictures in its memory at a time. During the game, teams have the option of picking up a memory upgrade pack which allows players to store extra pictures before needing to upload. There are two zones around the asteroid which determine the amount of points each picture is worth. Points are received only after uploading the pictures stored in each satellite's memory. Throughout the game, there are solar flares that can corrupt the pictures in memory, and if precautions are not taken, can also damage the satellite. Satellites are completely safe only in the shadow zone.

1.1 *Game Layout*

The Zero Robotics High School Tournament 2014 begins with competitions in simulation. The competition mimics the operational volume available aboard the International Space Station, where the ISS finals will be conducted in January 2015. For the 2D game, the arena is a plane with only X and Y cardinal dimensions. For the 3D game, the arena includes X, Y and Z components. The game arena encompasses the complete area where the SPHERES satellites can operate as shown in Figure 1. However, the game is played in a smaller area called the Interaction Zone. If players leave the Interaction Zone, they may still be within the arena operational area, but they will be considered out of bounds.

The dimensions of the *Interaction Zone* are:

Table 1 Interaction Zone Dimensions

	2D	3D	Alliance
X [m]	[-0.64 : +0.64]	[-0.64 : +0.64]	[TBD]
Y [m]	[-0.80 : +0.80]	[-0.80 : +0.80]	[TBD]
Z[m]	0.0	[-0.64 : +0.64]	[TBD]

Within the interaction zone there are three zones: Danger, Inner, and Outer zones.

When satellites enter the danger zone thrusters are automatically turned on to move them out of the danger zone. Inner and Outer zones only affect the amount of points a picture is worth.

The Shadow zone represents the area safe from Solar Flares.

Table 2 Zone Radii Positions

	2D	3D	Alliance
Danger min radius	0.2	0.2	TBD
Danger max radius/Inner min radius	0.31	0.31	TBD
Inner max radius/Outer min radius	0.42	0.42	TBD
Outer max radius	0.53	0.53	TBD

Table 3 Shadow Zone Dimensions

	2D	3D	Alliance
X [m]	[0.0 : +0.64]	[0.0 : +0.64]	[TBD]
Y [m]	[-0.2 : +0.2]	[-0.2 : +0.2]	[TBD]
Z[m]	0.0	[-0.2 : +0.2]	[TBD]

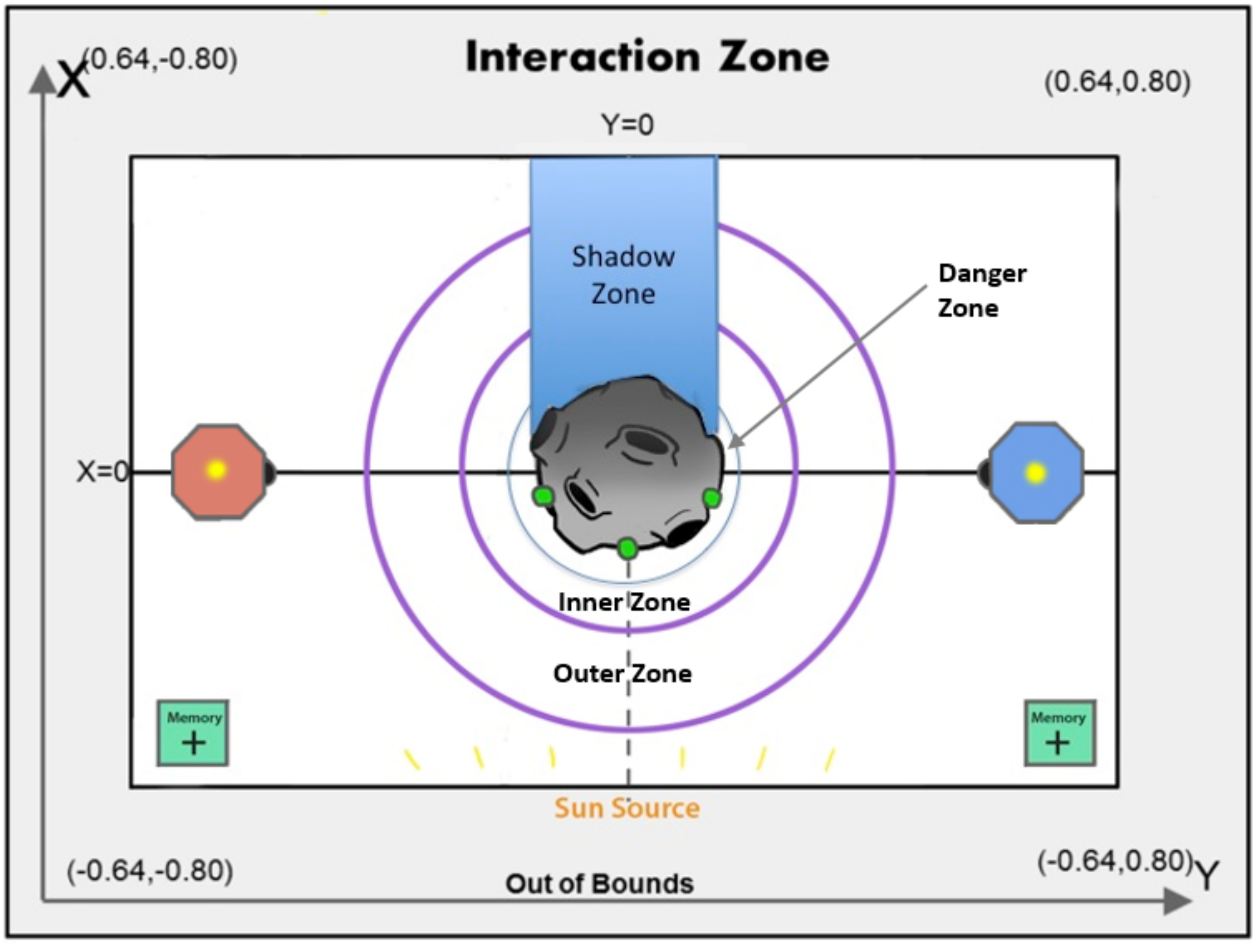

Diagram not to scale

Figure 1 Game Overview

1.2 Satellite

Each team will write the software to command a SPHERES satellite to move in order to complete the game tasks. A SPHERES satellite can move in all directions using its twelve thrusters. The actual SPHERES satellite, like any other spacecraft, has a fuel source (in this case, liquid carbon dioxide) and a power source (in this case, AA battery packs). These resources are limited and must be used wisely. Therefore, the players of Zero Robotics are limited in the use of real fuel and batteries by virtual limits within the game. This section describes the limits to which players must adhere to in order to wisely use virtual SPHERES resources.

1.2.1 ZR User API

The non-game-specific functions used to control the SPHERES satellite in Zero Robotics can be found in a document titled "ZR User API" on the Tutorials webpage under "Other Resources". There is also a link to this document at the end of section 5. Game specific functions, along with a link to the standard ZR User API functions, are also provided in section 5 of this document.

1.2.2 Time

Players have 240s to take and upload as many photos as possible. After 240s scores will be final and compared.

1.2.3 Fuel

Each player is assigned a virtual fuel allocation (Table 4) which is the total sum of fuel used in seconds of individual thruster firing. Calling the function `float getFuelRemaining()` returns remaining fuel. Once the allocation is consumed, the satellite will not be able to respond to SPHERES control commands. It will fire thrusters only to avoid leaving the Interaction Zone or colliding with the other satellite.

