Breaking Effect

Xiaoye Ma, max58@mcmaster.ca

October 1, 2017

1 Revision History

Date	Version	Notes
2017-10-01	1.0	New document

2 Reference Material

This section records information for easy reference.

2.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
m	length	metre
kg	mass	kilogram
s	time	second
J	energy	Joule

2.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
S	unit of length	displacement
t	s	the time in seconds since the start of the game
v_0	m/s	initial speed
a	m/s^2	acceleration
g	m/s^2	gravity acceleration
ΔE_k	J	variation of kinetic energy
ΔE_p	J	variation of potential energy
W_f	J	work done by kinetic friction
x_n	m	x coordinates of gravity center of piece n
y_n	m	y coordinates of gravity center of piece n
z_n	m	z coordinates of gravity center of piece n
θ_1	degree	angle between initial speed and horizontal
θ_2	degree	angle between x axiom and projection on horizontal of initial speed
S_x	m	displacement on direction of x axiom

S_y	m	displacement on direction of y axiom
S_z	m	displacement on direction of z axiom
μ_k		coefficient of friction

2.3 Abbreviations and Acronyms

symbol	**description**
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
Breaking Effect	Breaking Effect
T	Theoretical Model

Contents

3 Introduction

Because of the development of video games industrial and hardware such as CPU and GPU, there is a higher demand for high level experience in game visualization. Breaking effect plays a more important role in the visualization level of large scale video games. This project simulates the process of 3D objects destruction.

The following section provides an overview of the Software Requirements Specification (SRS) for a breaking effect program. The developed program will be referred to as Breaking Effect (BE). This section explains the purpose of this document, the scope of the system, the characteristics of the intended readers and the organization of the document.

3.1 Purpose of Document

The main purpose of this document is to describe the modeling of breaking effect. The goals and theoretical models used in the breaking effect code are provided, with an emphasis on explicitly identifying assumptions and unambiguous definitions.

3.2 Scope of Requirements

The scope of the requirements is limited to breaking effect of a single 3D object applied by force. Interact force between objects and collision among several objects are not in the scope. Given the appropriate inputs, the code for BE is intended to calculate pieces motion and display the process of target 3D object breaking in vision. The project is implemented in a game engine. 3-D objects and piece generation function are provided by platform.

3.3 Characteristics of Intended Reader

Reviewers of this documentation should have a basic knowledge in physics motion theory and an understanding of differential equations.

3.4 Organization of Document

The organization of this document follows the template for an SRS for scientific computing software proposed by Smith and Lai (2005); Smith et al. (2007). The presentation follows the standard pattern of presenting goals, theories, definitions, and assumptions.

4 General System Description

This section identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

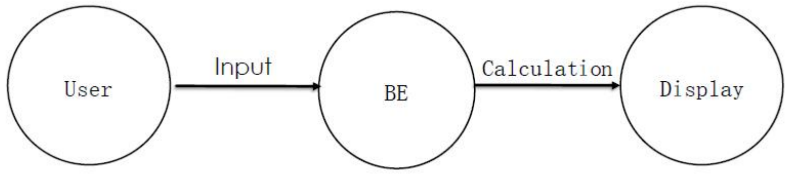

Figure 1

4.1 System Context

- User Responsibilities:
  - Provide inputs and make sure they are in an appropriate range
- Breaking Effect Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number
  - Determine if the inputs satisfy the required physical and software constraints.
  - Calculate the required outputs

4.2 User Characteristics

The end user of Breaking Effect should have a basic understanding of Physics and 3D models.

4.3 System Constraints

There are no system constraints.

5 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

5.1 Problem Description

This project tries to implement running time breaking effect in codes for 3-D models in unity3D without help from any similar plug-in. Including different shapes 3-D objects breaking based on physics and pieces interacting with the momentum provided by the breaking force. The breaking effect program simulates 3-D objects destruction process in vision by implementing scientific computing functions. This project concentrates on calculation while

HCI or GUI are not important parts. Applied force is decided in codes in advance as input and trace of motion is the output after calculation.

5.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- Focus point: The location where explosion happeds
- initial momentum level: Initial momentum of pieces

5.1.2 Physical System Description

The physical system of Breaking Effect includes visual terrian and a target 3-D object which will break into pieces.

5.1.3 Goal Statements

Given the target object, coefficient of friction, focus point and initial momentum level, the goal statements are:

GS1: Calculate initial status including pieces generation and initial speed of each piece.

GS2: Calculate trace of motion for each piece.

5.2 Solution Characteristics Specification

The instance models that govern Breaking Effect are presented in Subsection 5.2.5. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

5.2.1 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [T], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: The forms of energy that are relevant for this problem are kinetic energy and potential energy. Thermal energy is assumed to be negligible in the air. All other forms of energy are negligible.

A2: Air friction will be ignored.

A3: Powders will not be considered.

