Test Report: Breaking Effect

Marshall Xiaoye Ma

December 15, 2017

1 Revision History

Date	Version	Notes
2017-12-08	1.0	New document

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/MaXiaoye/cas741/blob/master/Doc/SRS/SRS.pdf

Contents

List of Tables

List of Figures

This document provides a report of the results of the testing performed on the Breaking Effect application. Both system testing and unit testing are covered. Traceability between testing and both requirements and modules is given in the final section. The tests that were implentmented in this report follow the Test Plan document. Please refer to this document for further information at `https://github.com/MaXiaoye/cas741/blob/master/Doc/TestPlan/TestPlan.pdf`.

3 Functional Requirements Evaluation

3.1 Getting input from user

This test suite is designed to ensure the program can handle both valid and invalid inputs from user that cover R1 and R2. Inputs provided by user include target object, explosion level (also known as initial momentum after explosion) and coefficient of friction on the ground.

Assumption

- In all test cases, $(X, Y, Z) = (a, b, c)$ means a target object with position (a, b, c). The target object for all test cases named "targetObj" consists of 32 cubes with size 1 x 1 x 1. It locates under projectDir/Assets/prefab/targetObj
- $\approx$ consider equal with a delta of 0.01
- All test codes are in dir /src/Assets/script/BETest.cs

1. testInput

   Type: Functional, Dynamic, Automated

   Initial State: New session

   Input: Please refer to table below

   Expect output: Please refer to table below.

   How test will be performed: Automated unit test

   Result: Pass

Name	**In**	**Expect output**	**Result**
CorrectIn	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.05$	-	Pass
NoMu	$E = 5$, $(X, Y, Z) = (0, 0, 0)$	InvalidMuException	Pass
NoObj	$E = 5$, $\mu_k = 0.05$	NoObjException	Pass
NoExpLv	$(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.05$	NoELvException	Pass
NonNumE	$E = a$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.05$	InvalidELvException	Pass
NonNumMu	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = a$	InvalidMuException	Pass
OutScoE	$E = -1$ or $E = 11$,$(X, Y, Z) = (0, 0, 0)$,$\mu_k = 0.05$	InvalidELvException	Pass
OutScoC	$E = 5$, $(X, Y, Z) = (0, 99999, 0)$, $\mu_k = 0.05$	InvalidCoorYException	Pass
OutScoM	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 99$	InvalidMuException	Pass

3.2 Acquire pieces

This section covers R3, R4 in SRS that class PieceObj is defined as required. Instance of PieceObj can be created by acquiring each piece in the scene correctly.There is an assumption for this section that all tests in 3.1 are passed.

Acquire all pieces

1. pieceInit

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$

   Expect output: A list of piece objects: pieceObj that pieceObj.length = 32. Piece shape is random that is decided by cutting method.

   How test will be performed: Automated unit test.

   Result: Pass

Calculation of angle between initial speed and horizontal(θ_1) as well as the angle between x axiom and projection on horizontal of initial speed(θ_2)

1. testAngles1

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$, $(x_n, y_n, z_n) = (1.5, 2.0, 0.5)$;

   Expect output: $pieceObj[0].\theta_1 = arctan\frac{y_n - Y}{\sqrt{(z_n-Z)^2+(x_n-X)^2}} \approx 0.90$ and $pieceObj[0].\theta_2 = arctan\frac{x_n-X}{z_n-Z} \approx 1.25$

   How test will be performed: Automated unit test.

   Result: Pass

2. testAngles2

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$, $(1.5, 2.0, 1.5)$

   Expect output: $\theta_1 \approx 0.76$ and $\theta_2 \approx 0.79$

   How test will be performed: Automated unit test.

   Result: Pass

3. testAngles3

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$, $(0, y_n, 0)$ and $0y_n$ can't be zero

   Expect output: $\theta_1 = 90^\circ \approx 1.57$ and $\theta_2 = 90^\circ \approx 1.57$

   How test will be performed: Automated unit test.

   Result: Pass

3.3 Calculation of displacement for each piece

This section covers R5 that displacement of each piece in the air should be calculated correctly. Testing for edge cases and error handling will be covered in Section 5.
There is an assumption for this section that all tests in 3.1, 3.2 are passed.

1. testMotionAir

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$, $\Delta t = 0.02$, $t = 2.86$

   Expect output: $S_x = v_0 \cdot cos\theta_1 \cdot cos\theta_2 \cdot \Delta t \approx 0.59$, $S_z = v_0 \cdot cos\theta_1 \cdot sin\theta_2 \cdot \Delta t \approx 0.20$ and $S_y = (v_0 \cdot sin\theta_1 - gt) \cdot \Delta t - \frac{1}{2}g \cdot \Delta t^2 \approx 0.40$. Above output should be shown both in text in console and in vision on the screen.

   How test will be performed: Automated unit test.

