

Zero Robotics Italian Virtual Programming Challenge High-School Tournament 2014:



Conducting Optical Research on Nearby Asteroids (CORONA)
GAME MANUAL V2.3

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.

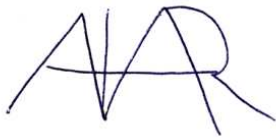
NASA calls upon its fellow scientists at MIT for a plan of action. With the help of these researchers, the joint CORONA program was born. CORONA (Conducting Optical Research On Nearby Asteroids) requires the use of MIT's most recent project, SPHERES, to capture visuals of the closest asteroids to Earth's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



Alvar Saenz-Otero
MIT SPHERES Lead Scientist

Contents

1	GAME OVERVIEW	4
1.1	Game Layout	4
1.2	Satellite	5
1.2.1	ZR User API.....	5
1.2.2	Time	5

1.2.3	<i>Fuel</i>	6
1.2.4	<i>Inter-satellite Communications</i>	6
1.2.5	<i>Code Size</i>	6
1.3	Initial Position	6
1.3.1	<i>PlayerID</i>	7
1.4	Game play.....	7
1.4.1	<i>Picture Taking</i>	7
1.4.2	<i>Memory Upgrade Packs</i>	10
1.4.3	<i>Solar Flares</i>	10
1.4.4	<i>Upload</i>	11
1.4.5	<i>Collisions</i>	11
1.4.6	<i>End of Game</i>	13
1.5	Item Collection	13
1.6	Out of Bounds.....	14
2	SCORING	14
3	TOURNAMENT	15
3.1	The Leaderboard.....	15
3.1.1	<i>Rating</i>	16
3.1.2	<i>Playing Matches</i>	16
3.1.3	<i>Competition Submissions</i>	16
3.3	3D Simulation Competition.....	16
3.6	Semifinal Simulation Competition	18
3.7	ISS Final Competition	18
3.7.1	<i>Overview and Objectives</i>	19
3.7.2	<i>Competition Format</i>	19
3.7.3	<i>Scoring Matches</i>	20
3.8	Virtual Finals Simulation Competition	20
4	SEASON RULES	21
4.1	Tournament Rules	21
4.2	Ethics Code.....	21
5	<u>ZR USER API</u>	22
5.1	Standard Zero Robotics API Reference	22
5.2	CoronaSPHERES API Reference.....	22
6	LISTS OF FIGURES AND TABLES	23
6.1	List of Figures.....	23
6.2	List of Tables.....	23
7	REVISION HISTORY	24

- **Game Overview**

Matches will be played between two SPHERES satellites, controlled by code written by two different teams. 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.

- **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 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 Interaction Zone Dimensions

	3D
X [m]	[-0.64 : +0.64]
Y [m]	[-0.80 : +0.80]
Z[m]	[-0.64 : +0.64]