A4: Initial kinetic energy and potential energy of target object are zero.

A5: The same coefficient of friction everywhere on flat ground.

A6: Pieces are not decomposable.

A7: Collision between pieces will not be considered.

5.2.2 Theoretical Models

This section focuses on the general equations and laws that Breaking Effect is based on.

Number	T1
Label	**Uniformly accelerated motion**
Equation	$S = v_0 t + \frac{1}{2}at^2$
Description	Description The above equation gives us the distance traveled without having to know the final velocity of the object. Where t is time duration and v_0 is initial speed. Acceleration a is defined as the rate of change of velocity with respect to time, in a given direction. This would mean that if an object has an acceleration of 1 ms-2 it will increase its velocity (in a given direction) 1 ms-1 every second that it accelerates.
Source	http://ibphysicsstuff.wikidot.com/uniformaccmotion
Ref. By	IM2,IM3

Number	T2
Label	**Conservation of mechanical energy**
Equation	$E_{mech} = E_k + E_p$
Description	The total mechanical energy (defined as the sum of its potential and kinetic energies) of a particle being acted on by only conservative forces is constant. The potential energy, E_p, depends on the position of an object subjected to a conservative force. The kinetic energy, E_k, depends on the speed of an object and is the ability of a moving object to do work on other objects when it collides with them.
Source	http://www.nuclear-power.net/laws-of-conservation/law-of-conservation-of-energy/conservation-of-mechanical-energy/
Ref. By	IM2

Number	T3
Label	**Work done by kinetic friction**
Equation	$\Delta E_k = W_f$
Description	Kinetic energy loses due to kinetic energy. W_f is work done by kinetic friction. As a result, kinetic energy transform to internal energy.
Source	http://teacher.pas.rochester.edu/phy121/lecturenotes/Chapter07/Chapter7.html
Ref. By	IM3

5.2.3 General Definitions

This section collects the laws and equations that will be used in deriving the data definitions, which in turn are used to build the instance models.

5.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given.

Number	DD1
Label	**Kinetic friction**
Symbol	F_k
SI Units	N
Equation	$F_k = \mu_k F_n = \mu_k mg$
Description	Kinetic friction F_k is a force that acts between moving surfaces. An object that is being moved over a surface will experience a force in the opposite direction as its movement. The magnitude of the force depends on the coefficient of kinetic friction between the two kinds of material. Every combination is different. The coefficient of kinetic friction is assigned the Greek letter "mu" (), with a subscript "k". The force of kinetic friction is k times the normal force on an object, and is expressed in units of Newtons (N).In this project, F_n equals to gravity.
Sources	http://www.softschools.com/formulas/physics/kinetic_friction_formula/92/
Ref. By	IM3

5.2.5 Instance Models

This section transforms the problem defined in Section 5.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 5.2.4 to replace the abstract symbols in the models identified in Sections 5.2.2 and 5.2.3.

The goals GS1 and GS2 are solved by IM1 to IM3. Piece generation is done by calling cutting function in game engine.

Number	IM1
Label	**Find the angle between v_0 and horizontal. Find the angle between x axiom and projection on horizontal of initial speed**
Input	x_n,y_n,z_n
Output	$\theta_1 = arctan\frac{\lvert z_n \rvert}{\sqrt{x_n^2+y_n^2}}$ $\theta_2 = arctan\frac{y_n}{x_n}$
Description	x_n is x coordinates of gravity center of piece n. y_n is y coordinates of gravity center of piece n. z_n is z coordinates of gravity center of piece n.
Sources	
Ref. By	IM2

Number	IM2
Label	**Uniformly accelerated motion to find displacement in the air**
Input	v_0,θ_1 from IM1,θ_2 from IM1,t,g
Output	$S_x = v_0 \cdot cos\theta_1 \cdot cos\theta_2 \cdot t$ $S_y = v_0 \cdot cos\theta_1 \cdot sin\theta_2 \cdot t,$ $S_z = v_0 \cdot sin\theta_1 \cdot t - \frac{1}{2}gt^2$
Description	v_0 is the initial speed. t is time from beginning. θ_1 is the angle between v_0 and horizontal. θ_2 is the angle between x axiom and projection on horizontal of initial speed. The above equation applies as long as the piece moving in the air.
Sources	
Ref. By	

Number	IM3
Label	**Uniformly accelerated motion to find displacement on the ground**
Input	v_0,θ_1,θ_2,t,g,μ_k
Output	$a = \mu_k g$ $S_x = v_0 \cdot cos\theta_1 \cdot cos\theta_2 \cdot t - \frac{1}{2}at^2$ $S_y = v_0 \cdot cos\theta_1 \cdot sin\theta_2 \cdot t - \frac{1}{2}at^2$,
Description	v_0 is the initial speed. t is time from beginning. θ_1 is the angle between v_0 and horizontal. θ_2 is the angle between x axiom and projection on horizontal of initial speed. μ_k is the Coefficientoffriction. The above equation applies as long as the piece moving on the ground.
Sources	
Ref. By	