   Result: Pass

2. testMotionGround

   Type: Functional, Dynamic, Automated

   Initial State: New Session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$, $\Delta t = 0.02$, $t = 2.86$

   Expect Output: $a = \mu_k g$, $v_x = v_x^{'} + a\Delta t$, $v_z = v_z^{'} + a\Delta t$, $S_x = v_x \cdot \Delta t - \frac{1}{2}a\Delta t^2 \approx 0.58$, $S_z = v_z \cdot \Delta t - \frac{1}{2}a\Delta t^2 \approx 0.19$

   How test will be performed: Automated unit test.

   Result: Pass

4 Nonfunctional Requirements Evaluation

4.1 Usability

Installation of Unity3D is not in the scope of Breaking Effect's usability test.

1. Participants
   Classmate: Jason (RenJie)
   Other participants: My 2 roommates.

2. Document given
   Readme.doc under project folder.

3. Task for participants
   Run test case successfully by reading Readme.doc individually . Without help from others.

4. How test will be measured and performed
   According to content of Readme.doc and complexity of the program. If all participants can run test case successfully by only reading Readme.doc individually, then Usability of the program is good.

4.2 Performance test

1. testPerformance

   Type: Nonfunctional,Dynamic, Manual

   Initial State: New session

   Input/Condition: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$ The range of the number of pieces is 4 - 64.

   Output/Result: Motion of each piece.

   How test will be performed: Adjust amount of pieces to see the performance that if program can run smoothly in huge amount of pieces.

4.3 Maintainability

1. testMaintain

   Type: Nonfunctional,Dynamic, Manual

   Initial State: New session

   How test will be performed: Extend the program with additional physics effect rebound (not perfectly).

4.4 Reusability

1. testReuse

   Type: Nonfunctional,Dynamic, Manual

   Initial State: New session

   How test will be performed: Combine Breaking Effect with existing Unity3D project.

5 Unit Testing

Unit Test Generator provided by Visual Studio 2017 will be used to implement automated unit testing for this project.

5.1 Unit test for Piece Object Module (M4)

1. ConstructorPieceObj

   Type: Functional, Dynamic, Automated

   Initial State: New session.

   Input: Refer to table below;

   Output: Refer to table below

   How test will be performed: Automated unit testing.

Name	In	Expect Output	Result
UTPieceClass1	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].obj.name="Cube1"	Pass
UTPieceClass2	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].onGround = false	Pass
UTPieceClass3	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].initSpeed = 50	Pass
UTPieceClass4	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].gravity = -9.8	Pass
UTPieceClass5	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].stop = false	Pass
UTPieceClass6	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].speedLastFrameX = 1	Pass
UTPieceClass7	$E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$	pieceObj[0].speedLastFrameZ = 1	Pass

5.2 Unit test for Target Object Module (M8)

1. UTTarObj1

   Type: Functional, Dynamic, Automated

   Initial State: New session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$

   Expect output: targetObj.transform.childCount = 32

   How test will be performed: Automated unit testing.

   Result: Pass

5.3 Unit test for Collision with ground detection Module (M9)

1. UTCollision1

   Type: Functional, Dynamic, Manually

   Initial State: New session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.2$

Expect Output: It is clear to see that pieces drop on the ground.(Don't go through the ground). pieceObj[1].onGround = true

How test will be performed: Manually test by developer

Result: Pass

5.4 Unit test for Output Module (M10)

1. UTOutput1

   Type: Functional, Dynamic, Automated

   Initial State: New session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.05$

   Expect output: Visualization on screen works well.

   How test will be performed: Manual test by developer

   Result: Pass

5.5 Unit test for Camera controlling Module (M11)

1. UTOutput1

   Type: Functional, Dynamic, Automated

   Initial State: New session.

   Input: $E = 5$, $(X, Y, Z) = (0, 0, 0)$, $\mu_k = 0.05$

   Expect output: User can control camera moving by keyboard and mouse.

   How test will be performed: Manual test by developer

   Result: Pass

6 Changes Due to Testing

- Put codes run every frame from Update() function to FixedUpdate() function

- Add function GetCenter() and GetExplosionPoint() to get accurate explosion point that is bottom center of target center instead of position attribute defined by Unity3D.
- Add some functions to get static value due to testing requirement.
- Add codes to check if speed of a piece object reaches zero on the ground. If it reaches zero then stop the object immediately to prevent the speed increases in opposite direction.
- Add camera controlling module to provide free camera controlling when program running to improve experience and make testing easier.
- Separate collision detecting module to a new file and improve it so that BreakingEffect support multiple target objects.

7 Automated Testing

Automated Testing are mentioned in test cases with Automated type in previous sections.

8 Trace to Requirements

	R1	R2	R3	R4	R5
3.1	X	X			
3.2			X	X	
3.3					X

Table 1: Traceability Matrix Showing the Connections Between Test Cases and Requirements

9 Trace to Modules

	M3	M4	M5	M6	M8	M9	M10	M11
3.1	X							
3.2			X					
3.3				X				
5.1		X						
5.2					X			
5.3						X		
5.4							X	
5.5								X

Table 2: Traceability Matrix Showing the Connections Between Test Cases and Modules

10 Code Coverage Metrics

References