**4 GAME TESTING**

All products, games included, must go through rigorous testing to ensure they meet the designer’s intended goals, and satisfy any other conditions applied upon them. Planet Hope went through several types of testing. The initial testing was technological in nature, and did not involve any playing by developer or test subject. Such testing included ensuring that moons properly orbited planets, that objects would gravitate towards the planet, and that objects that are supposed to stay upright do in fact stay upright. This testing was performed simply by ensuring all scripts compiled properly, and then observing if the object behaved as expected when running the simulation.

None of the objects behaved as expected during the first simulation run. For example, the moon crashed right into the planet, balls would fall to the planet but then vibrate uncontrollably, and if an upright object was facing towards the x-axis it would vibrate out of existence. Once these technological problems were fixed and the basic game engine was operating properly, we moved onto playtesting.

**4.1 Playtesting**

Playtesting is testing the game by actually controlling the player character and playing some part of the game to test its functionality. The simplest form of this is developer playtesting, where one or more developers plays the game as it’s being built. At this stage the game was in the alpha testing stage.



Fig 1. The developers of Planet Hope playtesting the game.

From left to right: Paul Ostlund, Shahd Saeed, and Jeme Jane Ogun

With the powerful and user-friendly development tool Unity, the developers were able to test every change made to the game, as it was made, with incredible ease. For example, every change made to the player’s speed was tested by the developers to see if they would lose control or if the game became too slow to be fun. Every cluster or trail of gems, Planet Hope’s primary collectable object, that was laid down was immediately tested to ensure that they were properly collectable, and that they didn’t clip into the ground as they cycle through their animation. Every static environment object was tested to ensure that the player couldn’t clip through them or climb them if they’re too steep. In all cases, this developer playtesting was absolutely essential in finding bugs and improving gameplay, as very few things ever worked correctly when initially implemented. It was routine for the developer playtester to walk through objects that should be solid, or spot gems dipping into and out of the solid ground.

One common cause of malfunctions was the improper use of the Update() function vs the FixedUpdate() function. All code that affected physics needed to be placed in the FixedUpdate() function, and all code that took player input needed to be in the Update() function. This is because unity performed physics updates and checks after the fixed update. So, if the physics code was in the regular update function, there would be a lag between the object’s physics and the physics of the rest of the world, causing glitches. Further, the FixedUpdate() function runs at a fixed but slower rate than the regular Update() function. So, if any code that reads player input was in the fixed update, there would be frames where the computer would not respond to player input.

One of the most important bugs found during playtesting was a glitch where a player could get stuck immobile in a v shaped pit. The cause of this was the way our control script evaluated slopes. The script would look at the normal vector for each point of contact and compare it to the local upward direction to get the slope angle. If any contact had a low enough slope, the character would be considered ‘grounded’ and could walk about freely. If no contacts had an appropriate angle, the character would simply slide down under the influence of gravity. Thus, if the character were in a steep v, both sides would have too steep a slope and the character would not be considered grounded even when perfectly still, and firmly on the ground. The solution was to have the script also evaluate the average normal vector of the contacts, and if the corresponding slope was within acceptable tolerances, the character would be considered ‘grounded’. This solved the problem. Without the rigorous playtesting of developers, a bug like this could have made it through to the end product and ruined the game. This underscores the importance of proper playtesting.

**4.2 Informal Individual Testing**

Informal individual testing involves causally providing the game for someone outside the development team to play, and informally examining how they play and what they do. The most important thing for the developers to remember when performing informal individual tests during the beta testing stage, is to not give the player any more information than they would have on their own. If a developer is standing by telling the player what to do, or even just hinting at what to do, it will completely taint the playtest and ruin any data the developers may have otherwise gotten from the test. For example, one of the most important things for the player to do in Planet Hope is to collect gems. It is of great interest to the developers to know whether or not players will understand that they’re supposed to go collecting the gems. So, if a developer even asks the player if they see the gems while they’re playing, this could alter their behaviour and make the player seek out the gems when they otherwise wouldn’t. Or alternatively such hints may make a more stubborn playtester less likely to collect gems than they otherwise would have been. Either way, even a small mention of gems could completely alter the course of the playtest and thus ruin any data that otherwise could have been collected. It is important to the developers that the playtesters think out loud and share their experiences as they have them.