Table 4 Fuel Allocation

	2D	3D	Alliance
Fuel Allocation [s]	120s	120s	TBD

The virtual fuel allocation is consumed any time the thrusters are fired. Potential reasons include:

- Motion initiated by player
- Motion initiated by the SPHERES controller to avoid leaving the Interaction Zone (see section 1.6)
- Motion initiated by the SPHERES controller to avoid a collision with the other satellite

1.2.4 Inter-satellite Communications

The satellites have the ability to communicate with each other using binary messages. The API functions `sendMessage` and `receiveMessage` may be used to send data between the satellites. The bandwidth available to the satellites is as follows: (adding function for cooperation)

Table 5 Inter-satellite communications bandwidth

	2D	3D	Alliance
Message size	Unsigned char	Unsigned char	TBD

Note: bandwidth allocation may either increase or decrease as the tournament progresses.

1.2.5 Code Size

A SPHERES satellite can fit a limited amount of code in its memory. Each project has a specific code size allocation. When you compile your project with a code size estimate, the compiler will provide the percentage of the code size allocation that your project is using. Formal competition submissions require that your code size be 100% or less of the total allocation.

1.3 Initial Position

Each satellite starts on the X axis on opposite sides of the asteroid.

The SPHERES satellites are deployed at:

Table 6 SPHERES Satellite Deployment Locations

	2D	3D	Alliance
Red			
X [m]	0.0	0.0	TBD
Y [m]	-0.6	-0.6	TBD
Z[m]	0.0	0.0	TBD
Blue			
X [m]	0.0	0.0	TBD
Y [m]	0.6	0.6	TBD
Z[m]	0.0	0.0	TBD

1.3.1 PlayerID

Users will identify themselves as "playerID = 0" and opponents as "playerID = 1" for all games, whether or not they are the red SPHERES satellite or the blue one.

1.4 Game play

For 2D games, there are 4 possible points of interest (POIs). The 4 points of interest will be randomly placed on the surface of the asteroid on the half closest to the sun and every 60 seconds a random 2 of these points will be visible.

For the 3D game, A *random set of 3 POIs* will be visible on the surface of the asteroid (sunny side) *every 60 seconds starting at time =0*. Each set of POIs will be distributed symmetrically with respect to the x-z plane with two of the points equal distance from y=0 in the + y and – y directions and the third point located on y=0.

Visible POIs are assigned an identification number (ID). For the 2D game the ID # will be 0 or 1. For the 3D game the ID # will be 0, 1 or 2.

Call the game function `getPOILoc(float pos[3], int id)` to find the location of each visible POI. The POI ID is used in the function for taking pictures. See section 1.4.1.

Table 7 Number of Points of Interest

	2D	3D	Alliance
Total number of POIs	4	6	TBD
Number of POIs visible at one time	2	3	TBD

Table 8 Asteroid and SPHERES Measurements

	2D	3D	Alliance
Asteroid radius (m)	0.2	0.2	TBD
SPHERE radius (m)	0.11	0.11	TBD

1.4.1 Picture Taking

Pictures are taken by calling the function `void takePic(int poiID)`

For taking pictures in 2D and 3D, there are two possible orbits from which the satellite can take pictures: a low orbit in the Inner Zone and a high orbit in the Outer Zone (dimensions listed above in Table 2) In order for a picture to be valid there are three qualifications:

- The satellite must be in the Inner or Outer Zone
- The angle between the line connecting the center of the sphere and the POI and the line connecting the POI and the center of the asteroid must be smaller than the Max Angle for the zone the satellite is in as described in Table 9
- The satellite must be facing the POI (with a tolerance of .05 radians).

Table 9 Max Angle of Zones (in radians)

	2D	3D	Alliance
Inner Zone	0.8	0.8	TBD
Outer Zone	0.4	0.4	TBD

For the satellite to be in the correct orbit, the center point of the satellite must be within the bounds of the respective Zone (Figure 2). Scoring will be determined by which orbit the satellite is in; more points will be awarded for pictures taken in the high orbit. The points values awarded for pictures taken in the two orbits (inner zone and outer zone) are provided in Table 10. The POI must be within field of view for the pictures to score points. The field of view is different for the two orbits (Figure 2).

Table 10 Picture Values in Inner and Outer Zones

	2D	3D	Alliance
Inner Zone	2	2	TBD
Outer Zone	3	3	TBD

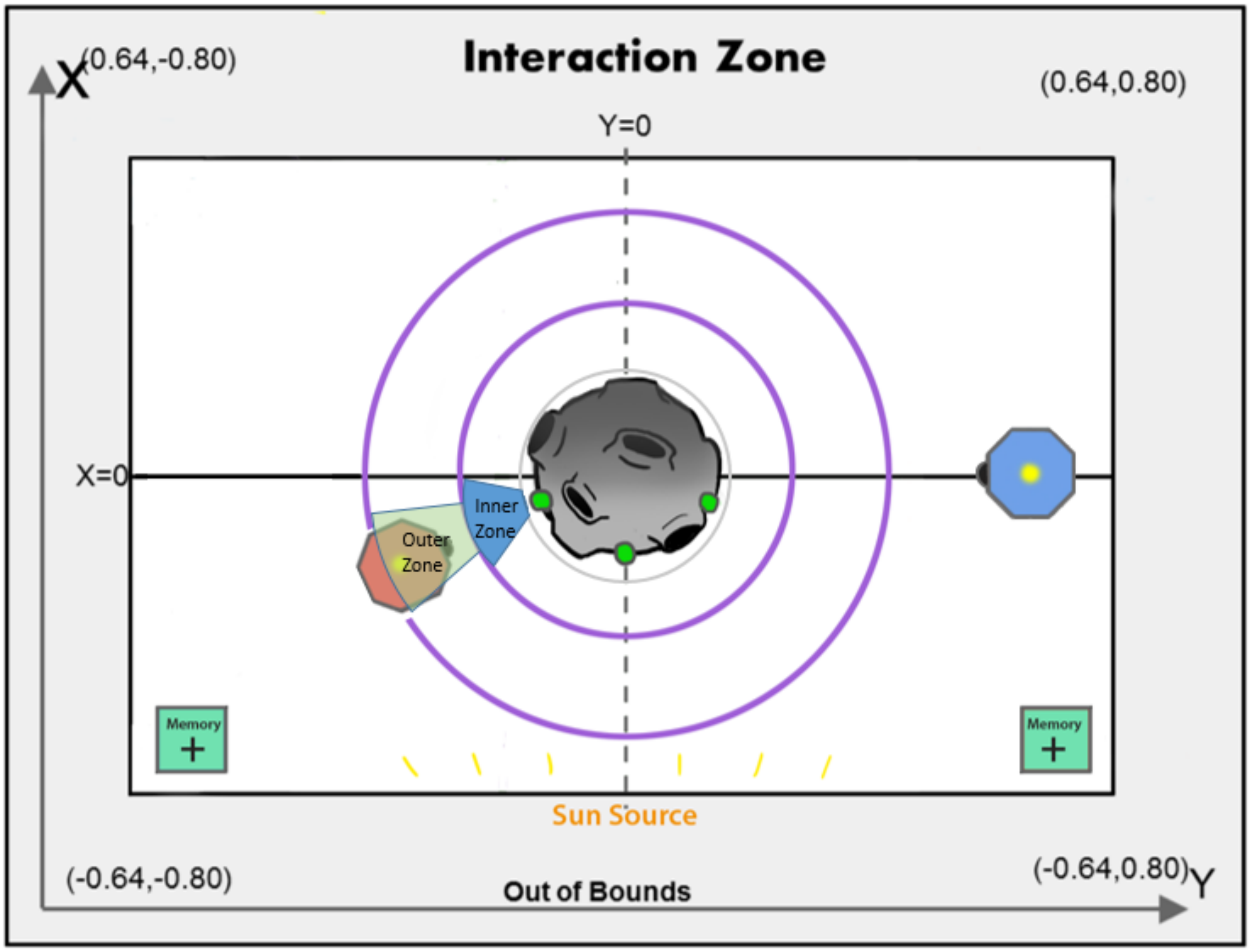

Diagram not to scale

Figure 2 Picture Taking Zones

The satellite can only store 2 photos at a time unless it has obtained an upgrade from a memory pack (see 1.4.2). The camera will be disabled once the photo storage is full until the pictures are uploaded.

