Shooter? Painter!

1 EXECUTIVE SUMMARY

Shooter? Painter! is a fast-paced, highly intuitive game that allows anyone to be an artist. You take on the role of a recent art school graduate who's been hired by an art museum to solve a major problem: burglars have stolen the museum's entire collection; the only thing remaining are pictures of the paintings and you're asked to recreate the original paintings to the best of your ability, allowing the museum to resume its operations.

Combining elements of shooting and puzzle games, our game asks you to use touch-based input to project various types (e.g., different colors and sizes) of moving paintballs onto a canvas, thereby re-creating paintings one at a time. Your score is determined by the accuracy of your work, i.e., how close it is to the original painting.

We believe that this game uniquely challenges players' memory, dexterity, and visual perception skills, appealing to lovers of puzzles, shooting games, and visual art alike. The simple and fast-paced game mechanics allow for quick, fun-fueled gameplay and a quick learning experience.

The game is developed purely for mobile, touch-enabled platforms. We expect the end-to-end implementation to take 5 weeks or, assuming 3 part-time designers/developers, 140 man-hours.

Note: throughout the document, "V2:" sections denote features to be tested / implemented in a more advanced version of the game, once the basic functionality has been successfully develop.

2 GAME DESIGN - CREATIVE

2.1 High Concept

Combining elements of shooting and puzzle games, our game asks the player to project various types (e.g., different colors and sizes) of paintballs onto a canvas, thereby re-creating a painting that's been displayed at the beginning of the play session. Paintballs are randomly generated at the canvas boundaries and moving around the canvas (e.g., floating, bouncing); they can be projected onto the canvas by tapping them with a finger. The player's score is determined by the accuracy of her work, i.e., how similar it is to the original painting.

2.2 Design Goals

2.2.1 Main Design Features

2.2.1.1 Player goals and objectives.

The player's high-level goal is to pass a series of levels to win the game. On a per-level basis, the player's goal is to create a painting that is as close as possible to the original (primary goal), using as few paintballs as possible (secondary goal); this maximizes her in-game score.

Winning condition: each individual level is won by hitting a certain threshold on the primary goal metric, % completion of the painting at hand (i.e., X% of the painting needs to have been recreated accurately). A level is completed either when the painting is 100% completed or when the player runs out of paintballs (whichever happens first). The overall game is won / played through once the player has won all the levels, i.e., recreated all the paintings.

Example of original painting to be recreated in the game:

2.2.1.2 Main rules and procedures

Level selection / museum view: before playing a level in the canvas view, the player needs to select a level to play using the level selection / museum view, i.e., by clicking on the desired level canvas. After playing a level in the canvas view, the museum view is reactivated. If the level was successfully completed, with a score higher than the previously achieve , the museum's painting and score entries for this level is updated to reflect the latest game. Otherwise, the museum's collection remains unchanged.

Canvas view: at the start of a level, a painting appears on a canvas and then vanishes, leaving behind only an outline / frame that delineates same-color areas in the original painting. The player is then asked to project various types (e.g., different colors and sizes) of paintballs onto the canvas, thereby re-creating the original painting. Paintballs are randomly generated at the canvas boundaries and moving around the canvas (e.g., floating, bouncing); they can be projected onto the canvas by tapping them with a finger. Only tapping a ball of the right color at the right position contributes to finishing the painting and thus to increasing the player's score. To help align a given paintball's trajectory with a targeted area on the canvas, the player has the option to manipulate the canvas, through (1) translation and (2) rotation. The player is able, at any time, to put the game on hold by pressing a pause button (and resuming the game by pressing the same button again).

One of two events can lead to the end of a level: (1) the targeted painting completion threshold has been achieved or (2) the player has run out of resources. In either case, the player's final score is calculated and control is handed back to the level selection view.
Core game loop: (1) a ball is generated on the canvas and follows a predefined trajectory

(e.g., floating along a straight line or bouncing off the southern canvas boundary); (2) the user follows / predicts the ball's trajectory and manipulates the canvas (through translation and/or rotation) as needed for the ball to pass over a targeted area; (3) as the ball passes over an area of the same color, the player may decide to tap the ball, projecting it onto the canvas in that location; (4) the player's score increases, and visual and audio feedback is generated to instantaneously reflect how well the ball was placed.

2.2.1.3 Player Resources

On a per-level basis, the player's resource in the game are paintballs of different colors. These paintballs need to be projected on the canvas in a manner that recreates as much as possible the original painting, using as few balls as possible.