Derivation of how to get angle between v_0 and horizontal

We have initial location (X,Y,Z) of target object, and location (x_n, y_n, z_n) for piece n, then we can calculate the angle between v_0 and horizontal by:

$tan\theta_1 = \frac{|z_n - Z|}{\sqrt{(x_n - X)^2 + (y_n - Y)^2}}$

If location of target object is (0,0,0), then we have:

$tan\theta_1 = \frac{|z_n|}{\sqrt{x_n^2 + y_n^2}}$

$\theta_1 = arctan\frac{|z_n|}{\sqrt{x_n^2 + y_n^2}}$

5.2.6 Data Constraints

Tables 1 and 3 show the data constraints on the input and output variables, respectively. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters in Table 1 are listed in Table 2.

Table 1: Input Variables

Var	**Physical Constraints**	**Software Constraints**	**Typical Value**	**Uncertainty**
X		$L_{\text{min}} \leq X \leq L_{\text{max}}$	0	10%
Y		$L_{\text{min}} \leq Y \leq L_{\text{max}}$	0	10%
Z		$L_{\text{min}} \leq Z \leq L_{\text{max}}$	0	10%
x_n		$L_{\text{min}} \leq x_n \leq L_{\text{max}}$	0	10%
y_n		$L_{\text{min}} \leq y_n \leq L_{\text{max}}$	0	10%
z_n		$L_{\text{min}} \leq z_n \leq L_{\text{max}}$	0	10%
v_0	$v_0 > 0$	$v_{\text{min}} \leq v_0 \leq v_{\text{max}}$	20 m/s	10%
μ_k	$\mu_k > 0$	$\mu_{\text{min}} \leq \mu_k \leq \mu_{\text{max}}$	0.05	10%

Table 2: Specification Parameter Values

Var	**Value**
v_{min}	10 m/s
v_{max}	100 m/s

Table 3: Output Variables

Var	**Physical Constraints**
S_x	$S_x \geq 0$
S_y	$S_y \geq 0$
S_z	$S_z \geq 0$

5.2.7 Properties of a Correct Solution

A correct solution must exhibit the principle of motion as well as conservation of energy.

6 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

6.1 Functional Requirements

R1:

symbol	unit	description
X		x coordinates of target object
Y		y coordinates of target object
Z		z coordinates of target object
E		Initial momentum level
μ_k		Coefficient of friction everywhere on flat ground

R2: Use inputs in R1 to find initial speed v_0, as follows:

$v_0 = E \cdot 10$

R3: Verify that the inputs satisfy the required physical constraints shown in Table 1.

R4: Generate pieces by calling cutting function in game engine. Then get gravity center of each piece.

R5: Calculate the angle between v_0 and horizontal(θ_1). Calculate the angle between x axiom and projection on horizontal of initial speed(θ_2).

R6: Calculate and output trace of motion for each piece in the air(S_x, S_y, S_z).

R7: Calculate and output trace of motion for each piece on the ground(S_x, S_y, S_z).

6.2 Nonfunctional Requirements

Performance is influenced by amount of pieces, capability of GPU and CPU. Other nonfunctional requirements are correctness and reusability.

7 Likely Changes

LC1: A5 - In real situation, users may have kinds of terrians and textures. As a result, coefficient of friction needs to be changed correspondingly.

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 4 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 5 shows the dependencies of instance models, requirements, and data constraints on each other. Table 6 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

	T1	T2	T3	DD1	IM1	IM2	IM3
T1						X	
T2						X	
T3				X			X
DD1							
IM1							
IM2							
IM3			X	X			

Table 4: Traceability Matrix Showing the Connections Between Items of Different Sections

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

	IM1	IM2	IM3	R1	R2	R3	R5	R6	R7
IM1				X			X		
IM2	X						X	X	
IM3				X					X
R1					X			X	X
R2									
R3									
R5								X	
R6									
R7									

Table 5: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A1	A2	A3	A4	A5	A6
T1		X				
T2	X					
T3					X	
DD1					X	
IM1						
IM2		X				
IM3					X	
LC1		X				X

Table 6: Traceability Matrix Showing the Connections Between Assumptions and Other Items

References

W. Spencer Smith. Systematic development of requirements documentation for general purpose scientific computing software. In *Proceedings of the 14th IEEE International Requirements Engineering Conference, RE 2006*, pages 209–218, Minneapolis / St. Paul, Minnesota, 2006. URL http://www.ifi.unizh.ch/req/events/RE06/.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ȧgerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.

W. Spencer Smith, John McCutchan, and Jacques Carette. Commonality analysis for a family of material models. Technical Report CAS-17-01-SS, McMaster University, Department of Computing and Software, 2017.

9 Appendix

9.1 Symbolic Parameters