**Real-time Physics Engine**

Jeremy Nguyen

**1. Overview**

A physics engine simulates how matter would interact under certain physical constraints and output the state of the system so that it could be utilized later. Many modern applications require the computation ability of a physics engine. A few examples of applications that depends on physic engines are game engines, ray tracing, and mathematics modeling.

This project attempts to create a simple 2-dimensional rigid-body physics engine which is capable of computing collisions between objects under natural or user-defined physical constraints including collision type (ranging from perfectly inelastic to perfectly elastic), gravitational force acting on the object, object mass, etc.

Furthermore, this program can also render the system of interest in real time, with the defined constraints for visualization purposes.

This paper will discuss about the design of the program by analyzing the mathematics and physics concepts behind rigid bodies, as well as the object-oriented design of the program in C++.

**2. Mathematical Foundation**

Because rigid body interactions involve object movements and force analysis, the major mathematical tool used to make interaction computation possible is linear algebra.

***2.1. Normal Vectors***

Normalizing vectors refers to scaling vectors down or up such that their normalized magnitude equals to 1. Here we can see the formula to find the normal vector of , :

Operating with unit vectors is easy because their magnitude is 1, thus, they can represent other vectors on the same line as a linear combination. This program normalizes vectors on a few occasions such as to separate objects from each other and to resolve collisions (more on these subjects later).

***2.2. Dot Product***

The dot product between two vectors is a scalar value obtained by summing the products of the nth component of the first vector and the nth component of the second vector:

This scalar value is also obtainable by the following equation:

,

θ is the angle between the two vectors