2.2.1.4 Conflicts

Obstacles: the combination of (1) a given painting to recreate and (2) a given pattern for generating paintballs during a level can be perceived as a puzzle to be solved by the player

Constraints:

(a) the paintballs generated throughout a given level are limited in number; the current paintball inventory (by color) is displayed next to the canvas and gives the player a real-time perspective on the resources still available. This inventory is depleted by projecting paintballs onto the canvas or paintballs colliding with a canvas boundary, upon which they vanish (exception: bouncing paintballs bounce off the southern canvas boundary) which puts pressure on the player to use paintballs judiciously / in a timely manner. When the player runs out of paintballs, the game session ends

(b) the paintballs follow trajectories that are generated by the game and cannot be influenced by the player; these trajectories thus strongly limit the player's degrees of freedom in projecting paintballs onto the canvas.

(c) image of original painting is only shown temporarily, at the beginning of a session; the game thus also becomes a test of the player's memory

Boundaries: canvas boundaries define the total game field and absorb paintballs upon collision. Painting outlines define the fields to be colored by paintballs of a certain color.

2.2.2 Appeal

The game incorporates elements of both puzzle and casual shooting games and thus targets players who enjoy either of those genres. Given the game's highly intuitive gameplay and light-hearted touch and feel, it can be enjoyed by players of all age categories, as long as they're able to operate touch-based devices at a reasonable speed.

The game's appeal is created by a unique combination of 5 elements:
(1) solving puzzles: direct intellectual stimulus and satisfaction from successfully passing a level

(2) vivid feedback: gratification from instant visual, audio, and score feedback upon projecting paintballs onto the canvas. To make things less predictable, some of this feedback is randomly generated, e.g., the color splatters on canvas

(3) engaging in creative process: satisfaction from producing own paintings and being able to share them / go back to them later; user investment expected to increase retention

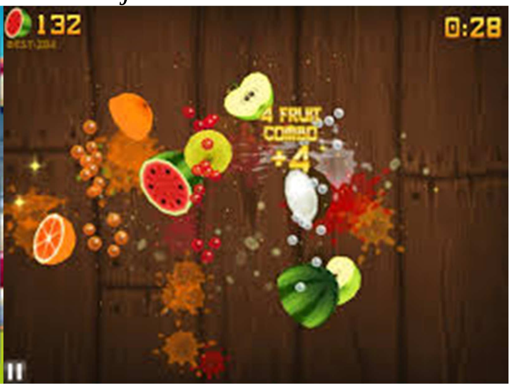
(4) physical learning effect: with game progression, satisfaction of gradually developing better vision / touch coordination and an ability to manage an increasing number of paintballs concurrently

2.2.3 Look and Feel

The game will have a light-hearted look and feel, with fast, action-packed game play and vivid / bright colors in an arcade style. In that sense, it will show some similarity with games such as *Candy Crush* and *FruitNinja*.

Candy Crush:

FruitNinja:



2.3 Worlds, Characters and Story

2.3.1 Back Story

The player takes on the role of an art school graduate who's been hired by an art museum to solve a major problem: burglars recently broke into the museum and stole its entire collection; the only thing remaining are pictures of the original paintings. The player is therefore asked to recreate the original paintings to the best of her ability, as per the pictures of the originals. Fulfilling this mission will allow the museum to resume its operations.

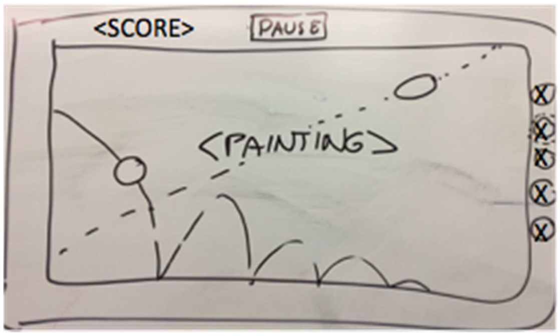
Rather than evolving as the player progresses in the game or being tightly integrated with the gameplay, this story provides a static context in which the game takes place. It explains why the player works her way through a series of (increasingly complex) paintings and is visualized through a dedicated "progression view" that shows all the paintings that the user has successfully recreated to-date.

2.3.2 Spaces/Worlds

All levels take place within the same view, namely that of the *canvas* (see below). This canvas has the following elements:

- Modifiable area (center): area onto which balls can be projected, i.e., that contains the painting being created
- Score caption (top boundary): displays the player's score in real-time
- Pause button (top boundary): allows the player to pause / unpause the game at any point in time
- Ball inventory captions (right boundary): one caption per color, indicating the number of balls remaining of that color

Canvas:

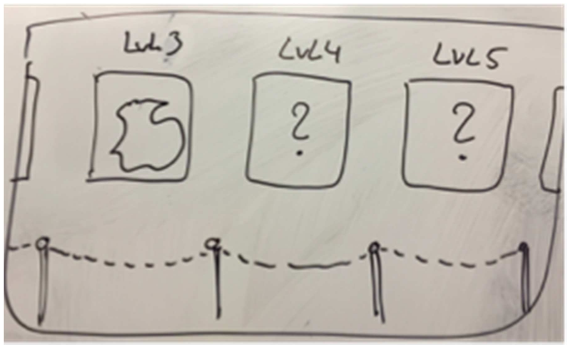


V2: to provide more diversity, one could consider embedding the canvas into a background picture and varying that background across levels and/or according to time of the day.