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 Zone Radii Positions

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

Table Shadow Zone Dimensions

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

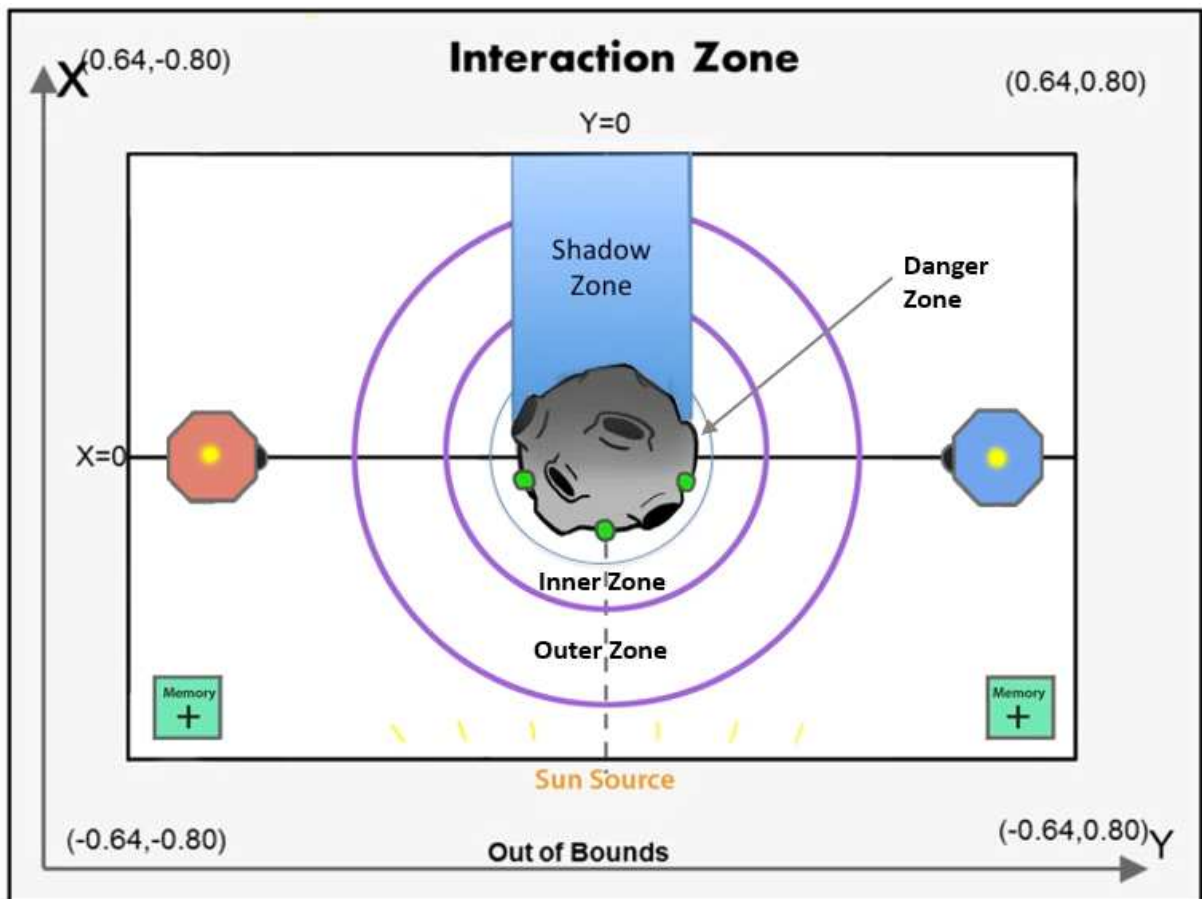


Diagram not to scale

Figure Game Overview

- **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.

- **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.

- **Time**

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

- **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 Fuel Allocation

	3D
Fuel Allocation [s]	120s

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

• 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 Inter-satellite communications bandwidth

	3D
Message size	Unsigned char

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

• 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.

• Initial Position

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

The SPHERES satellites are deployed at:

Table SPHERES Satellite Deployment Locations

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

• 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.

- **Game play**

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 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 Number of Points of Interest

	3D
Total number of POIs	6
Number of POIs visible at one time	3

Table Asteroid and SPHERES Measurements

	3D
Asteroid radius (m)	0.2
SPHERE radius (m)	0.11

- **Picture Taking**

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

For taking pictures in 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 Max Angle of Zones (in radians)

	3D
Inner Zone	0.8
Outer Zone	0.4

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 Picture Values in Inner and Outer Zones

