
Purposeful Gaming and the Biodiversity Heritage Library: Playfully Crowdsourcing Transcription

An Interim Project Report for Autumn 2014
From the Tiltfactor Research Laboratory at
Dartmouth College



Submitted by Dr. Mary Flanagan NOVEMBER 21 2014

Tiltfactor Research Laboratory

22 Lebanon Street

246 Black Family Visual Arts Center

Dartmouth College

Hanover, New Hampshire 03755 USA

(603) 646-1007

Website: <http://www.tiltfactor.org>

Contact: Mary.Flanagan@Dartmouth.edu

Research funded by the IMLS

INTERIM FINANCIAL REPORT

The intention of the Interim Financial Report is to document expenses that have been invoiced to the Garden between **12/1/13 and 10/31/14**. Todd J. Berube of Dartmouth has sent on the financial report on behalf of Dartmouth.

INTERIM NARRATIVE REPORT

Tiltfactor at Dartmouth College is a Subawardee as part of “Purposeful Gaming and the Biodiversity Heritage Library: Playfully Crowdsourcing Transcription” project. Funded by a grant by the Institute of Museum and Library Services (IMLS), the aim of the work is to test new means of using crowdsourcing and gaming to support the enhancement of texts from the Biodiversity Heritage Library (BHL). Tiltfactor is involved in the design and creation of two online transcription games whose aim is to collect quality data and ultimately improve access to digital texts through crowdsourced data enhancement tasks for data found within the BHL’s manuscript collections.

PROJECT STATUS

Since the start of the project, the games and backend infrastructure have been designed in phases and presented to BHL and its community. The two games we have designed are what we call the “Altruism Game,” referred to below by its working title *Beanstalk*, and a game that is focused on a game-playing constituency--a “Gamer Game/Transcription Tool”-- referred to below by its working title *Smörball*.

1) Scope of Work Item: Game Development Kickoff

Status: Completed July 31st 2014

The game development kickoff occurred in July 2014 with the Tiltfactor and Missouri Botanical Garden (MOBOT) teams. Since the time of the kickoff, meetings have been held biweekly or upon either team’s request. During the kickoff and subsequent meetings in the following months, the teams worked together to lay the foundations that subsequently shaped the nature of the games, including:

- Choosing target audiences and devices;
- Deciding on how to handle special case data such as special characters and human-transcribed data;
- Determining data import and export formats.

During this time Tiltfactor pursued the following activities:

- Brainstorming and testing at least five game concepts and narrowing down the efficacy and playability benefits to two games;
- Manually creating data files which approximated the actual digital text data that would be provided by BHL in order to test game prototypes;
- Developing and testing several models for estimating player input accuracy

The deliverable was approved.

2) Scope of Work Item: Design Creation

Status: Completed September 30th 2014

During September 2014 Tiltfactor iteratively developed and playtested rough prototypes of the games that eventually became *Beanstalk* and *Smörball*. Rough prototypes were provided to the MOBOT team for review. Tiltfactor also conducted special focus groups with a diverse range of players that included frequent and infrequent game players both within and outside of BHL's community. On September 30th Tiltfactor delivered complete game design documents for two games, as well as initial graphics, and pre-Alpha-stage functional game prototypes to the MOBOT team for discussion and approval.

The deliverable was approved.

See the Appendix to this report detailing the designs.

3) Scope of Work item: Backend Structure

Status: Completed October 31st 2014

The project's Backend Structure, that is, its core database and data storage functionality, was produced and demonstrated to the MOBOT team for feedback. During development, Tiltfactor worked with the MOBOT team to ensure that the database import and export formats met BHL's needs. Key considerations for the backend structure included: minimizing possible data corruption by hackers, maximizing efficiency, and ease of importing new page and book data.

The deliverable was approved.

Note: On November 17th 2014 Tiltfactor received the initial output of BHL's current transcription systems. While the data provided was in the format agreed upon between MOBOT and Tiltfactor, the quality shift from previous conversations and data examples marks a departure from what was expected. For example, word locations in this sample data are unreliable, and the majority of manuscript words were provided with only one transcription instead of the previously discussed two transcriptions. A meeting with MOBOT on November 21st about the issue led to a discussion of either a possible exclusion of problematic OCR pages or a possible push back in the SoW deliverable schedule, to be determined by Thanksgiving 2014. We are in discussion with MOBOT about this OCR output quality issue.

4) Scope of Work item: Alpha Stage Game Prototypes.

