

Designing Rhythm Games for Touchscreen Devices



Project Progress Report

Members: Philip H. Peng

Advisor: Dr. Stephen H. Lane

CIS 401, Fall 2011, University of Pennsylvania

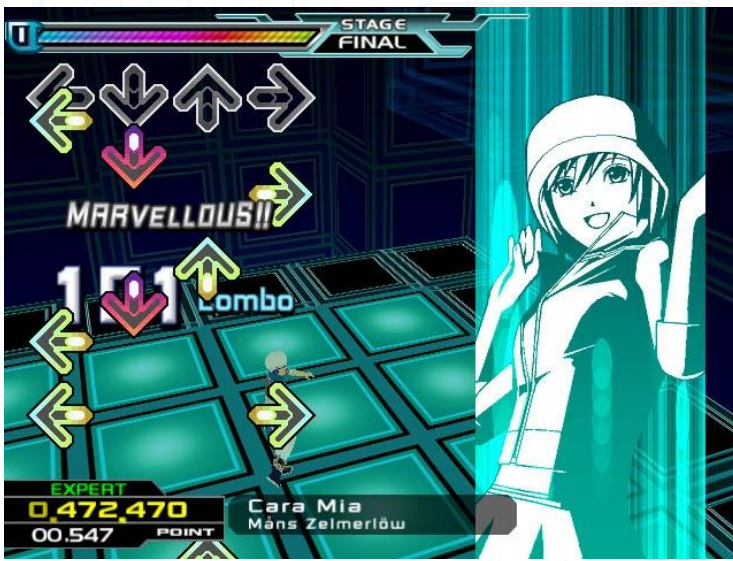
Presentation Overview

- 1) Summary
- 2) Project Proposal
- 3) Related Work
- 4) Project Outline
- 5) Progress
- 6) Demo
- 7) Results



Summary

- Rhythm game: time critical, response-based
- Touchscreen: new input method
- Rhythm game for touchscreen: how to design the interface for highly reactive gameplay?



Project Proposal

Goal:

Design, prototype, and evaluate different rhythm games interfaces for touchscreen devices.

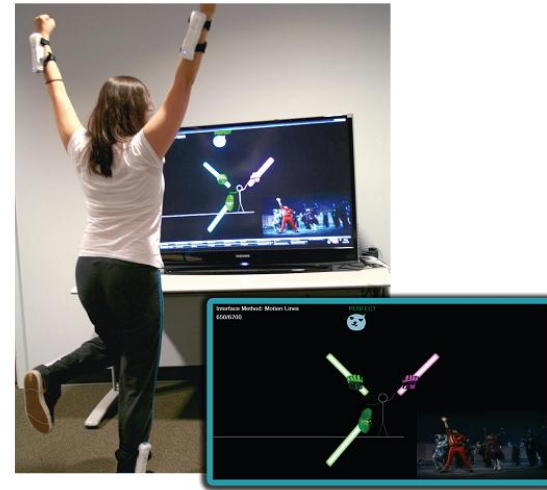
Approach:

Create a rhythm game prototype for Android tablets that demos various game interfaces and collects usage data to evaluate their effectiveness.

Related Work

Wiimote + Dance Game

“Understanding Visual Interfaces for the Next Generation of Dance-Based Rhythm Video Games” – University of Central Florida, Orlando, FL



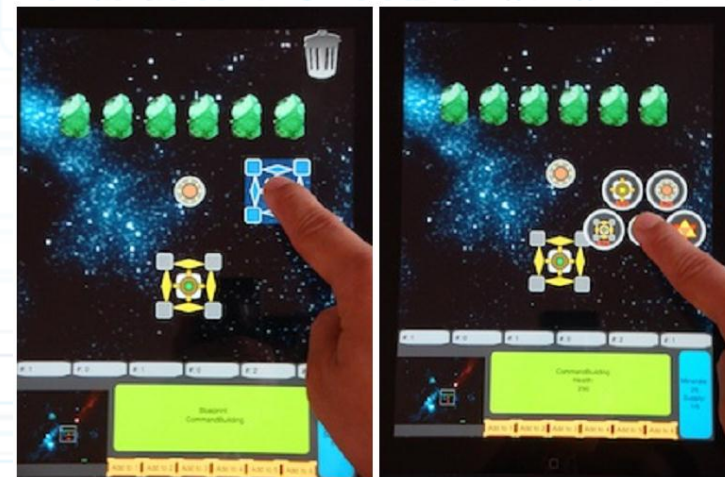
External Multi-touch Panel + Turn-Based Strategy Game