**4.3 Formal Group Testing**

After the informal individual testing, the testing process would continue onto the formal group testing stage. Here we would have external investigators get several playtesters to each individually play Planet Hope for about a half an hour, and then fill out a questionnaire.

When designing survey questions, the researcher must always be careful to word the questions as neutrally as possible so as to not lead the player’s answers. For example, in Planet Hope, the developers intended for the player to feel a sense of fear as they are being chased by the monster. If we ask the playtester if they felt afraid during this part, they may reason that they aught to have been afraid, and so answer that they were indeed afraid. Instead, it would be more useful to simply ask how they felt during this part of the game, so as to mitigate this effect.

It is also important to directly ask for negative feedback, as testers are typically reluctant to give it, and it is the most useful feedback to get. For example, when playing Planet Hope, playtesters may find it tedious to make their way back to the ship after collecting gems, but may be reluctant to share this as it may hurt the developer’s feelings. However, information like this is of utmost importance to the developers, as tedium is one of the most important things for developers to remove from their game. So instead of asking if the player found anything tedious, it’s better to directly ask the player to state what they found most tedious.

Lastly, it is important to ask the player what they enjoyed the most about the game, as this is what the developers should focus on expanding on.

Some questions we could have had investigators ask playtesters during formal group testing include the following:

Post Playtest Survey Questionnaire:

1. How would you describe your overall experience playing the game?
2. Did you feel like you understood how to control the character?
3. Did you feel like you knew what to do?
4. How did it feel collecting gems?
5. How did it feel being chased by the monster?
6. How did you feel about the size of the planets?
7. What could be changed to make your experience more enjoyable?
8. What did you find most tedious?
9. What was your least favourite part of the game?
10. What was your favourite part of the game?
11. How did your view of the game change throughout the time you were playing?
12. Did you ever feel confused? If so, where?

**4.4 Formal Individual Testing**

The final stage of playtesting is formal individual testing, where we would have investigators examine an individual playtester’s experience in detail, observing them as they play, and interviewing them afterwards.

**5 GAME BALANCING**

The most important thing for a game to be, is fun. Players have fun playing games when they are met by an appropriately difficult challenge. If the challenge is too difficult the player becomes frustrated and doesn’t have fun, if the challenge is too easy the player becomes bored and doesn’t have fun. As such, balancing the difficulty level of a game is of the utmost importance for the developers. Balancing a game is both an art and a science, involving copious amounts of raw data, as well as an intuitive knack for getting the game to ‘feel right’.

**5.1 Ability Balancing**

In planet Hope, the primary ability that needed to be balanced was player speed. First and foremost, it needed to be tuned for a smooth gem collecting experience. If the player was too fast, they wouldn’t have the accuracy needed to properly collect the gems, and would often have to circle back to collect a missed gem. If the player was too slow, gem collecting would become tedious. Poor controls and tedious gameplay are some of the worst flaws a game can have, so it was very important to properly balance this parameter.

Player speed was balanced by performing an experiment. The developer performing the playtest would have to collect the gems in a specific area in the game as fast as they could. The player speed was varied for each trial, and we counted the number of times the playtester missed a gem while trying to collect it.

Our hypothesis was that below a certain speed, the critical speed, the number of gems missed would be relatively constant, and above that critical speed, the number of missed gems would increase with increasing speed. The idea being that at any speed below the critical speed, the player should have no trouble collecting the gems, and only miss the occasional gem. Whereas above the critical speed, the player will begin to miss more and more gems as the speed increases and the player loses more and more control over the character. Thus, we expect that this critical speed will be the highest speed that a player can comfortably control.

So, to balance the player speed, its final value was then set to the highest possible speed in the constant region, as we would expect this to be the critical speed. The results of the experiment are detailed in Table 1 and Figure 2.