	3D
Inner Zone	2
Outer Zone	3

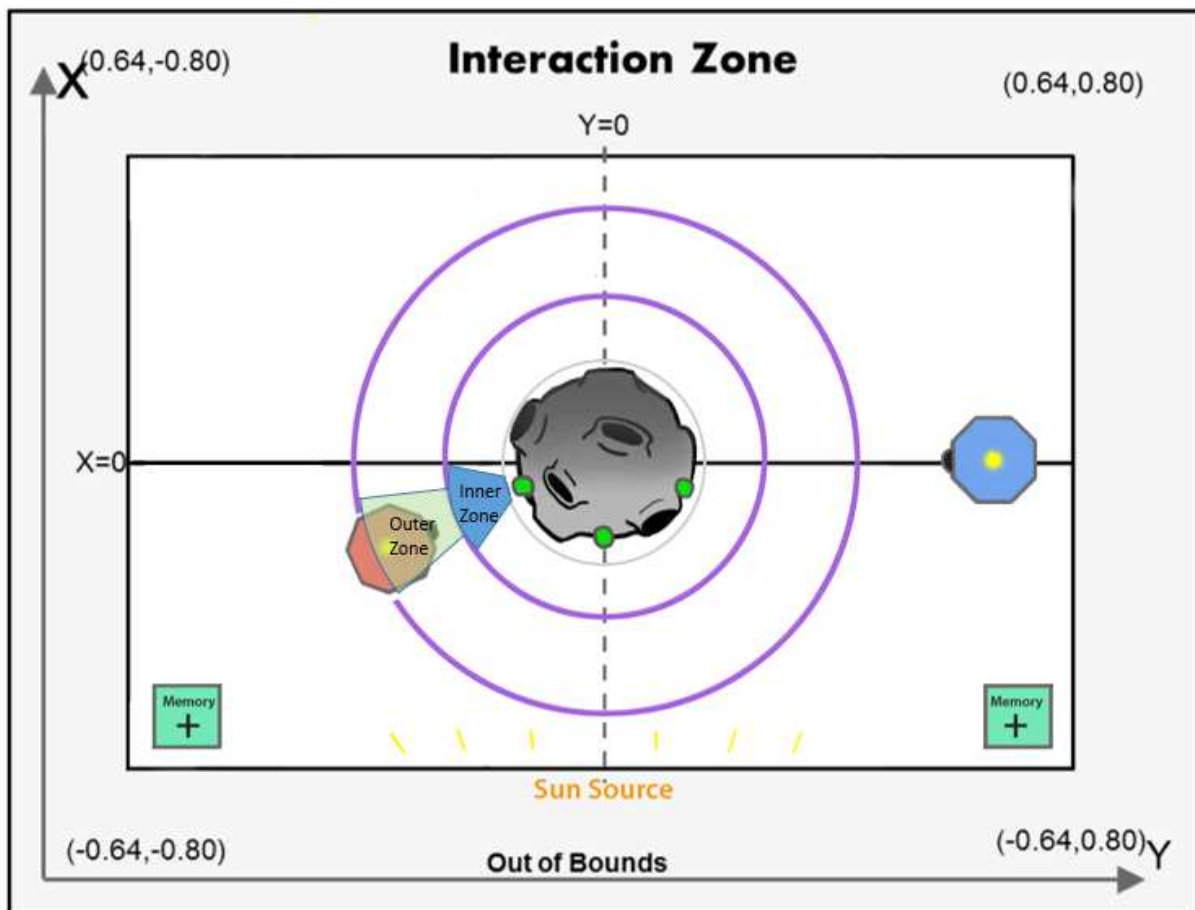


Diagram not to scale

Figure 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 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).

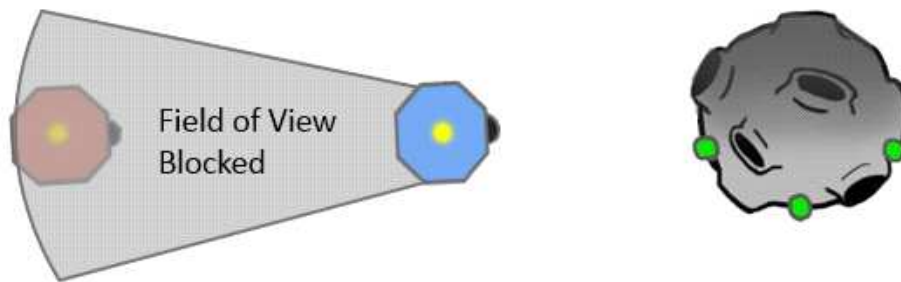


Diagram not to scale

Figure 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 Coordinates to locate the POI in the 3D game (see Tutorials for an explanation of how to use Spherical Coordinates).

• 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 Memory Upgrade Pack Locations

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

To collect Memory Upgrade packs see section 1.5 ITEM COLLECTION.

- **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.