- The function `int getMemorySize()` returns the current limit of picture storage.
- The function `int getMemoryFilled()` returns the number of valid pictures (taken from the right distance and angle from the poi) currently saved in camera

The camera will be disabled for 3 seconds each time after the `void takePic(int poiID)` function is called.

Pictures of a single point of interest cannot be taken from the same orbit within each 60s window.

In 2D mode, there are four possible pictures to take in each 60s window, a picture of each of the points can be taken from each orbit once. These limitations are reset after each 60s window.

In 3D mode, there are six possible pictures to take in each 60s window, a picture of each of the points can be taken from each orbit once. These limitations are reset after each 60s window.

In order to take a picture, the point of interest must be within your field of view. This means that that your opponent cannot be blocking your field of view (Figure 3).

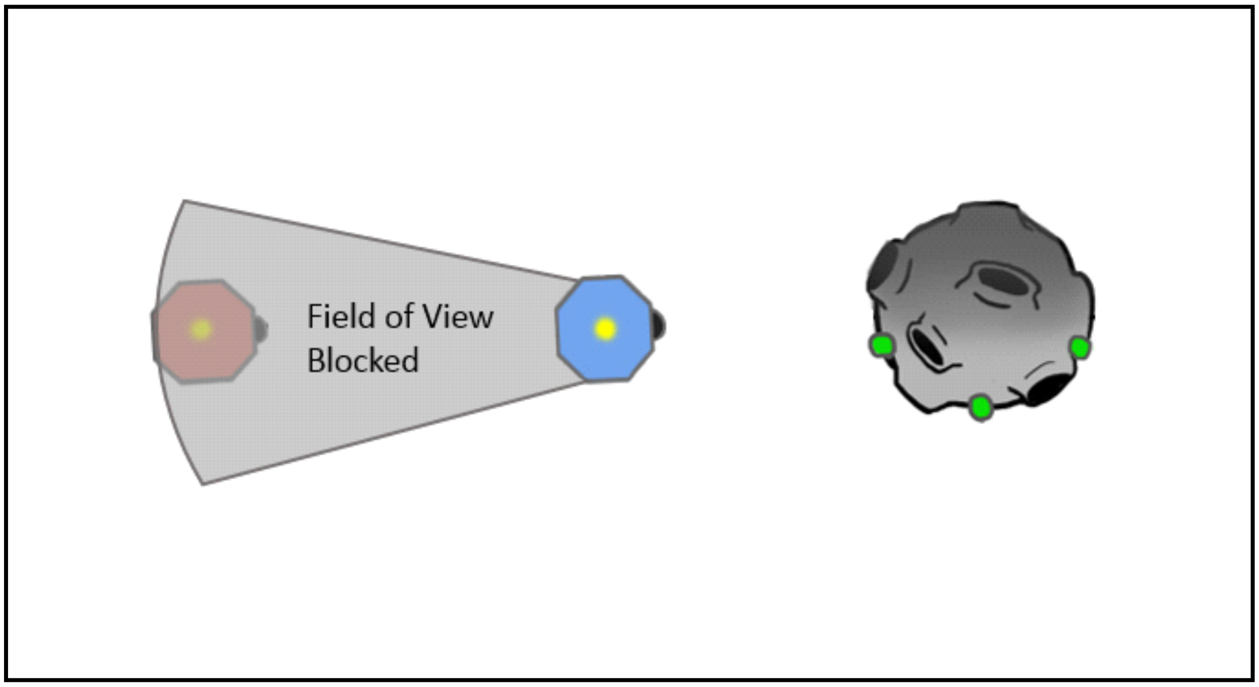

Diagram not to scale

Figure 3 Blocking Opponent's Field of View

To take a pictures:

- Camera must pointing at POI
- Game function `bool alignLine(int poiID)` Returns true if the SPHERE is facing the poi whose ID is given
- POI must be in field of view (an uninterrupted line can be made from camera to POI)
- Game function `void takePic(int poiID)` must be called

Hint: Consider using Spherical Coordiantes to locate the POI in the 3D game (see Tutorials for an explanation of how to use Spherical Coordinates).

1.4.2 Memory Upgrade Packs

Memory Upgrade packs allow players to store more pictures on their satellite. Memory Upgrade packs are located on the corners of the sunny side of the interaction zone. Memory packs will remain in play throughout game play. Once a memory pack has been taken by a satellite, it is then in the possession of that satellite and may not be picked up by the other satellite. A satellite can hold up to 2 memory packs. Each pack gives the satellite enough memory to hold 1 more photo. Calling the function `bool hasMemoryPack(int playerId, int packID)` returns "true" if specified player has specified memory pack

Table 11 Memory Upgrade Pack Locations

	2D	3D	Alliance
Memory Pack 1(ID = 0)			
X [m]	-0.5	-0.5	TBD
Y [m]	-0.6	-0.6	TBD
Z[m]	0.0	0.0	TBD
Memory Pack 2(ID = 1)			
X [m]	-0.5	-0.5	TBD
Y [m]	0.6	0.6	TBD
Z[m]	0.0	0.0	TBD

To collect Memory Upgrade packs see section 1.5 ITEM COLLECTION.

1.4.3 Solar Flares

Solar Flares pose a danger to satellites. Solar Flares occur randomly every 70s starting 30 seconds after the start of the game. Each Solar Flare lasts 3 seconds. All satellites exposed to the solar radiation for any period of time lose the photos stored in their memory. Satellites exposed to the solar radiation will lose 1 point every second they are exposed to the solar radiation, unless the satellites are powered off prior to exposure with the solar radiation.

While the satellite is powered off, players:

- Reduce Damage (lose half the points of satellites powered on)
- Lose any pictures currently stored on satellite
- Must wait 5 seconds for their instruments to turn on and warm up
- Cannot take pictures
- Cannot stop their satellites from drifting (collisions may occur)

To power off:

- Game function `void turnOff()` must be called

To power back on:

- Game function `void turnOn()` must be called

The other way to prevent damage is to seek shelter in the shadow zone. Players are protected from solar radiation in the shadow zone, preserving pictures and points. To be deemed safe, the center point of the satellite must be in the shadow zone.

Players will have a 30 second warning before a solar flare arrives. This warning can be received by calling the function: `int getNextFlare()`. During the 30 second period prior to the next solar flare `int getNextFlare()` returns the number of seconds until the next solar flare. Calling this function any time outside this 30 second window will return -1. Calling this function during the solar flare will also return -1.