Status: Ongoing

We have assembled a team to do production and are waiting on the data quality issue to commence full on production. See the next report in 2015 on the completion of the project.

BHL INTERIM REPORT

APPENDIX 1: GAME DESIGN DOCUMENTS



INTRODUCTION

Tiltfactor at Dartmouth College is a Subawardee as part of “Purposeful Gaming and the Biodiversity Heritage Library: Playfully Crowdsourcing Transcription” project. The aim of the work is to test new means of using crowdsourcing and gaming to support the enhancement of texts from the Biodiversity Heritage Library. Tiltfactor designed two online transcription games whose aim is to collect quality data and ultimately improve access to digital texts through crowdsourced data enhancement tasks for data found within the BHL’s manuscript collections. These games are described in detail below.

Table of Contents

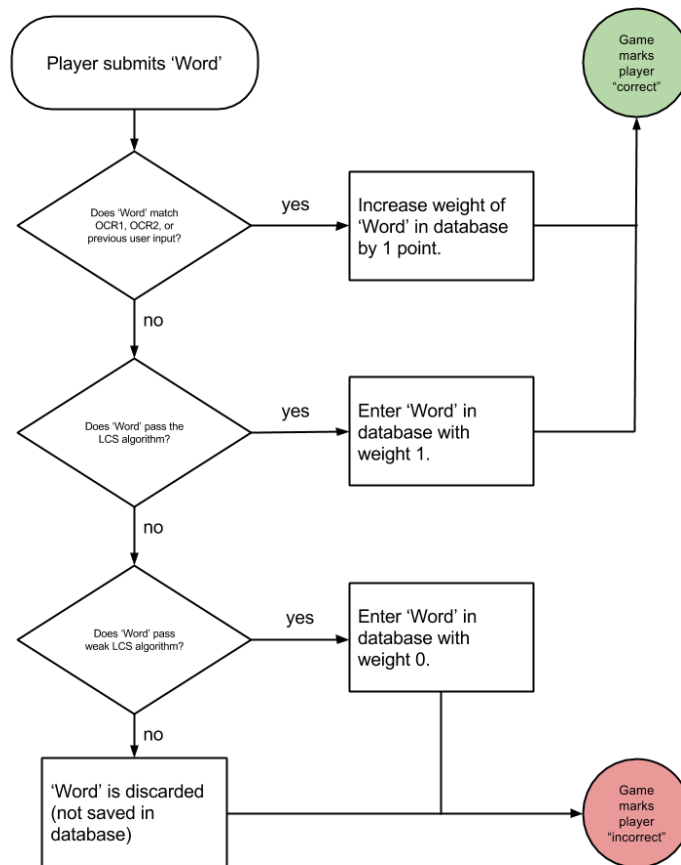
Data Gathering	2
Saving User Input to the Database	2
The Longest Common Substring Algorithm	4
Handling Special Characters and Edge Cases	6
Game #1 - <i>Smörball</i>	8
Description	8
Gameplay and Game Mechanics	8
State of the Prototype	10
User Experience	11
Game #2 - <i>Beanstalk</i>	12
Description	12
Gameplay and Game Mechanics	12
State of the Prototype	12
User Experience	13

Data Gathering

At its core, the Purposeful Gaming and BHL project is about gathering accurate OCR verification data. Implementing crowdsourcing games that collect high quality data is always a challenging task; games must have the ability to reward players for correct responses and punish players for incorrect ones, but crowdsourcing applications by definition do not know the correct answers to their problems. To this end, both Game #1 and Game #2 rely on the same core mechanic: players type a word as seen in snippets of facsimiles, and their responses are verified by comparing them to the OCR output, previous players' responses, and using the Longest Common Substring algorithm, as described in the sections below. This allows the games to distinguish between correct and incorrect input for the purposes of rewarding or punishing the player.

Saving User Input to the Database

To actually collect reliable player input, the game system will use a three-tiered approach, shown below.