A second view, the *level selection / museum view* (see below), serves the follow purposes:

- It allows the player to horizontally scroll through and contemplate the paintings of the levels that she's beaten so far. Selecting a painting reveals the score achieved with that painting and a button for replaying the level.
- It allows the player to select, through click, the next level to play; this can be either a level that's already been played (i.e., in order to achieve a score higher than the currently stored one) or the lowest level that has not yet been beaten.

Level selection/museum:



2.3.3 Characters

Technically, the game involves a single character, namely the art graduate controlled by the player herself. However, since the core gameplay doesn't involve this character in any shape or form, the character is not actually depicted in the game.

V2: to encourage the player to further invest into the game, one could create a fully customizable avatar that is present on the screen throughout the game and visually evolves as the player progresses. This likely won't add anything to the gameplay.

2.3.4 Levels of Difficulty

The level of difficulty increases as the player progresses in the game, i.e., reaches higher levels. A higher degree of difficulty is realized by:

(1) greater complexity of paintings, incl. more colors, smaller same-color areas etc.
(2) more challenging ball characteristics, e.g., different size, faster / more convoluted movement
(3) more balls concurrently present on the screen

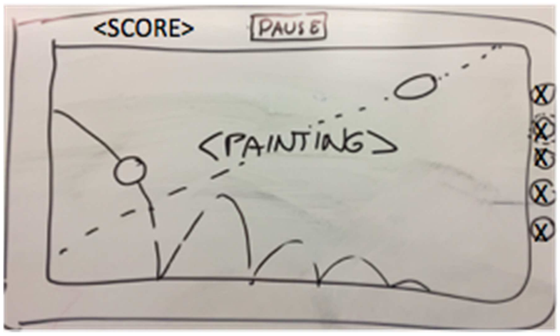
2.4 Interaction Models

2.4.1 User Interface - Navigation and Movement

Within a level, the user interface offers 4 ways of interacting with the game:

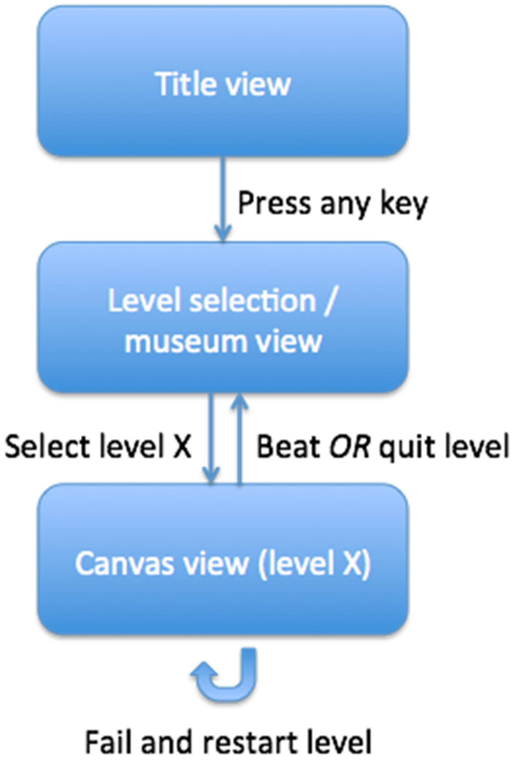
(1) Tapping the Pause button (1 finger): pauses/unpauses the game
(2) Tapping a ball (1 finger): projects the ball onto the canvas, at the ball's current location
(3) Tapping and translating the canvas itself (3 fingers): allows to move the canvas around, without rotation
(4) Tapping and rotating the canvas itself (3 fingers): allows to rotate the canvas, without translation

Canvas:



V2: (a) the ball inventory captions turn into selectable buttons; selecting a button of a certain color prioritizes the generating of balls of that color. This could enable more strategic play (details tbd); (b) instead of limiting the player's input to tapping paintballs, one could allow for swipes. The direction and intensity of a swipe would be reflected in the color splatters resulting from the projecting of paintballs onto the canvas; (c) instead of limiting the canvas manipulation to rotating and translating, we'd like to test whether resizing is a useful degree of freedom.