“A Study on Multi-Touch Interface for Game” – Chung-Ang University, Seoul, Korea



Overlaid Multi-touch Screen + Real-Time Strategy Game

“One-handed Interface for Multi-Touch Enabled Real-Time Strategy Games” – University of California, Santa Cruz, CA



Project Outline

1) Design – Draft

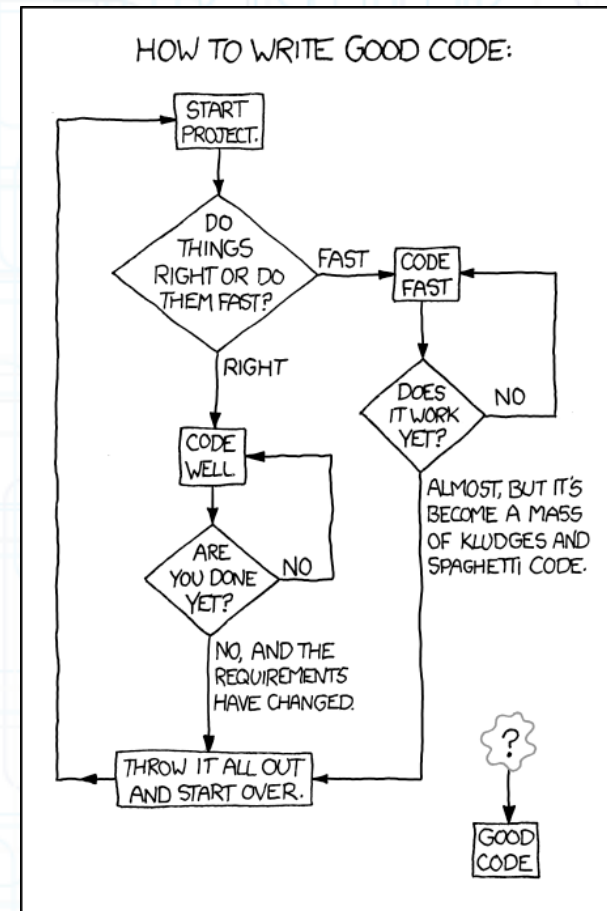
- Investigate existing interface designs
- Draft designs and evaluation metrics

2) Prototype – Code

- Implement these designs via Android
- Use common backbone to reduce non-relevant factors

3) Evaluation - Data

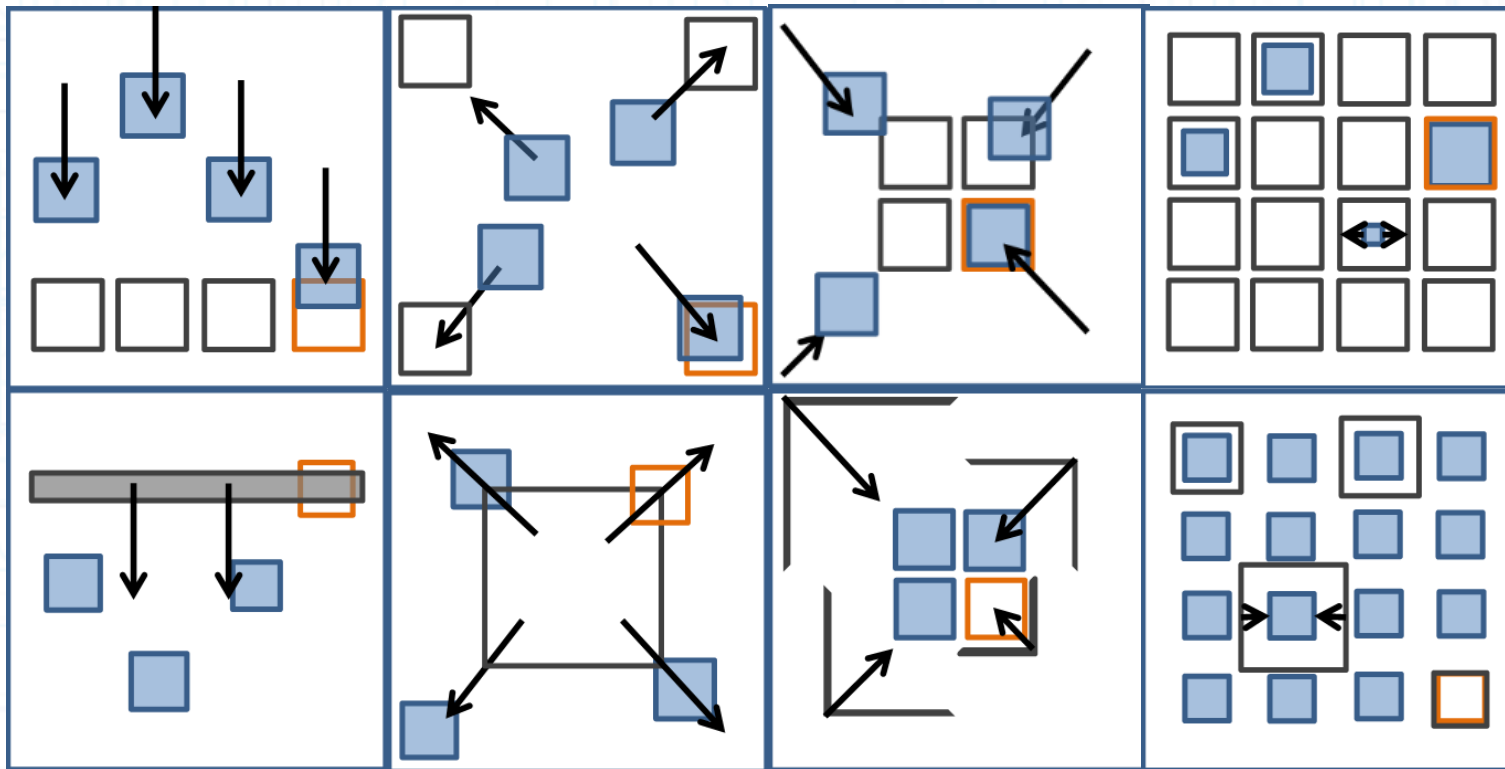
- Release to public with data analytics plugin
- Analyze collected data