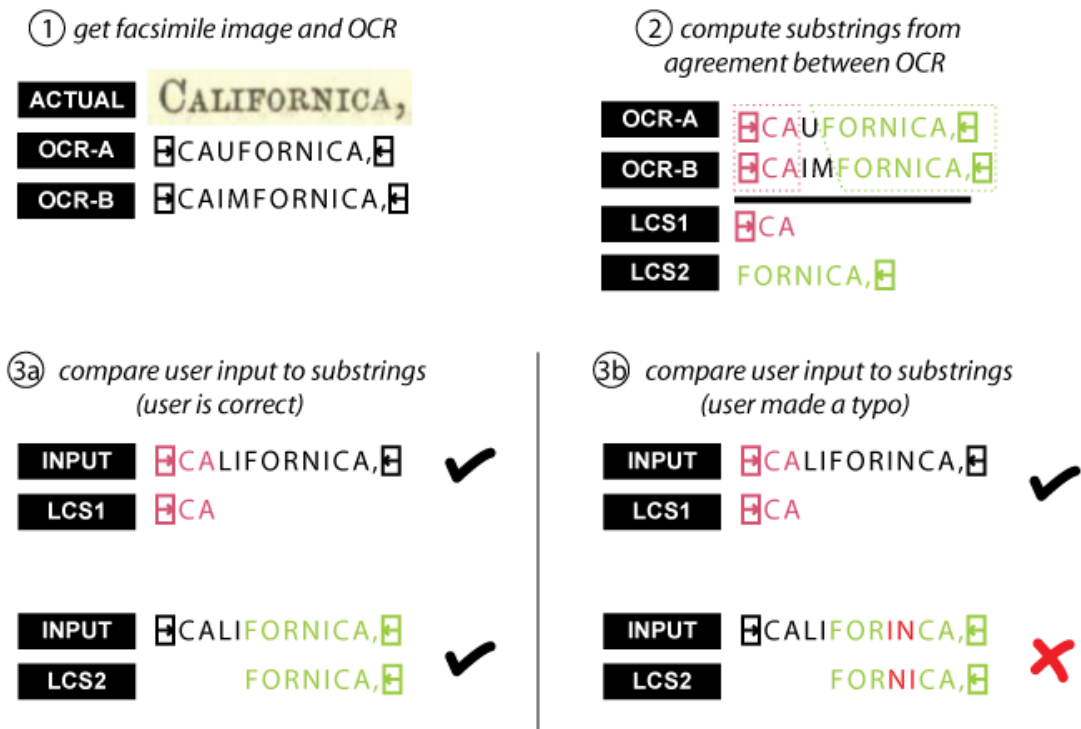
Upon submission of a word, the game's first check is to compare that word to both OCR outputs, and any previously accepted player submissions. If the player's word *does* match, that entry in the database has its score increased by 1, which will eventually be used to select the most accurate answer. This check serves two purposes. First, it reduces tax on the game, as in approximately 50% of all OCR disagreements, one of the OCR outputs is correct. Second, this check begins to mediate the few cases in which the LCS algorithm marks correctly typed words as incorrect (see *The Longest Common Substring Algorithm*, below). Since previous players' submissions are also possibilities for this check, even in cases when neither OCR is correct, *and* the LCS algorithm fails, players will still often have their correct answers marked as correct. In essence, in every case where neither OCR option is correct *and* the LCS algorithm fails, one player will be frustrated by being marked 'incorrect' when she should be marked 'correct' for that word once, and all future players will be rewarded correctly.

If the player's word matches neither the OCR output nor previous submissions, the game system runs the word through the LCS algorithm. This is particularly relevant in the approximately 46% of cases where neither OCR output is correct, for OCR disagreements that haven't been played before. In these cases, if the player's word passes the LCS algorithm (described in detail below), it is inserted into the database as a new option for future players to input, and the player is marked 'correct.'

If the player's word matches none of the above, it is checked against a weaker LCS algorithm, which allows for a specific number of character discrepancies from the LCS. If it passes *this* algorithm, then one of two things has happened: either the player made a small typo, or the original LCS algorithm failed (a 4% chance). Because it might be a typo, the game marks the player as 'incorrect.' However, because it's possible they typed the right word and the LCS algorithm failed, their submission is still entered into the database to see if future players agree with it. If their word *fails* the weak LCS algorithm, however, it bears no relation to the OCR output and can be safely discarded as a garbage word. This also prevents players from 'seeding' the database with garbage words to attempt to ensure future matches.

The Longest Common Substring Algorithm

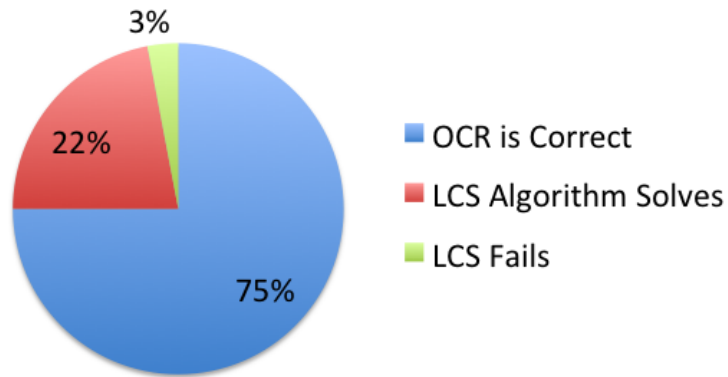
Both Game #1 and Game #2 use a modified Longest Common Substring (LCS) algorithm to infer accuracy of player input. Simply put, this algorithm compares the two OCR outputs to find character sequences upon which they agree, and requires the player's input to contain those sequences in order to mark the player as correct. In the diagram below the \square character refers to the start of a word, and the \square character refers to the end of a word.