Table 1. Results of the Player Speed Test Experiment.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Planet | Location | Player |  | Gems Missed |  |  |
| Name | Number | Speed | Trial1 | Trial 2 | Trial 3 | Average |
| Jardin | 1 | 4 | 1 | 0 | 0 | 0.33 |
|  |  | 5 | 2 | 1 | 0 | 1.00 |
|  |  | 6 | 3 | 3 | 2 | 2.67 |
|  |  | 7 | 6 | 2 | 5 | 4.33 |
|  | 2 | 4 | 0 | 0 | 1 | 0.33 |
|  |  | 5 | 0 | 0 | 1 | 0.33 |
|  |  | 6 | 0 | 1 | 0 | 0.33 |
|  |  | 7 | 0 | 1 | 0 | 0.33 |
| Regolith | 1 | 4 | 0 | 0 | 0 | 0.00 |
|  |  | 5 | 0 | 0 | 0 | 0.00 |
|  |  | 6 | 0 | 1 | 1 | 0.67 |
|  |  | 7 | 1 | 1 | 1 | 1.00 |
|  | 2 | 4 | 3 | 1 | 2 | 2.00 |
|  |  | 5 | 2 | 2 | 2 | 2.00 |
|  |  | 6 | 0 | 4 | 3 | 2.33 |
|  |  | 7 | 4 | 4 | 5 | 4.33 |
| Graben | 1 | 4 | 1 | 3 | 0 | 1.33 |
|  |  | 5 | 3 | 2 | 1 | 2.00 |
|  |  | 6 | 3 | 5 | 2 | 3.33 |
|  |  | 7 | 4 | 3 | 3 | 3.33 |
|  | 2 | 4 | 2 | 0 | 1 | 1.00 |
|  |  | 5 | 2 | 2 | 1 | 1.67 |
|  |  | 6 | 1 | 1 | 1 | 1.00 |
|  |  | 7 | 3 | 2 | 3 | 2.67 |

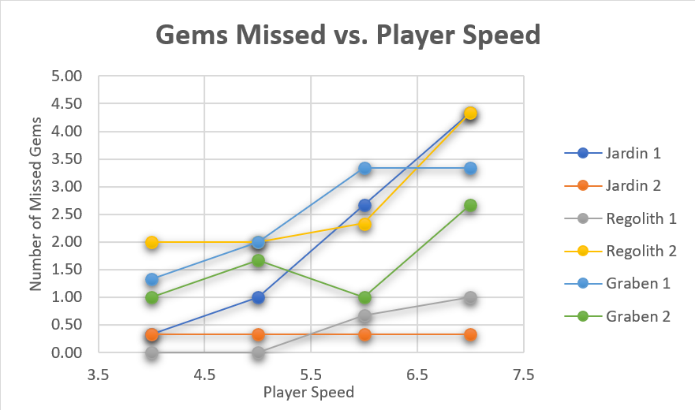


Fig 2. Graph of missed gems vs. player speed from the player speed test experiments.

Missed gems is used as a proxy value for difficulty of control. We observe that there

Is an overall upward trend in the data, and that there is generally a shallower slope at

lower speeds and a higher slope at higher speeds. This is consistent with the

developer’s hypothesis.

Based on the data, it was estimated that the critical speed is five units per second. With more time available, it would have been possible to find a more refined value for the player speed. For example, it may be that 5.1 or 4.9 gives more optimal performance than 5.0 exactly. However, the benefit of further refinement would have been small compared to the value of the time spent performing the experiment.

The next factor that needed balancing was the monster speed. To balance it independently of the player speed balancing, we examined the ratio of the player speed to the monster speed instead of examining the monster speed directly. The experiment we performed was to start the player character a fixed distance from the monster while it’s chasing them, and simply back away at the speed prescribed by the experimental trial. If the player advanced fast enough that the monster receded behind the horizon within thirty seconds, the outcome was recorded as ‘escape’. If the monster reached the player, the outcome was recorded as ‘catch’. If the monster receded but did not make it beyond the horizon within thirty seconds, the outcome was recorded as ‘evade’. And if there was no noticeable progression or recession of the monster, the outcome was recorded as ‘stalemate’.

The idea was to balance the player/monster speed ratio so as to satisfy two factors. Firstly, there should be no planet and difficulty level combination where the monster can catch the player while they are sprinting. We didn’t want to have a scenario where the player had no chance of success. It would be acceptable if, on the hardest level, the player could only ‘evade’ while sprinting rather than fully ‘escape’. Secondly, we wanted it so that there should be no planet and difficulty level combination where the player escapes the monster while not sprinting. The player should at best be able to evade the monster while not sprinting, and this should only be on the easiest planet and on the easiest difficulty. On harder planets and difficulties, the player should be caught when not sprinting. This way, as the player progresses throughout the game, the difficulty level of the game should steadily increase from ‘easy but not trivial’, to ‘hard but not impossible’. The results of the experiment are detailed in Table 2.