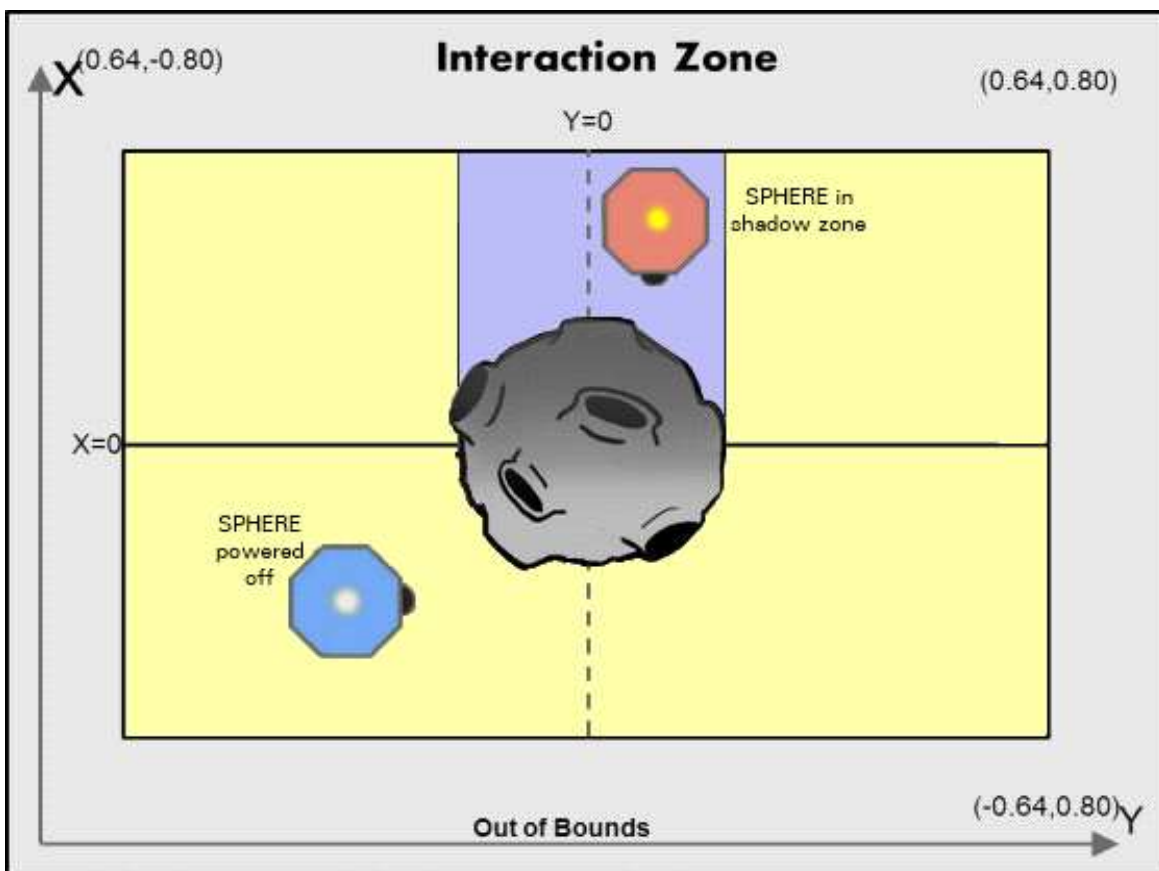


Diagram not to scale Fig

ure Solar Flare

- **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

- **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.