Progress

Design – Designs

- Interface designs finalized:



Progress

Design – Metrics

- Test song: smoooch (Beatmania IIDX)
 - High note frequency and 177 BPM
 - Strong audible baseline (good for rhythm)
 - Auto-generated stepfile (Dancing Monkeys)
- Metrics per design:
 - Total accuracy percent (data)
 - Missed note count (data)
 - Ranked enjoyability relative to other designs (feedback)
- Full timing chart will also be collected for overall trend analysis (if there are any)

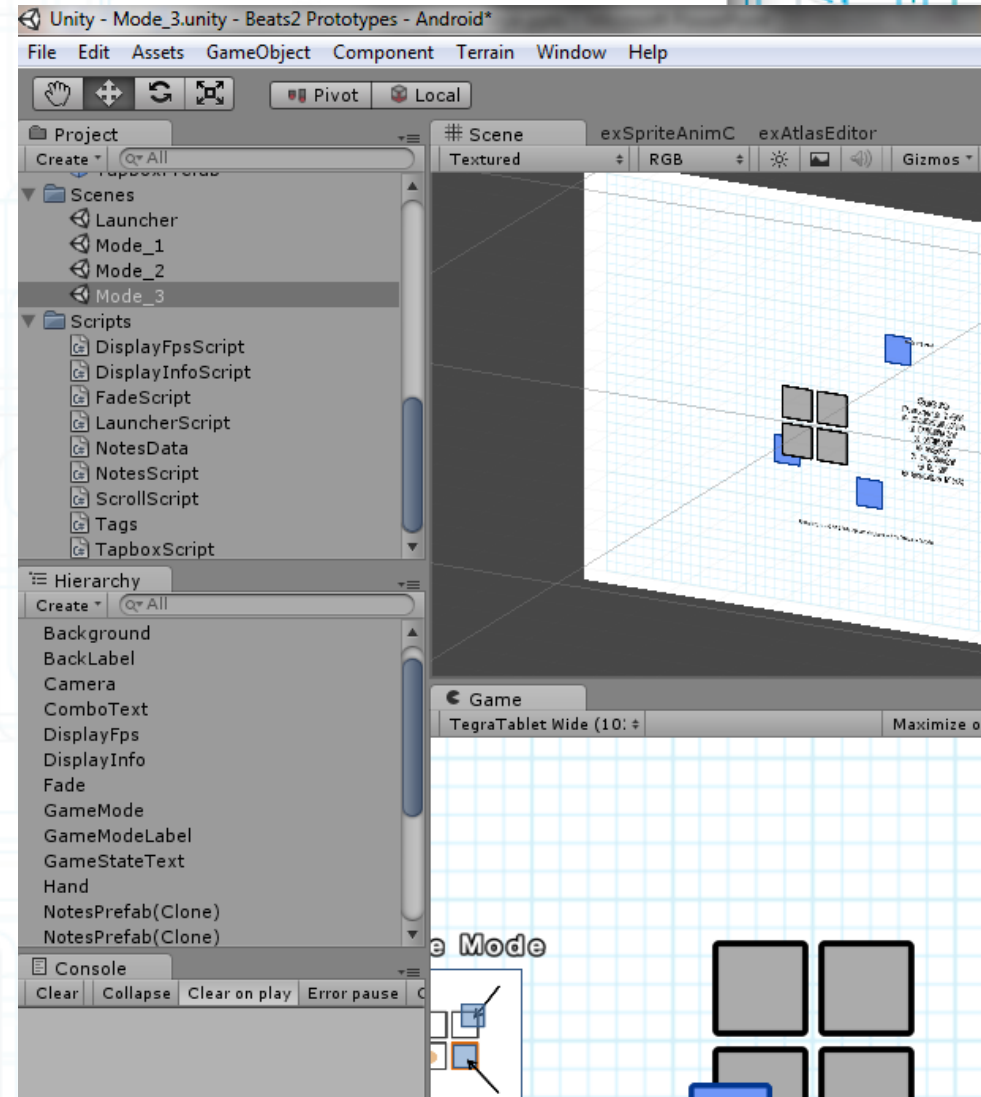


Progress



Prototype – Unity

