

Milwaukee School of Engineering
CS421 – Advanced Computer Graphics
Lab 4 – Animation

Student: Marek Handl

Date: January 2007

➤ Implementation and functionality

For each Lego brick almost everything is randomized – starting position, starting time, direction of movement, direction of rotation and color. Only parameter that is not randomized is y-axis movement. This motion is based on something like a gravity effect.

Core of equation I use to compute y-coordinate is “ $\text{abs}(\cos(\text{time})) * \exp(-\text{time})$ ”, which makes the bricks go down in accelerated motion and then bounce back up. The amplitude of y-coordinate is getting smaller with time.

Information about FPS (targeted FPS and actual FPS) is visible in lower part of the screen. To achieve targeted FPS the program changes timer delays based on last 3 FPS measured.

Each Lego brick is an independent object encapsulating all information about the brick.

➤ Controls

All characters have to be lower case.

.	and /	... change FPS
[and]	... speed up/down animation
-	and +	... zoom in/out
n		... regenerate Lego bricks
r		... reset Lego bricks values (no new generation)

➤ Extra features

- accelerated movement
- statistics about FPS
- possibility to change FPS in run-time
- enhanced FPS accuracy

➤ Problems

I spent a large amount of time trying to create the right equation for the gravity effect but I wouldn't call it a problem.

➤ Files

main.cpp – main program and functions using OpenGL. Other functionality is in header files, because used methods are not complicated and creating adequate cpp files would be only confusing.

vector.h – my class representing a vector of 3 values

myobject.h – represents one Lego brick, encapsulates all information about the brick and methods connected to the brick (e.g. movement of the brick)

fps_object.h – object used for FPS computation and analysis

Lego.cpp is used to create Lego bricks.

stdafx.cpp is used to include some standard libraries.