2.4.2 Game Play Sequence and Levels



2.4.3 User/Environment – Obstacles and Props

The objects in the canvas view are the following: paintball(s), same-color areas of the original painting, canvas, canvas boundaries

The player interacts with the paintballs by tapping on them. A successful tap on a ball (i.e., collision between player tap and paintball object) projects the same ball onto the canvas, resulting in color splatters in the ball’s most recent location. At the same time, the game checks for a collision between the ball and same-color areas of the original painting; if a collision is found and the color matches, scoring is adjusted accordingly. The ball is destroyed.

The player interacts with the canvas through three-finger touch moves that represent translation or rotation, as described previously.

Upon collision with a canvas boundary, balls are destroyed (exception: bouncing balls bounce off the southern boundary).

2.4.4 User/Character

No interaction with the main character.

V2: the player is able to customize her character, e.g., by dressing it differently.

2.4.5 Character/Character

<n/a>

2.4.6 Motion Tracking

n/a

V2: track the direction and intensity of single-finger swiping motions used to project balls onto the canvas.

2.4.7 Multi-Player

n/a

2.4.8 Mobile

The game will be built purely for mobile / touch-enabled devices, and leverages touch-based input as described above.

2.4.9 Networked Play

n/a

V2: instead of displaying the simple candy-crush like mechanism>

2.5 Performance and Scoring

2.5.1 State Variables

The following variables are necessary to save/restore or pause/resume the game:

- Overall game:
    - Player level: highest level that's been successfully beat
    - For each level previously beat: latest score and associated painting
- Within individual level:
    - Current player score
    - Paintball resources (per color)
    - State of all balls currently alive in the canvas
    - State of painting (i.e., balls previously projected on the canvas)

2.5.2 Feedback

The only negative feedback mechanism currently employed by the game is the difficulty that gradually increases with the level.

2.5.3 Performance and Progress Metrics

At the overall game level, progress is monitored through the levels that have already been beaten, and the score achieved on each of these levels. The game is won by beating all levels.

Within an individual level, progress is monitored through three metrics:

(1) the share of the original painting that has been recreated (in %). This metric gradually increases as balls are successfully placed on the canvas,

(2) the player's current score (in points). While the level is being played, this score increases every time (a) a paintball is successfully placed on an area of the same color as the ball (the score increase is determined by the accuracy of the placement and by the ball characteristics including speed and size) and/or (b) the share of the original painting that has been recreated increases.

A game session ends either when the painting is 100% recreated or when the player runs out of paintballs (whichever happens first). The level is beaten only if a certain threshold on the % completion of the painting has been hit.

3 GAME DESIGN - IMPLEMENTATION DETAILS

3.1 Design Assumptions

3.1.1 Hardware

800 MHz or higher CPU; 512 MB ram, Tegra 2, SGX 540, Mali 400, Adreno 205 or higher, 30mb free space and 2.2+ Android.

http://best-gamesinfo.blogspot.com/2014/02/candy-crush-saga-for-android-phones.html
http://candycrushsaga.com/help/en
https://support.halfbrick.com/hc/en-us/articles/202156094-What-are-the-system-requirements-for-Fruit-Ninja-Kakao-

3.1.2 Software

We will use Unity3D Game engine pro to develop the game. The game will be built for Android and ios platforms.

3.1.3 Algorithms and Techniques

Algorithms:

- In game points updating: During the game, the score shown on the screen updates when the player shoot a ball. To implement this, we firstly reduce the size of original picture to get ride of details.Then compare the color of the ball on the canvas with the color of the original image. The more similar the two color is, the higher score the player will get. More specifically, for balls with radius r, we reduce the size of the original image (with pixel h*w) to (h/r * w/r) to make sure each ball represents one single pixel on the shrunk image.
- Overall similarity comparison: After each level is finished, we would use a simple image similarity comparison algorithm to estimate the completion of the painting and calculate the final score. Below are the basic process of the algorithm:
    1) Reduce size : Shrink pictures to 8*8 with 64 total pixels.
    2) Reduce color : Convert 8*8 pictures to a gray-scale.
    3) Average the colors: Compute the mean value of the 64 colors.
    4) Compute the bits : Set each bit based on whether the color value is above or below the mean
    5) Construct the hash : Set the 64 bits into a 64-bit integer.
    6) Compare the hash value of the origin painting and the player's painting.
    7) (References: 1. http://www.hackerfactor.com/blog/?/archives/432-Looks-Like-It.html ; 2.http://stackoverflow.com/questions/25977/how-can-i-measure-the-similarity-between-two-images ; 3. https://code.google.com/p/mybackupcode/source/browse/trunk/python/imgHash/imgHash.py?r=31)