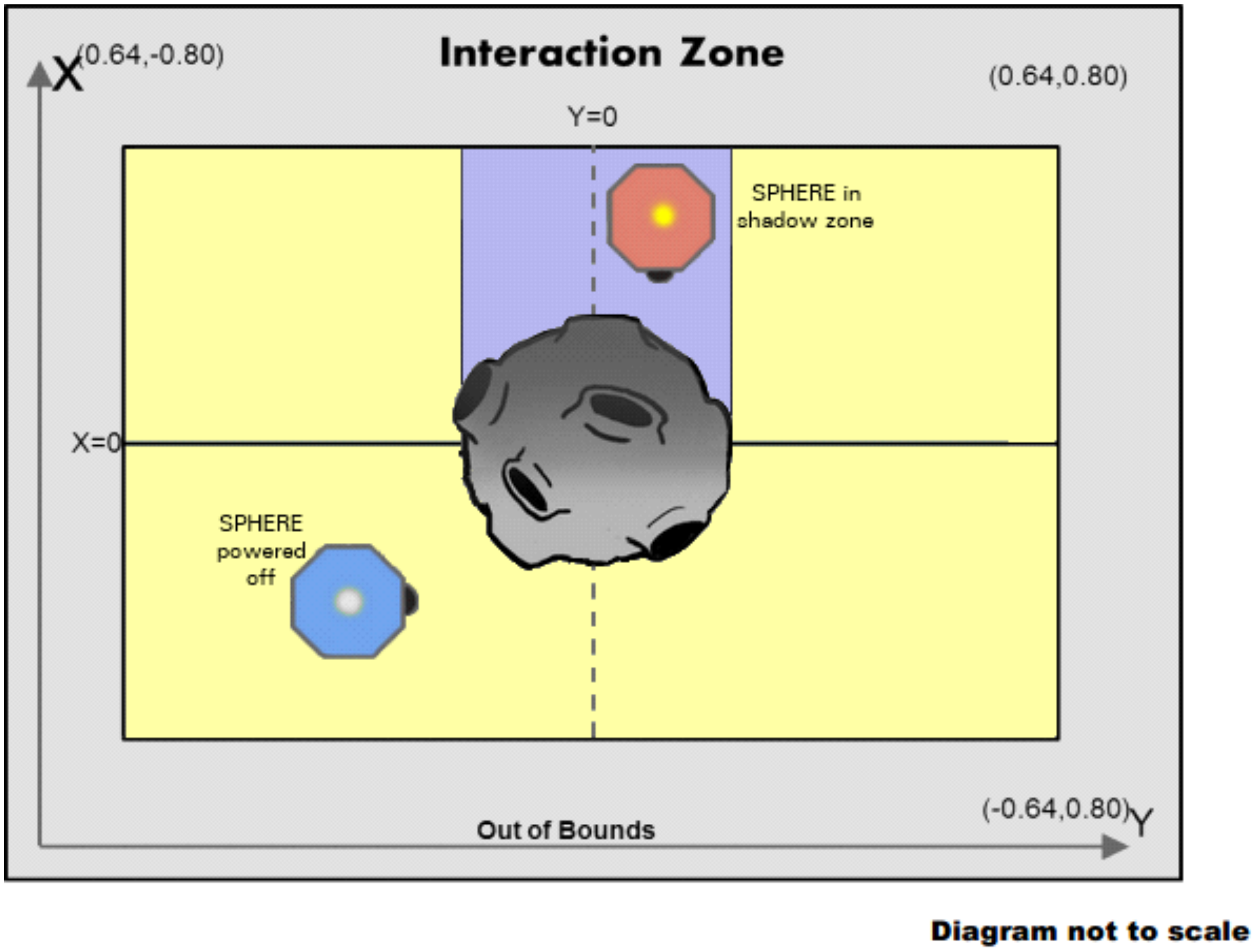

Figure 4 Solar Flare

1.4.4 Upload

Once the satellite's memory is full, the pictures must be uploaded. Pictures must be uploaded to score full points for taking the picture. Without uploading, only 0.1 points will be awarded for successfully taking a photo and .01 points will be awarded for attempting to take a photo.

During the upload process the camera will be disabled for 3 seconds.

There is a Discover Bonus for the first satellite to upload a picture of a specific point of interest within a cycle. The Discover Bonus is +0.5. The Discover Bonus resets with each 60s POI rotation cycle.

To upload photos:

- The center of the SPHERE must be either outside of the Danger Zone and both Picture Taking zones, or in the Shadow Zone
- The game function `void uploadPic()` must be called

1.4.5 Collisions

While it is not possible to collide with the other satellite, collision with the asteroid is possible. If the center point of your satellite enters the Danger Zone, it is considered a collision with the asteroid.

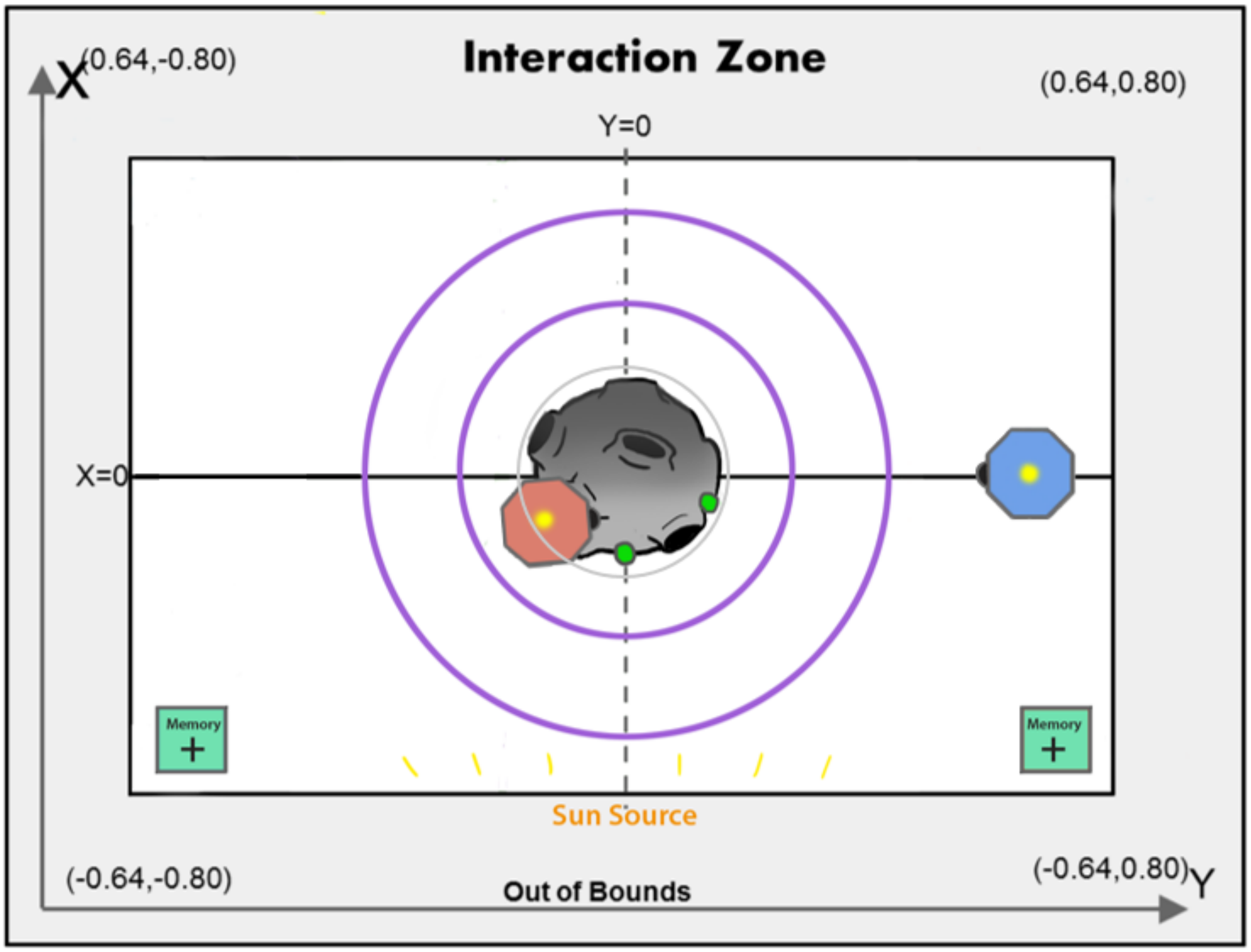

Diagram not to scale

Figure 5 Asteroid Collision

If the satellite crashes into the asteroid:

- The satellite will come to a full stop
- The satellite can only move away from the asteroid
- There will be a fuel penalty equal to 2 thruster seconds of fuel per second in the asteroid
- Points will be lost equal to the number of POI affected in crash site

The crash site is all points on the asteroid within a satellite diameter (.22m) of the satellite's center point.