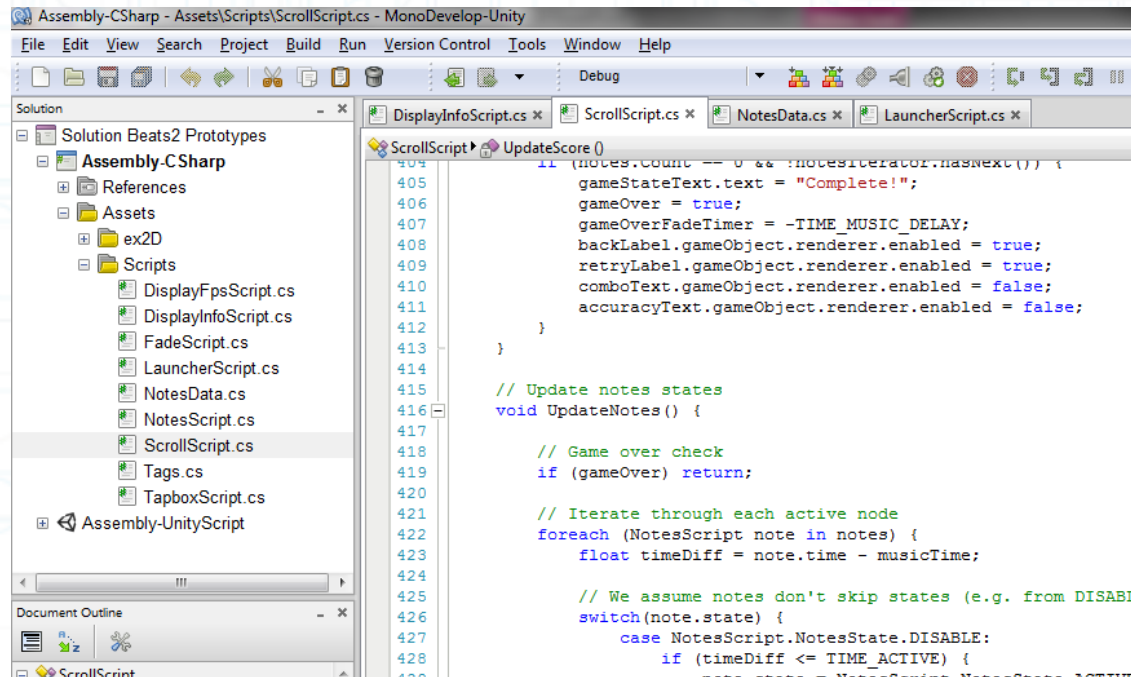
- Biggest challenge of project
 - Game designer vs. Programmer
 - Unity = GameObject driven workflow
 - New to C#
- Game Engine Experimentation
 - Android + iOS license
 - Othello2D vs ex2D
 - TouchGestures overcomplicated
 - MonoDevelop minimalistic



Progress

Prototype – Coding

- Flexible game engine
 - Timer, audio-synchronized
 - Dynamic object generator, memory efficient
 - Score tracking, based on timing accuracy
 - Runs on Android tablet and Windows
 - Adding new modes (interfaces) easy