Table 2. Results of the Player/Monster speed ratio experiments.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Planet |  | Player/Monster | Sprint |  |  | Planet |  | Player/Monster | Sprint |  |
| Name | Difficulty | Speed Ratio | Active | Outcome |  | Name | Difficulty | Speed Ratio | Active | Outcome |
| Jardin | easy | 1.00 | yes | escape |  | Graben | easy | 0.67 | yes | stalemate |
|  |  | 1.00 | no | stalemate |  |  |  | 0.67 | no | catch |
|  |  | 1.25 | yes | escape |  |  |  | 0.83 | yes | escape |
|  |  | 1.25 | no | evade |  |  |  | 0.83 | no | catch |
|  |  | 1.50 | yes | escape |  |  |  | 1.00 | yes | escape |
|  |  | 1.50 | no | escape |  |  |  | 1.00 | no | catch |
|  | normal | 1.00 | yes | evade |  |  | normal | 0.67 | yes | catch |
|  |  | 1.00 | no | catch |  |  |  | 0.67 | no | catch |
|  |  | 1.25 | yes | evade |  |  |  | 0.83 | yes | evade |
|  |  | 1.25 | no | stalemate |  |  |  | 0.83 | no | catch |
|  |  | 1.50 | yes | escape |  |  |  | 1.00 | yes | escape |
|  |  | 1.50 | no | evade |  |  |  | 1.00 | no | catch |
|  | hard | 1.00 | yes | stalemate |  |  | hard | 0.67 | yes | catch |
|  |  | 1.00 | no | catch |  |  |  | 0.67 | no | catch |
|  |  | 1.25 | yes | evade |  |  |  | 0.83 | yes | catch |
|  |  | 1.25 | no | catch |  |  |  | 0.83 | no | catch |
|  |  | 1.50 | yes | escape |  |  |  | 1.00 | yes | escape |
|  |  | 1.50 | no | stalemate |  |  |  | 1.00 | no | catch |
| Regolith | easy | 1.00 | yes | evade |  |  |  | 1.17 | yes | escape |
|  |  | 1.00 | no | catch |  |  |  | 1.17 | no | catch |
|  |  | 1.00 | yes | evade |  |  |  |  |  |  |
|  |  | 1.00 | no | stalemate |  |  |  |  |  |  |
|  |  | 1.20 | yes | evade |  |  |  |  |  |  |
|  |  | 1.20 | no | stalemate |  |  |  |  |  |  |
|  | normal | 0.80 | yes | stalemate |  |  |  |  |  |  |
|  |  | 0.80 | no | catch |  |  |  |  |  |  |
|  |  | 1.00 | yes | evade |  |  |  |  |  |  |
|  |  | 1.00 | no | catch |  |  |  |  |  |  |
|  |  | 1.20 | yes | evade |  |  |  |  |  |  |
|  |  | 1.20 | no | stalemate |  |  |  |  |  |  |
|  | hard | 0.80 | yes | catch |  |  |  |  |  |  |
|  |  | 0.80 | no | catch |  |  |  |  |  |  |
|  |  | 1.00 | yes | evade |  |  |  |  |  |  |
|  |  | 1.00 | no | catch |  |  |  |  |  |  |
|  |  | 1.20 | yes | evade |  |  |  |  |  |  |
|  |  | 1.20 | no | catch |  |  |  |  |  |  |

As the levels progress, the Monster is required to move at a faster rate, this is the reason for the seemingly inconsistent Player/Monster speed ratio values used throughout the experiments.

Using the data, the developers determined that the player/monster speed ratio aught to be 1.25=5/4 on Planet Jardin (the easiest level), 1.00 on Planet Regolith, and 0.83=5/6 on Planet Graben. These values satisfy both of the above requirements, as well as present a difficulty level that steadily increases.

Some important game parameters that needed to be balanced were the gravity levels of each planet. The developers had a particular vision for how the player character should look and feel while jumping under the influence of gravity on each planet. We decided to give the first planet earth-like gravity, so as to set a familiar benchmark. We wanted the second planet to have higher gravity and the third to have lower gravity. This way the player could experience a nice range of planet gravities throughout the game. However, if the gravity was too strong, the player would not be able to properly jump and collect gems. If the gravity was not strong enough, the player character became very difficult to control, and could even sometimes jump into orbit, essentially making it impossible for the player to continue the game. With extensive tinkering with the gravity parameters, acceptable values were found for the high gravity world and the low gravity world. This process was done entirely by feel.

