Assignment – Drop The Puck

OVERVIEW

Date Due: Thursday, April 16th by 11:59 p.m. EST. ✧ **Value:** 40 points

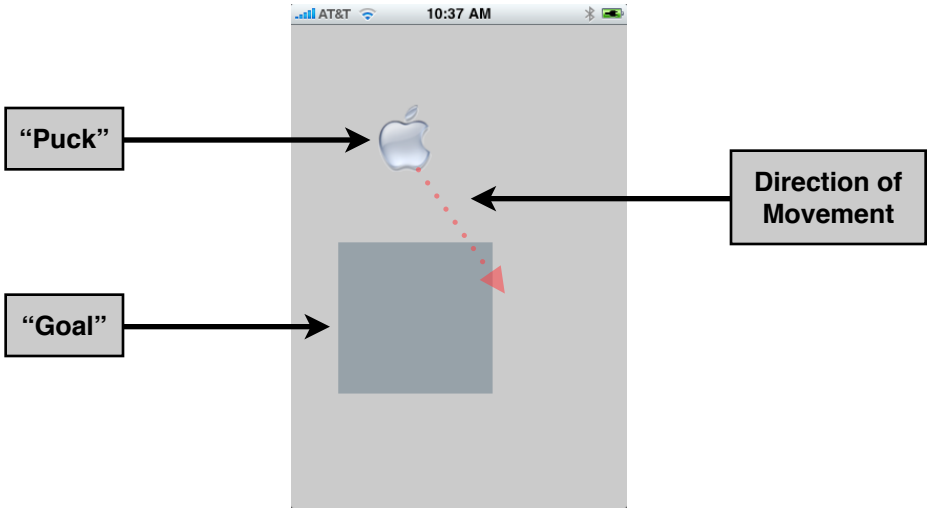
Overview

In this assignment, you are going to write a pretty boring hockey game. This is a single player hockey game, with no defense. Players position the goal using a multi–touch gesture, and drop the puck with a single fingered tap gesture. The puck slides around the screen, using the accelerometer to guide its path, until a goal is scored. A goal is scored when the puck is completely contained in the goals bounds, at which point the puck is removed from the screen and the user gets to hear a congratulatory clapping sound!

You will practice processing multiple touches, learn to work with UIAccelerometer (making use of the low–pass filter discussed in class), and learn to play an audio file.

Getting Started

You will be provides some code, which can be found at: http://www.cs.umd.edu/class/spring2010/cmsc498i/files/lab9/Lab9_DropThePuck_resources.zip. GeometryUtilities.h defines a function you can use to make sure the puck doesn't leave the visible screen. When attempting to move the puck, use CGRectByConstrainingMovingRectToContainerRect to make sure you don't move the puck offscreen. The iPhone Simulator does not have an accelerometer and will not generate accelerometer events. However, you can use the -simulateRandomGravity category method in UIAccelerometer_Additions.h to simulate some if you can't immediately get your hands on a device.



NOTES

Touch Tracking

Track touches from the point they contact with the screen. Continue tracking all movements until they break contact with the screen. The goal of your touch processing is to differentiate between a mult-touch and single tap gesture. If the user touches the screen with a single finger and breaks contact without ever moving, you can consider this a tap. UITouch makes determining this easy. The tapCount property will be set to 1 in exactly this scenario.

When a single tap gesture if processed, place the puck at the tap location. When a multi–touch gesture is processed, place the goal on screen and resize it so that it completely enclosing the all touches.

If you repsenent the puck and goal using UIViews, you may need to make sure they don't intercept touch events you want to process in the container. By disabling user interaction (view.userInteractionEnabled = NO), you effectively make a view invisible to the touch input system.

Game Timer

Using input from the accelerometer, you will update the position of the puck ever 1/30th of a second. To do this, you need to set up a repeating NSTimer to fire at this rate. (Another option is to use CADisplayLink if you want to investigate that class).

Create a and start a timer with this single line of code:

```
myTimer = [[NSTimer scheduledTimerWithTimeInterval:UPDATE_RATE target:self
                                          selector:@selector(animationTick)
                                          userInfo:nil repeats:YES] retain];
```

When you want to stop a timer, use this code:

```
[myTimer invalidate];
[myTimer release];
myTimer = nil;
```

The scheduled timer will invoke the target's -(void)animationTick; method every UPDATE_RATE seconds. Your animationTick code should update the puck's position, and determine if a goal has been scored. Finally, make sure to stop the timer whenever possible. For a better user experience, you may consider stopping the timer while touches are being tracked. Regardless, the timer and accelerometer should be disabled when not needed -- when the puck is not visible.