The screenshot shows the MonoDevelop-Unity IDE with the following details:

- Window Title:** Assembly-CSharp - Assets/Scripts/ScrollScript.cs - MonoDevelop-Unity
- Menu Bar:** File, Edit, View, Search, Project, Build, Run, Version Control, Tools, Window, Help
- Toolbar:** Includes icons for file operations, debugging, and development.
- Solution Explorer:** Shows a project named "Solution Beats2 Prototypes" containing:
 - Assembly-CSharp
 - References
 - Assets
 - ex2D
 - Scripts
 - DisplayFpsScript.cs
 - DisplayInfoScript.cs
 - FadeScript.cs
 - LauncherScript.cs
 - NotesData.cs
 - NotesScript.cs
 - ScrollScript.cs (selected)
 - Tags.cs
 - TapboxScript.cs
 - Assembly-UnityScript

- Code Editor:** Shows the following C# code in ScrollScript.cs:

```
404     if (notes.Count == 0 && !notesIterator.HasNext()) {  
405         gameStateText.text = "Complete!";  
406         gameOver = true;  
407         gameOverFadeTimer = -TIME_MUSIC_DELAY;  
408         backLabel.gameObject.renderer.enabled = true;  
409         retryLabel.gameObject.renderer.enabled = true;  
410         comboText.gameObject.renderer.enabled = false;  
411         accuracyText.gameObject.renderer.enabled = false;  
412     }  
413 }  
414  
415 // Update notes states  
416 void UpdateNotes () {  
417  
418     // Game over check  
419     if (gameOver) return;  
420  
421     // Iterate through each active node  
422     foreach (NotesScript note in notes) {  
423         float timeDiff = note.time - musicTime;  
424  
425         // We assume notes don't skip states (e.g. from DISABLE  
426         switch (note.state) {  
427             case NotesScript.NotesState.DISABLE:  
428                 if (timeDiff <= TIME_ACTIVE) {  
429                     note.state = NotesScript.NotesState.ACTIVE;  
430                 }  
431             default:  
432                 // Do nothing  
433             }  
434     }  
435 }
```

Progress

Evaluation – Setup

- Tweaking stage
 - Change object placements based on feedback
 - Tweak timing parameters
 - Improve graphics?
- Google Analytics vs own server
 - Analytics: generate graphs but limited info collection
 - Server: custom information but have to set up
- Mass release on Android Market
 - Use Beats' update notifier to advertise (100k+ active users)

Demo



Game Mode

Score
Percents: 1
1 MARVE
2 PERF
4 GRE/
0 GOC
0 ALM
0 MIS
7 Combo

7 Combo
GREAT (85ms)

Game Mode

Score
Percents: 48.6%
11 MARVELOUS
17 PERFECT
10 GREAT
6 GOOD
5 ALMOST
17 MISS
7 Combo MAX

Game Mode

Score
Percents: 0.0%
4 MARVELOUS
8 PERFECT
8 GREAT
6 GOOD
4 ALMOST
91 MISS
6 Combo MAX

Game Mode

Score
Percents: 1
5 MARVE
10 PER
9 GRE
5 GO
1 ALM
8 MI
12 Comb

1 Combo
MARVELOUS (-12ms)

Game Mode

Score
Percents: 3
3 MARVE
8 PERF
16 GR
15 GO
11 ALM
25 MI
7 Combo

GOOD (-126ms)

Game Mode

Score
Percents: 0.0%
4 MARVELOUS
8 PERFECT
8 GREAT
6 GOOD
4 ALMOST
91 MISS
6 Combo MAX

MISS (-206ms)

Game Mode

Score
Percents: 1
1 MARVE
2 PERF
4 GRE/
0 GOC
0 ALM
0 MIS
7 Combo

MISS (-211ms)

Results

Programming:

- Unity hard to learn but very flexible and worthwhile - write once, deploy everywhere
- ex2D very useful but very buggy and crashes
- Framerate very good (consistent 60fps) but timing window a bit too big (~15ns update rate)
- Code currently in one main script – need to learn more about Unity coding conventions (static global object?)

Results

Informal surveys:

- Testers used to DDR style scrolling
- Speed needs to be tweaked (some tester just had slow visual processing times)
- Tapbox placement definitely a factor (proximity to each other, spread of focus)
- Enjoyment factor independent from performance
- May try two songs to eliminate song familiarity from evaluation

Questions?

↕ [MercurialMadnessMan](#) 767 points 13 days ago [-]



Sorry, this isn't really a question