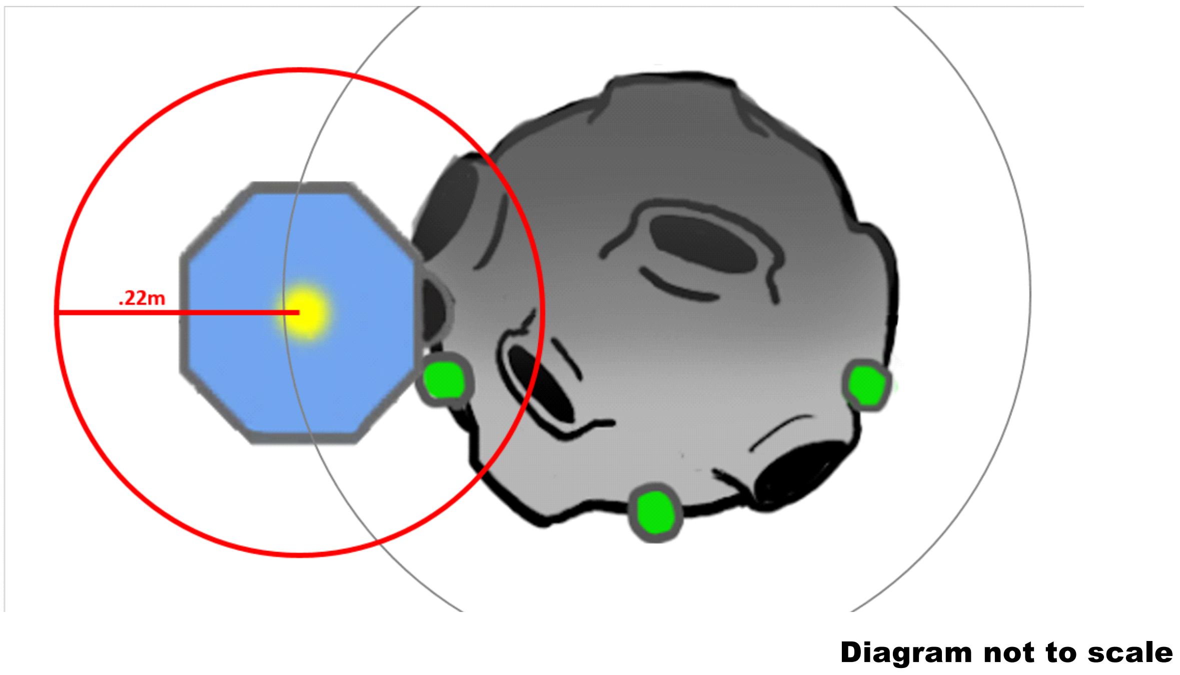

Figure 6: Asteroid Collision Close-up

In the figure above, the portion of red circle that intersects with the asteroid represents the crash site. All points of interest within this crash site will be counted against the team whose satellite crashes into the asteroid.

1.4.6 End of Game

The game ends when time runs out.

1.5 Item Collection

To increase the memory capacity of the satellites, teams have the opportunity to collect memory upgrade packs found in the 2 corners of the interaction zone closest to the sun.

Table 12 Memory Upgrade Pack Locations (repeated)

	2D	3D	Alliance
Memory Pack 1			
X [m]	-0.5	-0.5	TBD
Y [m]	-0.6	-0.6	TBD
Z[m]	0.0	0.00	TBD
Memory Pack 2			
X [m]	-0.5	-0.5	TBD
Y [m]	0.6	0.6	TBD
Z[m]	0.0	0.00	TBD

In order to pick up an item, you need to perform a spinning maneuver (see Figure 7). The steps to collect the Memory Upgrade packs are:

- Position the satellite within 0.05m of the item's center.
  - The satellite's velocity must be less than 0.01m/s.
  - The satellite's angular velocity must start at less than 2.3°/s.
- Rotate the satellite >90° along about the Z axis for 2D. Do not attempt to rotate faster than 80°/s.

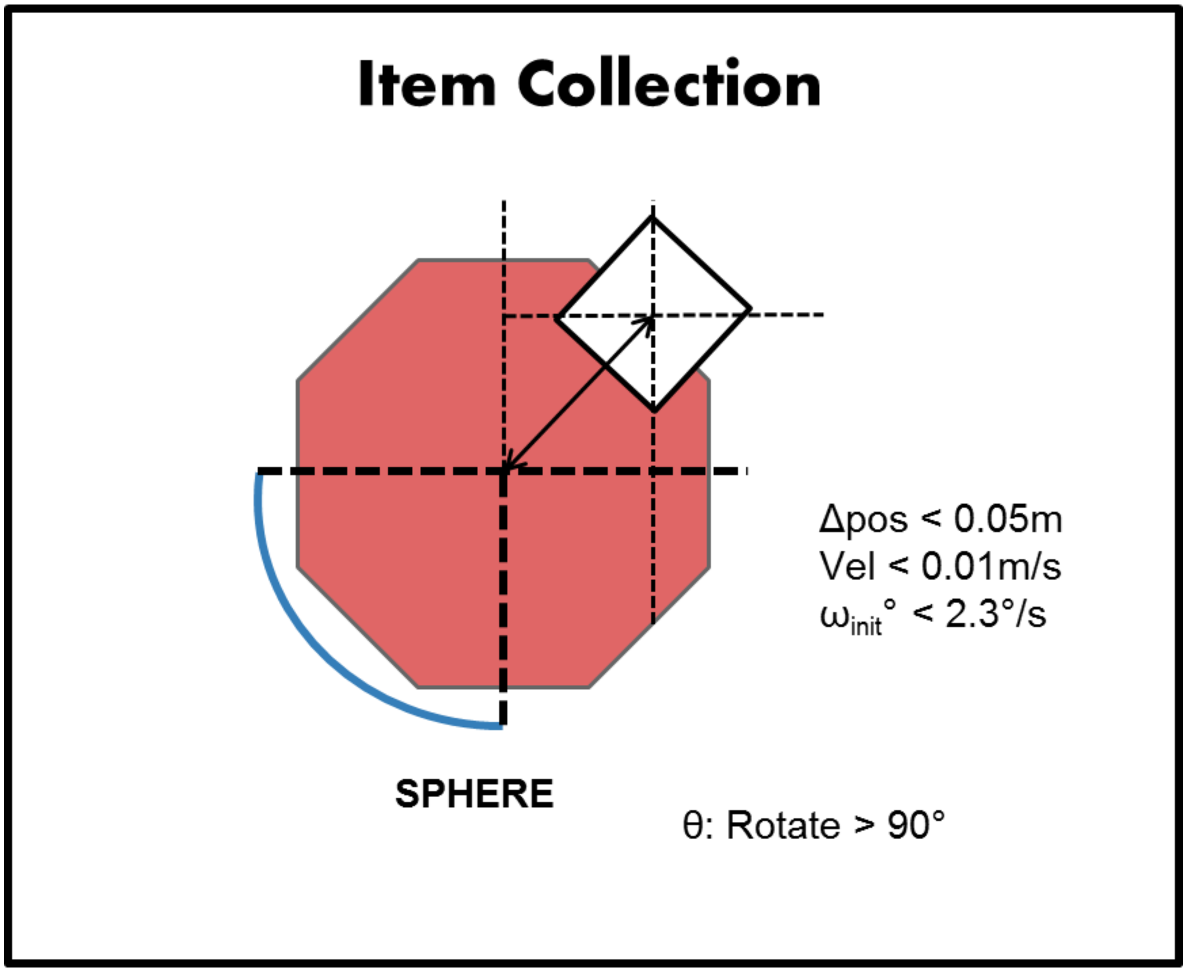

Figure 7 Maneuver to Collect Item

1.6 Out of Bounds

You must remain within the boundaries of the Interaction Zone to avoid a fuel penalty.