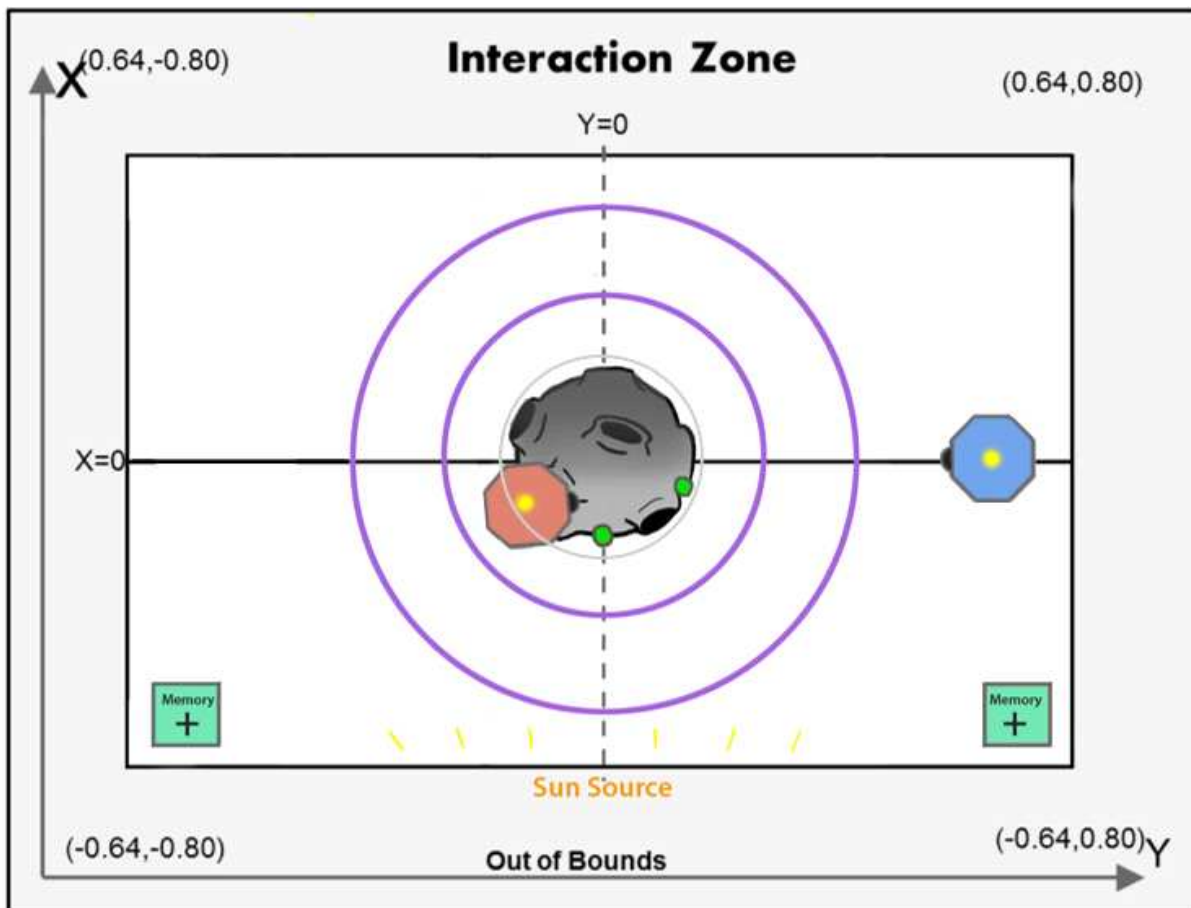


Diagram not to scale

Figure 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.

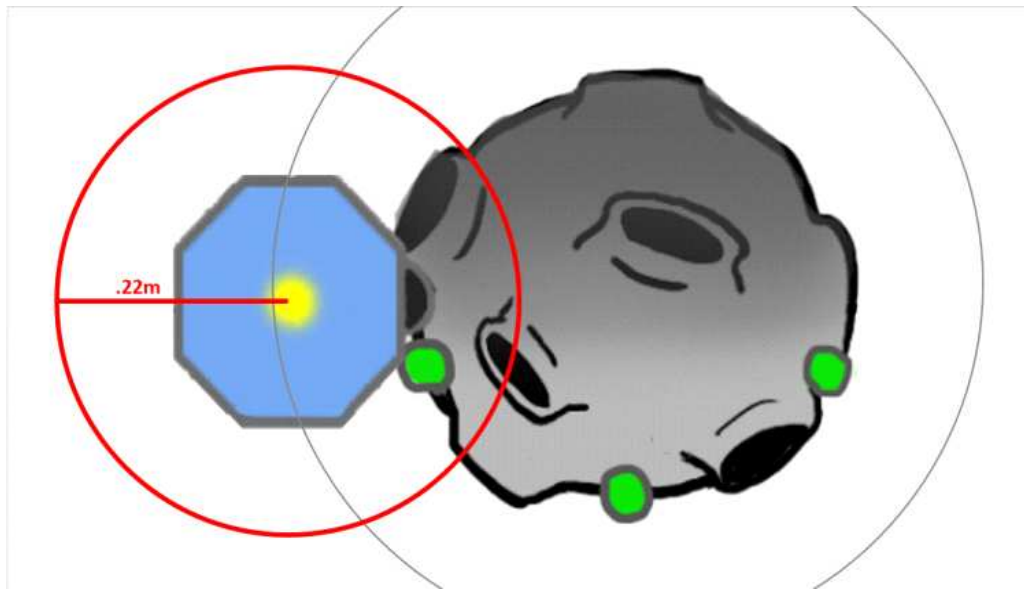


Diagram not to scale

Figure : 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.

- **End of Game**

The game ends when time runs out.

- **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 Memory Upgrade Pack Locations (repeated)

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

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°. Do not attempt to rotate faster than 80°/s.