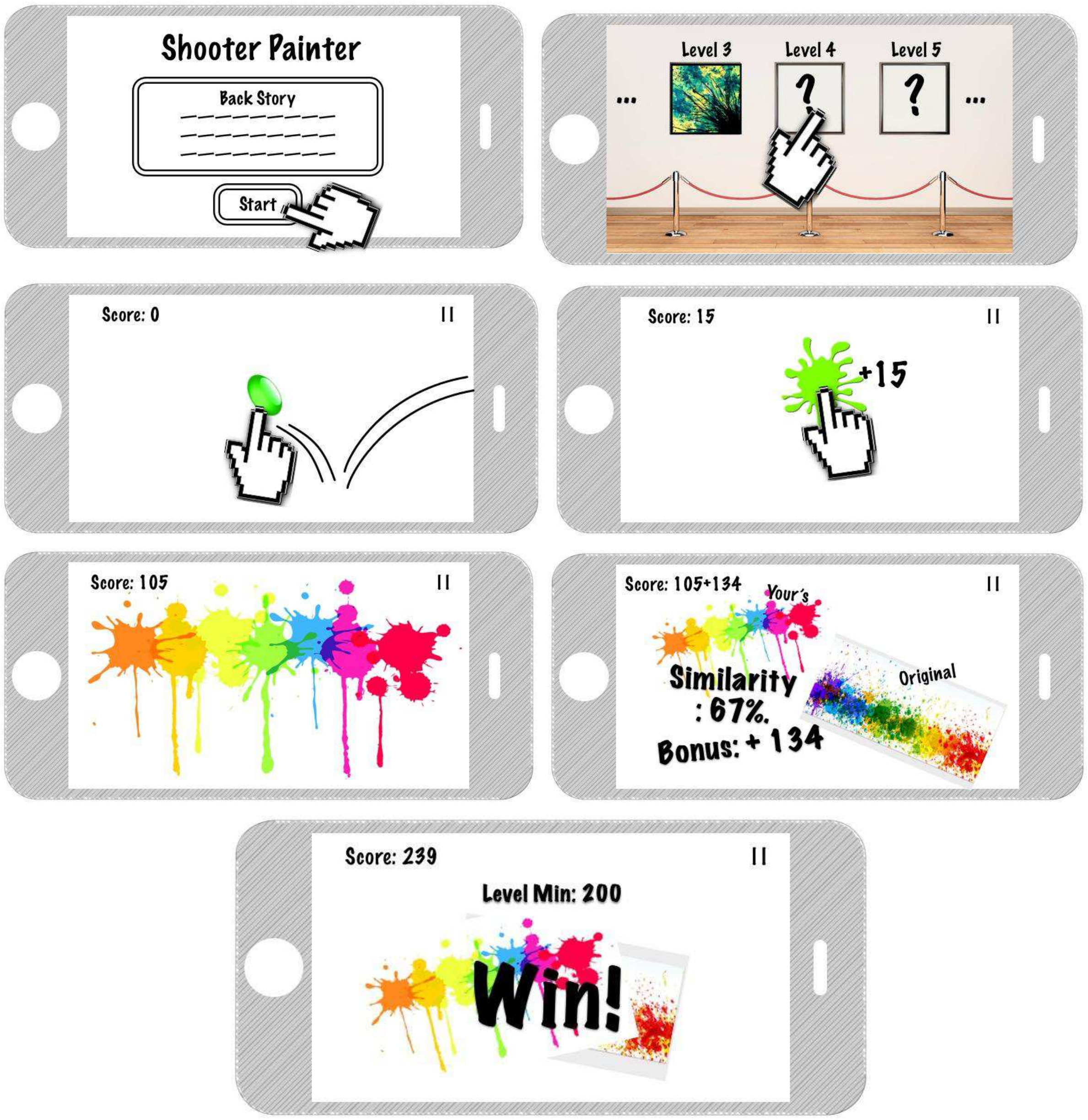
Plug-ins:

- None

Animation techniques:

- Sprites for animation of ball splatters
- Animation of elastic deformation for bouncing balls.