If you exit out of bounds, the SPHERES controller will override your commands and force the satellite to stop its motion in the direction that would continue to push it out of bounds (other directions are not affected). The fuel used to stop this motion will be charged to your fuel usage. There is an additional fixed penalty of **2 thruster-seconds** (4% of total initial fuel) for each second spent out of bounds.

2 Scoring

Your satellite begins with a score of 9 points. Your final score is based on the pictures you take and upload minus the damage obtained from solar flares and collisions. Pictures are worth different point values depending on which orbit they were taken from. The score will be totaled from the number and location of the photos taken. The seconds exposed to the solar radiation will subtract points from your score.

The scoring calculation is as follows:

Table 13 Point Values

	2D	3D	Alliance
Number of points at time = 0	9	9	
Pictures attempted (both valid and invalid pictures)	.01	.01	TBD
Non Uploaded valid photos	0.1	0.1	TBD
Uploaded: Taken in Inner Zone	2	2	TBD
Uploaded: Taken in Outer Zone	3	3	TBD
Collision with Asteroid: Points deducted per second per POI contacted	-1	-1	TBD
Discover Bonus	+0.5	+0.5	TBD
Solar Flare: Points deducted per second: SPHERE powered, outside shadow zone	-1	-1	TBD
Solar Flare: Points deducted per second: SPHERE un-powered, outside shadow zone	-0.5	-0.5	TBD
Solar Flare: Points deducted per second: SPHERE inside shadow zone	0	0	TBD
To avoid ties:		See note below	

Note: At the end of the match, to avoid ties, points will be subtracted from the score based on the following calculation: (satellite distance from the center of asteroid) x (0.00001)

3 Tournament

A Zero Robotics tournament consists of several phases called *competitions*. The following table lists the key deadlines for the 2014 tournament season:

Table 14 Tournament Key Dates

Date (2014)	Event
Sep 6 (Sat)	Kick-off webcast
Sept 26 (Fri)-extended to Sept 30	2D Simulation Competition Deadline (end of practice)
Updated: Nov 7 (Fri)	3D Simulation Competition Deadline
Updated: Nov 15 (Sat)	Alliance Formation Event
Updated: Dec 12 (Fri)	Alliance Simulation Submission Deadline
Updated: Jan 5 (Mon)	ISS Code Submission Deadline
mid-Jan TBA	ISS Final Competition

The rankings in each competition are determined by a *Leaderboard*, described below. The 2D Simulation Competition is *not* an elimination round; everyone who submits code which achieves a nozero score against a “blank” player (player with no code) to the 2D Simulation Competition advances to the 3D Competition. At the end of the 3D Simulation Competition an elimination round takes place, with the top TBD ranking teams moving to the Alliance phase of the Tournament. The top TBD ranking teams will then form alliances of three (3) teams each: TBD alliances will be formed. The alliances will compete against each other, with the top TBD alliances moving to the ISS Final Competition.

3.1 The Leaderboard

This year’s tournament uses a continuously updated ranking system called *The Leaderboard*. The Leaderboard uses a system similar to the Elo rating system for chess players called Whole History Ranking, as well as ideas from the *TrueSkill®* rating system used for the Xbox Live gaming platform. The Leaderboard tracks all matches a team has played against other players in the course of the competition and creates a rating based on the outcomes. At the end of each competition phase, the final standings on the Leaderboard will determine which teams advance to the next phase.

3.1.1 Rating

(To be updated to reflect current leaderboard documentation)

3.1.2 Playing Matches

(To be updated to reflect current leaderboard documentation)

3.1.3 Competition Submissions

(To be updated to reflect current leaderboard documentation)

3.2 2D Practice Simulation Competition

All teams that complete a valid registration are eligible to participate in the 2D practice simulation competition.

3.3 3D Simulation Competition

All teams that complete a valid registration and submit code that achieves a non-zero score when competed against a “blank player” (a player with no code) by the 2D practice competition deadline are eligible to participate in the 3D simulation competition.

When the 3D competition starts the game will be updated with new challenges and the corresponding TBA values will be announced.

3.4 Alliance Formation Event

The top ranked 81 teams on the leaderboard at the end of 3D simulation play plus 3 wild cards (a total of 84 teams) will form 28 alliances of three (3) teams each. The 28 alliances will work cooperatively to complete the semifinals and, if not eliminated, the finals. Teams ranked below rank 81 will be invited to participate in the Virtual Finals. See Section 3.8 for details about the Virtual Finals.

The ranking of the teams will be determined by the rankings at the close of the 3D leaderboard.

After the 3D leaderboard rankings are announced, advancing teams will have three days to contact each other to discuss their alliance preferences. Any teams that do not wish to continue in the Tournament will have the opportunity to cede their position to the next ranked team.

The Alliance Formation Event will take place on November 15th . The alliance formation event or “draft day” is an online event during which teams will follow the alliance selection process described below to invite other teams into alliance and/or accept their place within an alliance. At least one representative from each team must participate in the online “draft day” event for the team to continue to the alliance phase. Additional details about the online “draft day” and how to participate will be provided in advance of the event.

Due to the large number of teams involved, the draft will be split into two parts as shown in Figure 8. All teams ranked with odd numbers will participate in Draft part 1 and all teams ranked with even numbers will participate in Draft part 2. Each draft will include 42 teams.

Draft part 1
teams with odd numbered rankings

1	29	57
3	31	59
5	33	61
7	35	63
9	37	65
11	39	67
13	41	69
15	43	71
17	45	73
19	47	75
21	49	77
23	51	79
25	53	81
27	55	**WC**

Draft part 1: 10:00 -13:00 EST / 16:00-19:00 CET

Draft part 2
teams with even numbered rankings

2	30	58
4	32	60
6	34	62
8	36	64
10	38	66
12	40	68
14	42	70
16	44	72
18	46	74
20	48	76
22	50	78
24	52	80
26	54	**WC**
28	56	**WC**

Draft part 2: 13:00- 16:00 EST / 19:00-22:00 CET

Figure 8: Division of Teams between Draft Events

The alliance selection process will follow a serpentine pattern (illustrated in Figure 9):

- The highest ranked team selects their partner from any except the top 6 ranked teams in their draft.
- The next highest ranked team selects their partner from any of the remaining teams except the top 6 ranked teams in their draft
- 14 pairs are created in this manner
- A break takes place for the new pairs to discuss their selection for the 3rd alliance team
- The "lowest" ranked pair then selects their 3rd team from the remaining 14 teams
- The "2nd lowest" rank pair make the next selection
- Continue until all 14 alliances are formed.
- This process is duplicated in both draft events to end with a total of 28 alliances