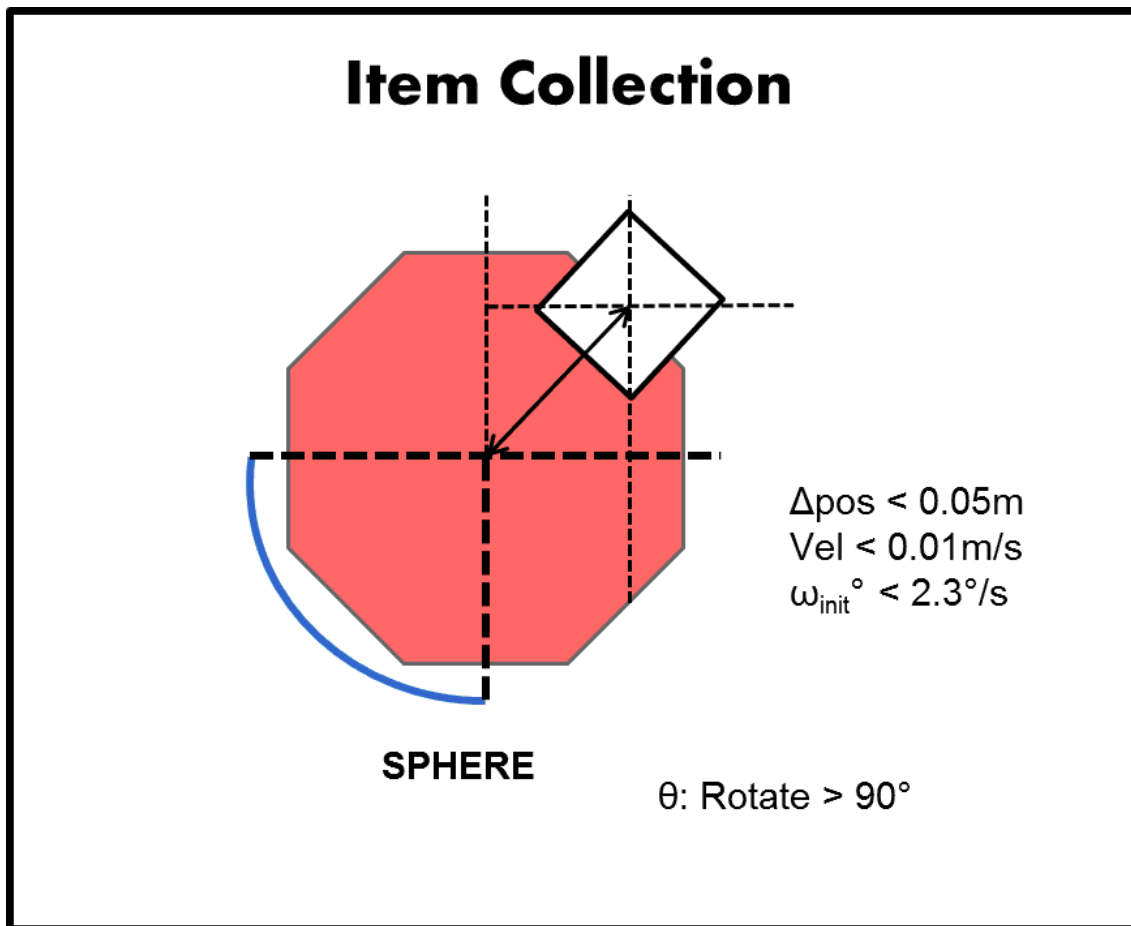


Figure Maneuver to Collect Item

- **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.

- **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 Point Values

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

Solar Flare: Points deducted per second: SPHERE un-powered, outside shadow zone	-0.5
Solar Flare: Points deducted per second: SPHERE inside shadow zone	0
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)

• Tournament

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

Table Tournament Key Dates

Date (2014)	Event
Sep 24	Kick-off, at Politecnico di Torino
Until Dec 5	Registration open to the first 50 teams, by submitting application document as indicated in the Web Page
Nov 1 to Nov 29	3D Warmup
Nov 30 to Dec 19	3D Competition; at the end 6 finalist teams will be selected
Jan 16, 2015	Final code submission (only the 6 finalists); no leaderbord for this code
mid-Jan 2015 (to be confirmed)	National Virtual Finals (in Torino)

The rankings in each competition are determined by a *Leaderboard*, described below. At the end of the 3D Competition an elimination round takes place, with the top 6 ranking teams moving to the Virtual Final Competition to be held in Torino in January 2015 (to be confirmed).