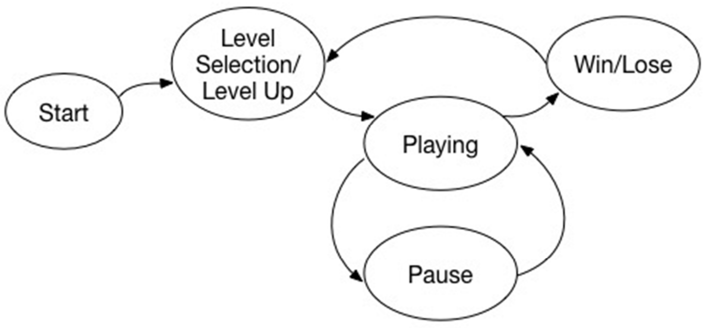
3.2 Storyboards



3.3 Design Logic

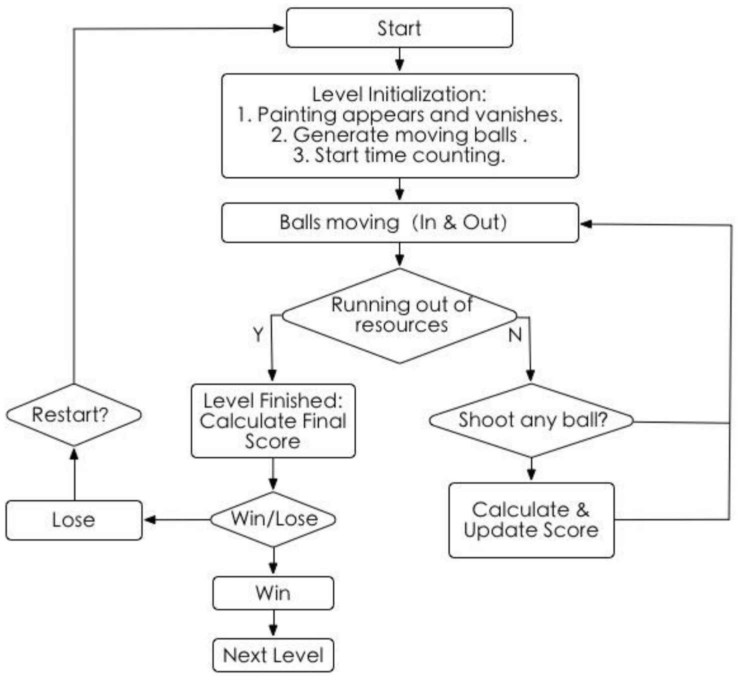
3.3.1 FSM - State/Effect

The states of the game are shown below in the Game State Graph. Player starts to enter the level selection scene.With certain level selected, the player then enters the playing mode. During the time he/she is playing , he/she can pause and resume. As long as the player finished the level, the win or lose scene can take the player go back to the level selection scene. The player can also choose to restart this level.



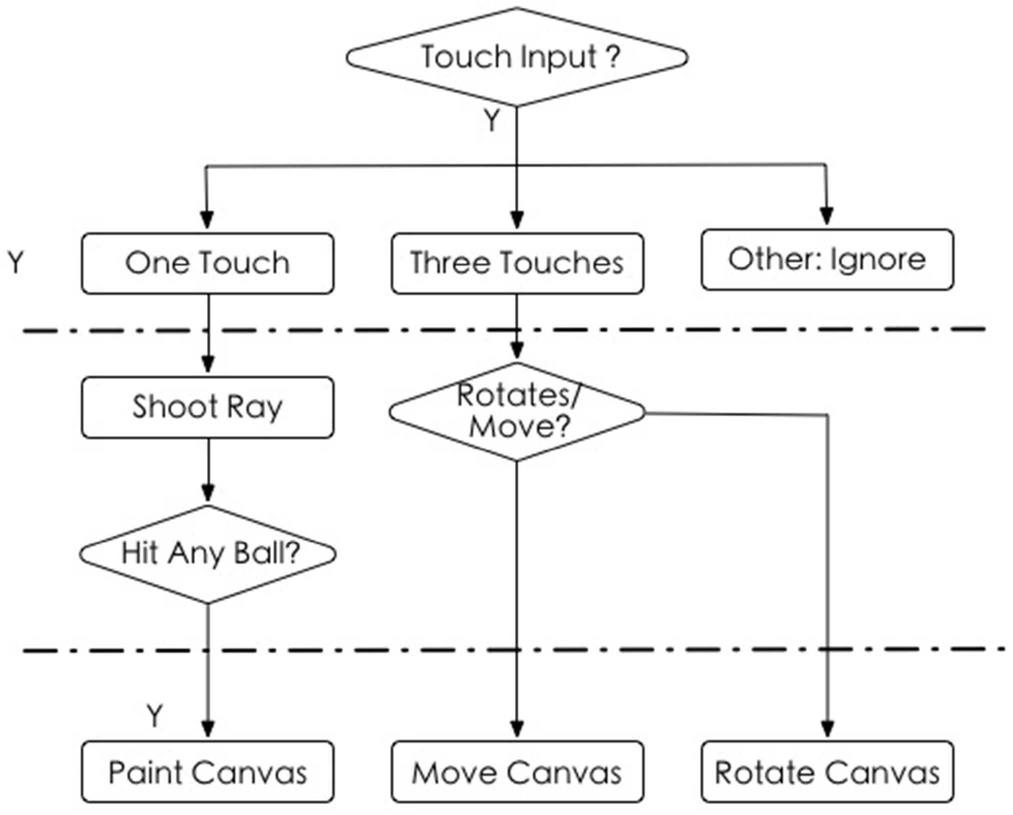
(Game State Graph)

During each Level, the game flow logic is shown in the Game Play Flowchart figure.