An unimplemented feature that would have needed extensive balancing would have been the weapons that would have been available to fight the minor enemies that would have been present throughout the various levels. It would be essential that any weapons are capable of defeating enemies only when the player has used sufficient skill to warrant it. For example, if the player were immediately given a gun that shoots homing rounds that instantly defeat all enemies, the player would have no challenge and thus become bored. On the other hand, if the player gets a gun that requires the aim of a trained sharpshooter, has only one bullet, and deals only one damage, they will likely become extremely frustrated. Thus, it would be important to use the methods used to balance player speed and player/monster speed ratio, to also balance these weapons.

**5.2 Positive and Negative Feedback System**

Feedback systems are mechanisms whereby the player’s chances of winning are either enhanced or diminished by their progress in the game. Feedback systems can be positive where the player’s successes lead to a higher chance of further success, or negative where the player’s successes lead to a lower chance of further successes. Feedback systems can either be inscribed by the developers or they can emerge dynamically as the game is played.

In Planet Hope the most obvious example of a feedback system is the negative feedback caused by collecting gems. As a player collects gems they disappear from the map. It may be rewarding to the player to see their work collecting gems manifest as a cleared-out level, but as there are fewer and fewer gems, it becomes harder and harder to find the few remaining gems, and thus harder to progress in the game. This is a clear example of a negative feedback mechanism, as the game becomes more challenging as the player has more success.

An unimplemented positive feedback mechanism would have been the effects of collecting the gear items that were to be present on the various levels. For example, it would have been that on the first level of the game there would be three jetpack pieces. If the player collected all three pieces they would be given limited jump boost abilities. These abilities would give the player a major advantage when it came time to avoid the monster. Thus, if the player got a quick and efficient start, getting the jetpack pieces early, they would then have a major advantage later when the monster came. Whereas a player who procrastinated or even neglected to get the jetpack pieces would be at a relative disadvantage evading the monster later on.

**6 PLAYER GUIDANCE**

In any game it is of the utmost importance that the player knows what they are supposed to be doing, and where they are supposed to be going. In Planet Hope, the player is supposed to travel to a level, collect gems in the level to unlock more levels, avoid the hazardous monster, and ultimately unlock Planet Hope itself. The player was directed towards these goals using the three methods of direct guidance, indirect guidance, and sequencing.

**6.1 Direct Guidance**

Direct guidance is any guidance where the player is given explicit instructions on how to play or what to do. In Planet Hope direct guidance is used at the start of the game to teach players the basic controls and present them with the basic mission. Directly in the starting view of the player is a poster that outlines the controls. One such control is the ability to ‘interact’ with ‘the ship’ and ‘control panels’ by walking up to said object and pressing the Q key. The poster even shows images of the player character touching ‘the ship’ and a ‘control panel’ to show exactly what is meant by these terms. This poster thus gives explicit instructions on how to play the game. To ensure that it doesn’t overwhelm or bore the player, it is just one poster right at the beginning that can be safely ignored if the player already knows how to play the game. Since the controls are very simple and are all used right away, there is no major drawback to presenting them at the start of the game.

If the player decides to walk up to this poster and press Q, it will change to another poster giving additional direct guidance in the form of a call to arms to the player. The poster will now explain that the player should go to the planets and collect gems, that unlocking planet hope is the ultimate goal of the game, and that there is a monster in the levels that the player should beware of. This poster is also completely optional so as to not inconvenience a player that already knows what they’re doing.

Inside of the ship there is additional direct guidance in the form of the control panels that allow the player to access the planets. Each one shows exactly which planet it leads to, as well as the relative difficulty level of the planet. If the planet is still locked, the control panel shows exactly how many gems from each planet are needed to unlock it. While much of the game should be relatively straight-forward to an experienced player of games, it would be impossible for the player to know what exactly is needed to unlock a planet without this direct guidance.