Rule: *No more than 2 of the teams in an alliance can be from the same country. (If necessary eastern and western US will be treated as separate countries.)*

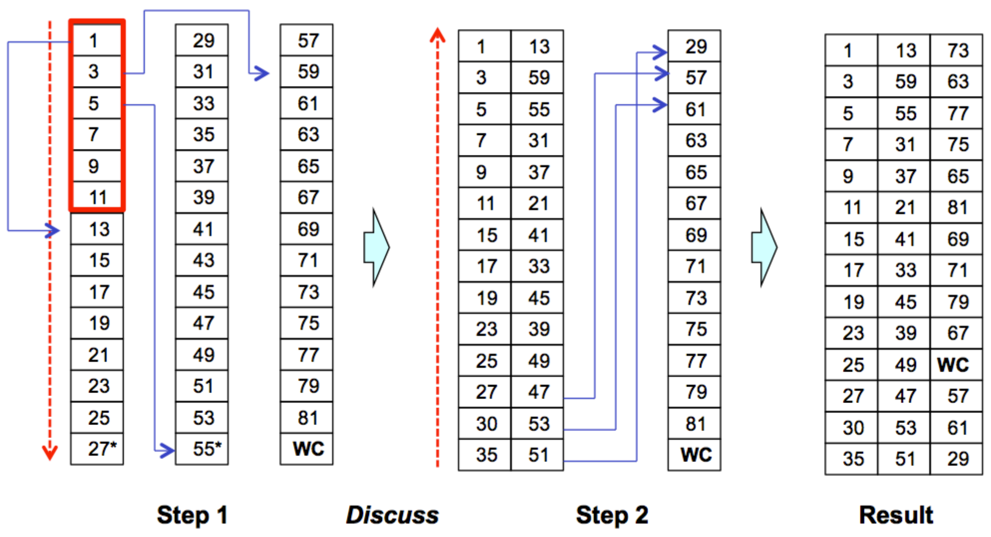

Figure 9 Alliance Creation Process- demonstrated for teams with odd number rankings

(*The example shown above is approximate and depends on the placement of the pilot teams)

3.5 *Wild Cards Explained*

This year we welcome Russian and Mexican partners to the Zero Robotics High School Tournament 2014. As this is their pilot year, one Russian alliance (comprised of two Russian teams and a third team, TBD) and a Mexican alliance (comprised of one Mexican team and two other teams, TBD) will participate in the ISS competition as shown in Figure 10. Therefore we have reserved 3 spots for the Russian and Mexican partners out of the total 84 teams advancing to the alliance phase. Since we cannot predict the ranking of these teams during the 3D competition, 3 wildcards spots have been included. The pilot teams will advance whether or not they place within the top 81 ranked teams on the leaderboard. However, if they do rank within the top 81 teams, this will open up additional spots in ranking order for teams ranked below 81. One of the pilot alliances will be included in the draft with the teams ranked with even numbers and the other will be included in the draft with the teams ranked with odd numbers. The TBD teams in each of the pilot alliances will be determined during their respective draft events.

Rus	Rus	TBD

Mex	TBD	TBD

Figure 10: Pilot alliances

3.6 *Semifinal Simulation Competition*

The 28 alliances created during the Alliance Formation Event will participate in the semifinal simulation competition.

When the semifinal competition starts, the game will be updated with new challenges and the corresponding TBA values will be announced. These new challenges are intended to be substantial enough to require participation of all alliance teams in preparing competition submissions.

3.7 *ISS Final Competition*

The top 12 alliances on the leaderboard at the end of semifinal play plus two wildcard alliances (a total of 14 alliances) will advance to the ISS Finals Competition. Two of these alliances will include the alliances formed with the Russian and Mexican pilot teams. Alliances ranked below rank 14 will be invited to participate in the Virtual Finals. See Section 3.8 for details about the Virtual Finals.

The ISS finals will take place aboard the International Space Station with live transmission to MIT. All teams will be invited to live broadcast events at MIT (US) and ESTEC (EU.)

3.7.1 Overview and Objectives

Running a live competition with robots in space presents a number of real-world challenges that factor into the rules of the competition. Among many items, the satellites use battery packs and CO_2 tanks that can be exhausted in the middle of a match, and the competition must fit in the allocated time. This section establishes several guidelines the Zero Robotics team intends to follow during the competition. Keep in mind that as in any refereed competition, additional real-time judgments may be required. Please respect these decisions and consider them final.

Above all, the final competition is a demonstration of all the hard work teams have put forward to make it to the ISS. The ZR staff's highest priority will be making sure every alliance has a chance to run on the satellites. It is also expected that the competition will have several "Loss of Signal" (LOS) periods where the live feed will be unavailable. We will attempt to make sure all teams get to see a live match of their player, but finishing the competition will take priority.

To summarize, time priority will be allocated to:

1) Running all submissions aboard the ISS at least once
2) Completing the tournament bracket
3) Running all submissions during live video

We hope to complete the tournament using only results from matches run aboard the ISS, but situations may arise that will force us to rely on other measures such as simulated matches.

3.7.2 Competition Format

The alliances will be divided into 2 conferences for the ISS competition. All teams ranked with odd numbers will participate in Conference A; all teams ranked with even numbers will participate in Conference B, as shown in Figure 11.

Conference A Alliance ranks	Conference B Alliance ranks
1,3,5,7,9,11,13	2,4,6,8,10,12,14

Figure 11: Division of Teams between Conferences

Each conference will include one "bye" team (alliances ranked #1 and #2 automatically advance to the conference semi-finals) and 2 brackets of 3 alliances each (as shown in Figure 12). Each bracket will play 3 matches in round-robin style: alliance A vs. B, B vs. C, and C vs. A.

After the round-robins are complete, there will be a winner of each bracket (shown as BR1, BR2 in Figure 12.) The following rules determine the winner:

1. The alliance with the most wins advances
2. If alliances are tied for wins, the alliance with the highest total score advances
3. If scores are tied, simulation results will be used to break the tie

The semi-final match between the top 2 bracket winners and the "bye" team will also be played in round-robin style. The winner of this match is determined in the same way as the bracket winners:

1. The top 2 alliances with the most wins in their bracket, advance
2. If there is a tie for wins, the alliance(s) with the highest total score in their bracket advance
3. If scores are tied, simulation results will be used to break the tie

The winning alliance from each conference will play a single match to determine the Zero Robotics ISS Champion. The losing alliance will be awarded 2nd place.