(Game Play Flowchart)

For the user input during the game, we define 3 kinds of inputs. Single touch means shooting the ball. After the detection with a single touch, a ray is shooted towards the canvas, if the ray hits any ball, we then draw a split on the certain position on canvas. When there is three finger touches the screen, we then detect if they are rotating or moving the canvas.



(Player Input&Output Graph)

3.3.2 User Solution/Actions

To accomplish a level, the player needs to gain certain score. The score the player get are composed of two parts. The first part is immediate score that updates during the gameplay. When player shoot the right ball, he/she will get an instance score increase. The other part of the score is the final bonus score that is calculated and added to the total score only after the player finishes the whole level.
The bonus score weighs more in the total score. Thus, to pass the level, the player should focus more on the overall look of the painting rather than shoots as many balls as possible.

3.4 Software Versions

3.4.1 Alpha Version Features (vertical slice through total experience)

For our alpha version, our main goal is to get the core mechanics for shooting and scoring because we will be able to test different constraints for our beta version.

Thus, the following set of features are to be included in the alpha version:

- Ball generation: We should be able to create randomized velocities and different colored balls from different angles
- Ball physics: The ball should be able to bounce on the floor and ceiling of the canvas and leave the canvas from the left and right
- Shooting: The basic touch implementation to painting the canvas should be in place. Additionally, the canvas should be painted on the canvas.
- Scoring: A basic scoring scheme should be in place that depends on the percentage of the canvas colored correctly.
- Scenes: There should be one level (faded painting in the background) and an end game scene.

3.4.2 Beta Version Features

For our beta version, our main goal is to have more advanced mechanics and more functionality in general. However, after playtesting, some of the mechanics may not make it to the beta version because the game does not feel fun.

Thus, the following set of features are to be included in the beta version:

- Scenes: There should be multiple levels, and end game scene, and a level selection scene.
- Pause screen: There will be a pause button and a pause screen.
- Swiping: A swiping implementation to break the balls will be implemented.
- Ball manipulation: Some sort of mechanic to manipulate the ball physics/path.
- Canvas manipulation: The canvas should be able to resize, translate a bit, and rotate using touch controls.
- Effects: Basic animation and audio will be implemented.

3.4.3 Description of Self-running Demos

The self-running demo should show off the ball physics from different angles and velocity, shooting the ball, the painting of the canvas, and the end game screen. All features should be demoable with their respective versions.

4 WORK PLAN

4.1 Tasks

		Description	Time	Member
1	**Task: Ball mechanics**	**All necessary ball mechanics in the game.**	**1 day**	**SC**

1.1	Subtask: Bouncing off floor but nothing else	As described.	0.5 days	SC
1.2	Subtask: Ball generation	The ball should have a random velocity, angle, and a set color based on resources.	0.5 days	SC
2	**Task: UI**	**Making the UI be user friendly and look nice.**	**4 days**	**SC**
2.1	Subtask: Make the in-game screen/UI.	Make the level user friendly (eg the pause button is in an aesthetically and functionally pleasing place)	1 day	SC
2.2	Subtask: Make pause screen.	Make a usable pause screen.	0.5 days	SC
2.3	Subtask: Making the level selection screen.	Make the level selection user friendly (eg with scrolling)	2 days	SC
2.4	Subtask: Making the end game screen.	Make both screens where the player wins and loses	0.5 days	SC
3	**Task: Level Design**	**Design all of the different levels**	**4 days**	**SC/ZZ**
3.1	Subtask: Design levels of varying difficulty	design as many levels as possible that vary in complexity.	2 days	SC/ZZ
3.2	Subtask: Tuning	Tune how many resources the player should get at each level.	2 day	SC/ZZ
4	**Task: Touch mechanics**	**All user interaction in game**	**13 days**	AH
4.1	Subtask: Basic shooting	Click on a ball, ball disappears and paints on canvas	5 days	AH
4.2	Subtask: Swiping	Swipe a ball, ball disappears and paints on canvas	3 days	AH
4.3	Subtask: Canvas manipulation	Pinch to resize, 2-finger translate to translate, 2-finger rotate to rotate	5 days	AH
4.4	Subtask: Ball manipulation	Different inputs to change the physics/path of the ball	2 days	AH
5	**Task: Effects**	**All visual and audio effects**	**9 days**	**ZZ**
5.1	Subtask: Animations	Get splatter animation, point animation as well as other necessary animation	5 days	ZZ

5.2	Subtask: Modeling	Make the balls, the canvas, look good	3 days	ZZ
5.3	Subtask: Audio	Make sound effects for different hits and background music	1 day	ZZ
6	**Task: Scoring**	**Score the player**	**3 days**	AH