• 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.

• Rating

(To be updated to reflect current leaderboard documentation)

• Playing Matches

(To be updated to reflect current leaderboard documentation)

• Competition Submissions

(To be updated to reflect current leaderboard documentation)

• 3D Simulation Competition

All teams that complete a valid registration 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.

• Virtual Final Competition

The top 6 teams on the leaderboard at the end of 3D competition will advance to the Virtual Final Competition.

The Virtual Finals will take place in real-time simulations in Torino around the second half of January

- **Virtual Final Competition Format**

The teams will be divided into 2 brackets. All teams ranked with odd numbers will participate in Bracket A; all teams ranked with even numbers will participate in bracket B, as shown in Figure 11.

Bracket A	Bracket B
Team Ranks	Team Ranks
1, 3, 5	2, 4, 6

Figure : Division of Teams between Brackets

Each bracket will play 3 matches in round-robin style: team 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:

- The team with the most wins advances
- If teams are tied for wins, the team with the highest total score advances
- If scores are tied, remaining fuel will be used to break the tie

The winning team from each bracket will play a single match to determine the Italian Zero Robotics Champion. The losing team will be awarded 2nd place.

Third and fourth place will be awarded based on the individual scores of the second team of each bracket

- **Scoring Matches**

TBD

- **Season Rules**

- ***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 / Italian Organizing Committee) 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.

- ***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.

- **ZR User API**

- ***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

- ***CoronaSPHERES API Reference***

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

Name	Description
<code>int getNextFlare()</code>	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.
<code>int getMemoryFilled()</code>	Returns number of valid pictures (taken from the right distance and angle from the poi) currently saved in camera.
<code>int getMemorySize()</code>	Returns current limit of picture storage
<code>void takePic(int poiID)</code>	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
<code>void uploadPic()</code>	Uploads pictures from satellite, disables camera for 3 seconds
<code>int numActivePOIs()</code>	Returns number of POI visible
<code>void getPOILoc(float pos[3], int id)</code>	Returns location of each visible POI. Visible POIs are assigned ID # 0, 1 or 2
<code>float getScore()</code>	Returns player's score
<code>float getOtherScore()</code>	Returns opponent's score
<code>void turnOff()</code>	Turns off satellite to protect from solar flare. Satellite will drift.
<code>void turnOn()</code>	Begins process to turn satellite on. Takes 5 seconds. (unless game just beginning)
<code>float getFuelRemaining()</code>	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

• Lists of Figures and Tables

• **List of Figures**

Figure 1 Game Overview	5
Figure 2 Picture Taking Zones	8
Figure 3 Blocking Opponent's Field of View	9
Figure 4 Solar Flare	11
Figure 5 Asteroid Collision	12
Figure 6: Asteroid Collision Close-up.....	13
Figure 7 Maneuver to Collect Item.....	14
Figure 8: Division of Teams between Draft Events	17
Figure 11: Division of Teams between Brackets.....	19

• **List of Tables**

Table 1 Interaction Zone Dimensions	4
Table 2 Zone Radii Positions	4
Table 3 Shadow Zone Dimensions	4
Table 4 Fuel Allocation	6
Table 5 Inter-satellite communications bandwidth.....	6
Table 6 SPHERES Satellite Deployment Locations	6
Table 7 Number of Points of Interest	7
Table 8 Asteroid and SPHERES Measurements	7
Table 9 Max Angle of Zones (in radians).....	7
Table 10 Picture Values in Inner and Outer Zones.....	8
Table 11 Memory Upgrade Pack Locations	10
Table 12 Memory Upgrade Pack Locations (repeated).....	13
Table 13 Point Values	15
Table 14 Tournament Key Dates.....	15

- **Revision History**

Revision	Date	Changes	By
2.2	10/26/14	Originated from rev 2.2 of USA+ESA Reflect changes made to algorithm for 3D POI locations, clarifies impacts with the asteroid, updates tournament schedule, corrects typos	leonardo.reyneri@polito.it
2.3	11/19/2014	Changed deadlines for code submissions	leonardo.reyneri@polito.it