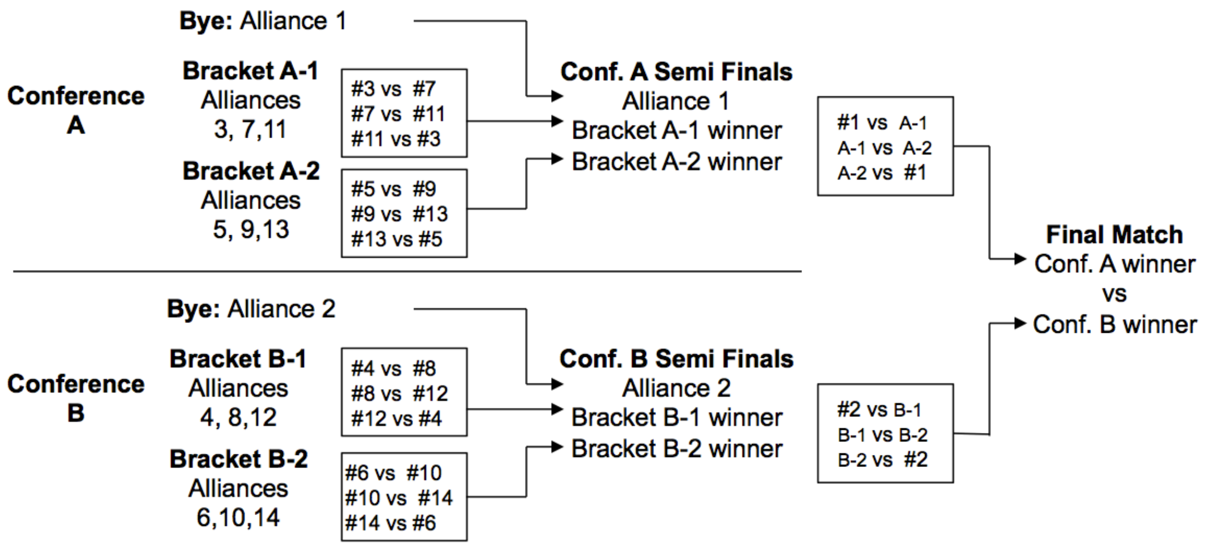

Figure 12: ISS Competition Bracket

Definition: *Successful Match*

- Both satellites move correctly to initial positions
- Both satellites have normal motion throughout the test
- Both satellites return a valid score
- Neither satellite expends its CO_2 tank during a test run

Definition: *Simulated Match*

In advance of the competition, the ZR Team will run a simulated round robin competition between all participating teams. The results from matches in this competition will be used in place of ISS tests if necessary (see below.) The results of a simulated match will only be announced if they are used in the live competition.

3.7.3 Scoring Matches

TBD

3.8 Virtual Finals Simulation Competition

All teams participating in the 3D competition that do not advance to the Semi Final competition (Alliance Phase) and all teams participating in the Semi Final Competition that do not advance to ISS Finals will be invited to participate in the Virtual Finals.

The Virtual Finals game will be identical to the Semi Final competition (Alliance Phase) game. Teams participating in the Virtual Finals will submit to a Virtual Finals Leaderboard.

Teams may choose to participate in the Virtual Finals as individual teams or as alliances. Teams that did not participate in the original alliance formation event are welcome to create alliances, if desired. However, once a team/alliance submits code to the Virtual Finals Leaderboard the team/alliance composition cannot be changed.

Once the Virtual Finals Leaderboard closes, the top ranked TBD teams will advance to the Virtual Finalist competition. The final format of the Virtual Finalist competition will depend on the number of teams participating in the Virtual Finals and will be announced as the tournament progresses. The Virtual Finalist competition will be run in simulation by the ZR staff. The results of the competition will remain secret until the ISS Final competition.

During the live broadcast of the ISS Finals the ZR Staff will replay the Virtual Finalist matches with commentary and will announce the results. ***Time permitting, the final championship match of the Virtual Finalist competition will be competed aboard ISS.***

4 Season Rules

4.1 Tournament Rules

All participants in the Zero Robotics High School Tournament 2014 must abide by these tournament rules:

- The Zero Robotics team (MIT / Top Coder / Aurora) can use/reproduce/publish any submitted code.
- In the event of a contradiction between the intent of the game and the behavior of the game, MIT will clarify the rule and change the manual or code accordingly to keep the intent.
- Teams are expected to report all bugs as soon as they are found.
    - A “bug” is defined as a contradiction between the intent of the game and behavior of the game.
    - The intent of the game shall override the behavior of any bugs up to code freeze.
    - Teams should report bugs through the online support tools. ZR reserves the right to post any bug reports to the public forums (If necessary, ZR will work with the submitting team to ensure that no team strategies are revealed).
- Code and manual freeze will be in effect 3 days before the submission deadline of a competition.
    - Within the code freeze period the code shall override all other materials, including the manual and intent.
    - There will be no bug fixes during the code freeze period. All bug fixes must take place before the code freeze or after the competition.
    - The code is finalized at the ISS Final Competition freeze (unless there is a critical issue which will affect the final tournament, including lessons learned from ground hardware testing and simulation.)
- Game challenge additions and announcement of TBA values in the game manual may be based on lessons learned from earlier parts of the tournament.

4.2 Ethics Code

- The ZR team will work diligently upon report of any unethical situation, on a case by case basis.
- Teams are strongly encouraged to report bugs as soon as they are found; intentional abuse of an un-reported bug may be considered as unethical behavior.
- Teams shall not intentionally manipulate the scoring methods to change rankings.
- Teams shall not attempt to gain access to restricted ZR information.
- We encourage the use of public forums and allow the use of private methods for communication.
- Vulgar or offensive language, harassment of other users, and intentional annoyances are not permitted on the Zero Robotics website.
- Code submitted to a competition must be written only by students.

5 ZR User API

5.1 Standard Zero Robotics API Reference

A guide to the standard Zero Robotics API functions is available here:

http://www.zerorobotics.org/documents/10429/374963/ZR_user_API.pdf

Note: Math functions in this table do not need the "api." prefix

5.2 CoronaSPHERES API Reference

The functions in this section are called as members of the game object, that is, game.functionName(arguments);

Name	Description
`int getNextFlare()`	During the 30 second period prior to the next solar flare this function returns the number of seconds until NextFlare. Warnings are only issued 30 seconds in advance. Calling this function any time outside this window will return -1. Calling this function during the flare will also return -1.
`int getMemoryFilled()`	Returns number of valid pictures (taken from the right distance and angle from the poi) currently saved in camera.
`int getMemorySize()`	Returns current limit of picture storage
`void takePic(int poiID)`	Takes picture and stores them in the satellite memory. Camera disabled if picture fills last memory slot. Camera is disabled for 3 seconds each time takePic is called
`void uploadPic()`	Uploads pictures from satellite, disables camera for 3 seconds
`int numActivePOIs()`	Always returns 2 in 2D
`void getPOILoc(float pos[3], int id)`	Returns location of each visible POI. Visible POIs are assigned ID # 0, 1 or 2
`float getScore()`	Returns player's score
`float getOtherScore()`	Returns opponent's score
`void turnOff()`	Turns off satellite to protect from solar flare. Satellite will drift.
`void turnOn()`	Begins process to turn satellite on. Takes 5 seconds. (unless game just beginning)
`float getFuelRemaining()`	Returns remaining fuel
`bool hasMemoryPack(int playerId, int packID)`	Returns true if specified player has specified memory pack
`bool alignLine(int poiID)`	Returns true if the SPHERE is facing the poi whose ID is given. It does not check if poi is in field of view or if sphere is in the correct angle or zone.
`void sendMessage(unsigned char inputMsg)`	Sends **inputMsg** to other satellite
`unsigned char receiveMessage()`	Returns the most recent message sent by other satellite

6 Lists of Figures and Tables

6.1 List of Figures

6.2 List of Tables

7 Revision History

Revision	Date	Changes	By
1.0	9/7/14	Initial release	zerorobotics@mit.edu
1.1	9/14/14	Clarification about various scoring and game function details	zerorobotics@mit.edu
1.2	9/14/14	Clarification about POI IDs	zerorobotics@mit.edu
2.0	9/30/14	Add 3D game details and fix typos, bugs from 2D game, satellite color corrected in figures	zerorobotics@mit.edu
2.1	10/15/14	Replace TBDs in sections 3.4-3.8 with details about Alliance formation, ISS Competition and Virtual Finals). Correct typos.	zerorobotics@mit.edu
2.2	10/26/14	Reflect changes made to algorithm for 3D POI locations, clarifies impacts with the asteroid, updates tournament schedule, corrects typos	zerorobotics@mit.edu