4.2 Milestones

4.2.1 Minor

- Ball generation
- Ball physics
- Shooting
- Scoring
- Scenes

- Pause screen
- Swiping
- Ball manipulation
- Canvas manipulation
- Effects

4.2.2 Major

Alpha Version

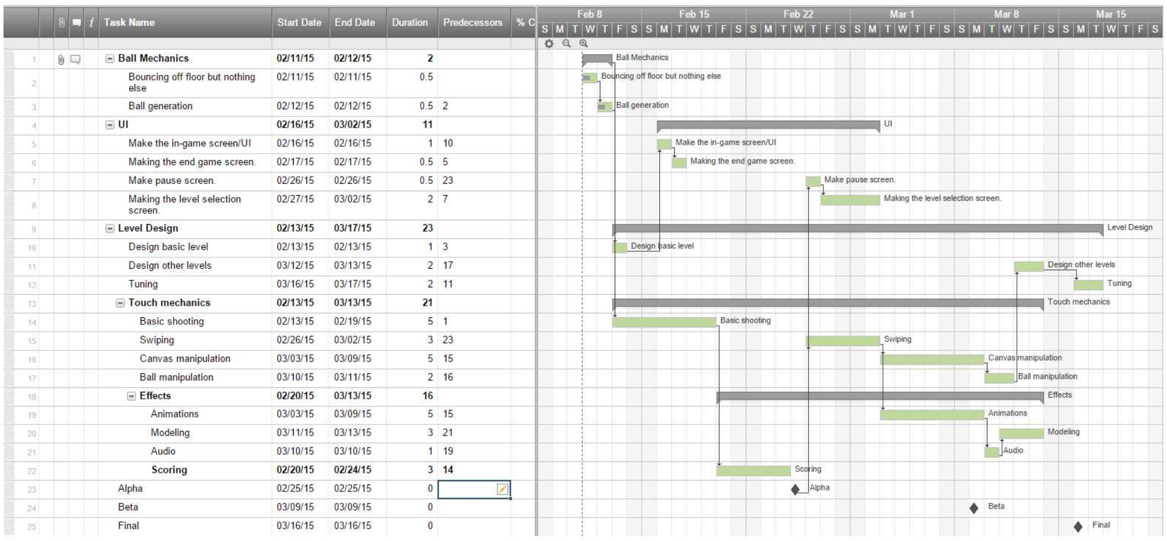
The alpha version should just have the core mechanics of the game - namely, balls should generate and bounce on the floor, touching a ball should paint the canvas and disappear, scores should be calculated, and an end screen display should show after a level ends.

Beta Versions

The beta version should have more complex mechanics in the game. Through playtesting we will possibly include the following: swiping, canvas manipulation, ball manipulation. There will be some basic animation and audio. There will be multiple levels of varying difficulty and a pause screen.

4.3 Development Schedule

Organize your work plan tasks in the form of a Gantt chart. List all the major tasks and subtasks (including durations) and the dates of each minor (weekly) milestone and major milestone (e.g. alpha, beta, final). Please use Microsoft Project for the creation of the Development Schedule.



	Task Name	Start Date	End Date	Duration	Predecessors	% C
1	**Ball Mechanics**	**02/11/15**	**02/12/15**	**2**		
2	Bouncing off floor but nothing else	02/11/15	02/11/15	0.5		
3	Ball generation	02/12/15	02/12/15	0.5	2	
4	**UI**	**02/16/15**	**03/02/15**	**11**		
5	Make the in-game screen/UI	02/16/15	02/16/15	1	10	
6	Making the end game screen.	02/17/15	02/17/15	0.5	5	
7	Make pause screen.	02/26/15	02/26/15	0.5	23	
8	Making the level selection screen.	02/27/15	03/02/15	2	7	
9	**Level Design**	**02/13/15**	**03/17/15**	**23**		
10	Design basic level	02/13/15	02/13/15	1	3	
11	Design other levels	03/12/15	03/13/15	2	17	
12	Tuning	03/16/15	03/17/15	2	11	
13	**Touch mechanics**	**02/13/15**	**03/13/15**	**21**		
14	Basic shooting	02/13/15	02/19/15	5	1	
15	Swiping	02/26/15	03/02/15	3	23	
16	Canvas manipulation	03/03/15	03/09/15	5	15	
17	Ball manipulation	03/10/15	03/11/15	2	16	
18	**Effects**	**02/20/15**	**03/13/15**	**16**		
19	Animations	03/03/15	03/09/15	5	15	
20	Modeling	03/11/15	03/13/15	3	21	
21	Audio	03/10/15	03/10/15	1	19	
22	**Scoring**	**02/20/15**	**02/24/15**	3	**14**	
23	Alpha	02/25/15	02/25/15	0		
24	Beta	03/09/15	03/09/15	0		
25	Final	03/16/15	03/16/15	0		