All of the direct guidance mentioned above is the only direct guidance in the game. It occurs only in the ship scene, the first scene of the game that also serves as the hubworld. Thus, direct guidance is used only scarcely and briefly. However, the information is always available in that location should the player ever want to revisit it. The controls of Planet Hope are very simple, so even if the player doesn’t understand the exact meaning of the control poster, they will still know which buttons are used, and under what conditions. For example, the control poster says “Touch ship or control panels and press Q to interact.” The player may not understand what will happen when they touch a control panel and press Q, but it should be clear to the player that this is something they should try. Then they can directly experience the outcome of that action.

**6.2 Indirect Guidance**

Indirect guidance is any form of guidance used to usher the player in a particular direction, without explicitly stating what the player is intended to do. Planet Hope uses several classic methods of indirect guidance to show the player what they aught to be doing.

Gem collecting is the main short-term goal of the game, therefore it is imperative that players view gem collecting as a favourable thing. Gems were specifically selected as the collection object because humans are naturally drawn to the shiny, strongly coloured, geometric surfaces of gemstones. In Planet Hope the gems were given an emissive material so that they would actually glow, standing out strongly against a dark background. And they were animated to constantly rotate and bob up and down, constantly sparkling as they catch the light at different angles. This made them extra eye-catching, and ideally makes the player far more likely to notice them, and approach them. Furthermore, on each planet the gems were given a color complimentary to the primary colour of the ground so as to make them stand out even more.

When the player character touches a gem, it is collected. In addition to the gem being removed from the scene, a pleasant C major chord is played. This immediately tells the player that this was a positive action, so that even if they haven’t yet figured out that they’re supposed to be collecting gems, they should now at least see it as a positive thing to do. Further, this sound acts as a reinforcing encouragement to the player, to encourage them to find more gems. This is a classic form of indirect guidance.

Similarly, the monster constantly plays an unsettling tune. When the player is far from the monster, they cannot hear this tune, but as the monster gets closer and closer, the music gets louder and louder. By using minor chords and tritones, the tune naturally signals to the human brain that something is somehow discordant and should be avoided. This is a use of indirect guidance to alert the player that the monster is to be avoided.

When it comes to directing the player where to go, we wanted to direct the player to explore as much of the planets we designed as possible. Specifically, we wanted to direct the players towards the special features on the planets. To accomplish this, we placed gems in clusters and rings at the various locations of interest throughout the planets, as well as in trails that lead between these locations. The idea being that this would indirectly lead gem seeking players towards landmarks. The gems clustered at the location would act as a reward for visiting the location, and the trails would indirectly guide the player to new locations.

**6.3 Sequencing**

Sequencing is the process of presenting new information to the player in a gradual way, building up from the basic ideas in an isolated introduction, then expanding on these ideas, adding danger, and increasing the difficulty, before finally integrating the information into the rest of the game.

Planet Hope is a relatively simple game, and so uses sequencing to teach the player how to play the overall game, rather than using it multiple times, once for each skill.

At the beginning of the game, the player starts in the ship, where they are given instructions on how to control the game. In this isolated introductory setting, there is no danger or risk, it is simply a small area where the player can practice the controls without any cost. To move on from this area, the player must demonstrate that they understand the controls by walking up to a control panel, and pressing Q. For a player familiar with the game, this introductory segment can be completed in a literal second, thus preventing boredom when replaying the game. For the new player, they can spend as long as they want here until they get the hang of the basic controls.

The first planet the player visits is a low difficulty world with an earth-like gravity. Even on the hardest difficulty the player has a full minute before the monster is released, and an even longer time on easier difficulties. During this part of the game the gameplay is expanded beyond basic controls to include gem collecting, as well as introducing a more varied environment to traverse. As the monster has not yet been released, there is still no direct danger or penalty.

However, as soon as the time limit is up, the monster will begin chasing the player. In the first level the monster is slow, and can be evaded by the player even without sprinting, thus keeping the danger relatively low. Furthermore, if the player doesn’t yet understand that they’re supposed to avoid the monster, when the player dies and has to start over, the loss will be relatively low, since at this point in the game players should only have collected a few gems, rather than having unlocked multiple planets.

Once the player unlocks a second planet, they will face the same challenge as before but with an increased difficulty. On each successive planet, the monster is faster, and the gems are less abundant and harder to obtain. The first planet is essentially a tutorial level, whereas the others are challenge levels. Once the player unlocks a third planet they will have essentially shown mastery over the gameplay. They can then integrate their new skill of collecting gems with the overall gameplay goal of unlocking Planet Hope.