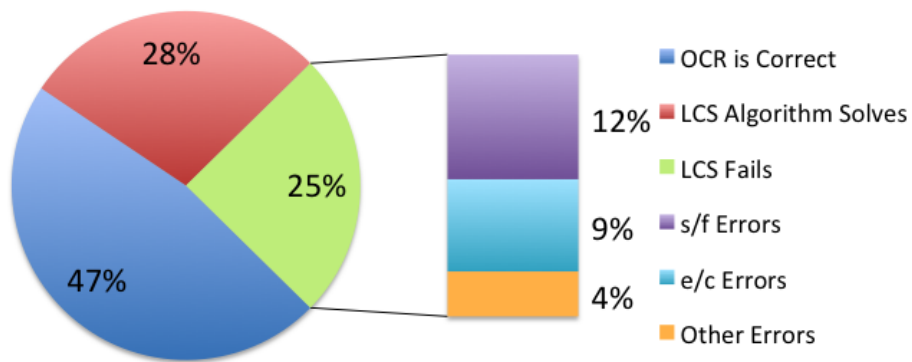
To verify that the LCS algorithm functions as intended, the Tiltfactor team tested it against over 500 sample errors, investigating the frequency of *true negatives*: occurrences when the user typed the word correctly but the algorithm didn't recognize it was correct. These true negatives are particularly problematic because they have the potential to frustrate the player immensely, and occur when *both* OCR make the same error. This happens almost exclusively when one character looks very much like another, for example 'e' and 'c' in fonts where the e's connector is very thin.

As shown below, in the 'good' (fairly OCR-readable with straight lines, dark ink, few stray marks, and good quality scans) sample data the LCS algorithm accepted the correct answer when typed by the player for 97% of all OCR disagreements. In the less OCR-readable sample data (caused by stray marks, water-warped paper, etc.), however, the algorithm generated true negatives for 25% of the OCR disagreements.

Error Types in 'Good' Sample Data




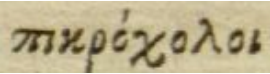
Error Types in 'Bad' Sample Data



While less than stellar, upon further inspection the cases in which the LCS failed turned out to be primarily composed of 2 very specific types of problems: when both OCR software read an 'e' as a 'c', and when they both read a 'long s' ('r') as an 'f'. Adjusting the algorithm to allow players to input 'e' instead of 'c' and 's' instead of 'f' brings the number of true negatives across all sample data to a very acceptable 3.5%. True negatives in the LCS algorithm are further mediated by checking against previous players' input (as described in *Saving User Input to the Database*, above).

Handling Special Characters and Edge Cases

While the vast majority of OCR disagreements encountered will be easily parsable by both the game system and the players, special characters and other edge cases are expected to occur, albeit infrequently. The table below outlines predicted edge cases and how the games and the game system will handle these cases.

Special Case	Example	Solution
Capitalization	Larrea → Larrea LARREA → LARREA	Players will be required to type word with proper capitalization, as enforced by the LCS algorithm.
Accents and Modifiers	á → a Ñ → N	Players will be instructed to simplify accented characters to non-accented equivalents, <i>if they cannot type those characters</i> . Players who can type those characters will be encouraged to do so. Both of these types of input will be accepted by the LCS algorithm
Dashes	-- → - — → -	The game system will simplify all dashes present in the OCR to hyphens (regular '-' character)
Bold, Font Weights	word → word <i>word</i> → word	The game system will remove all font weighting.
Ligatures	æ → ae  → ct	Players will be instructed to simplify ligatures to individual character equivalents, <i>if they cannot type those ligatures</i> . Players who can type those ligatures will be encouraged to do so. Both of these types of input will be accepted by the LCS algorithm
Characters that require specialized knowledge	ƒ → f/s □ → 2/R	Players will be instructed to type the character to the best of their ability, as enforced by the LCS algorithm.
Non-Latin alphabets	 → ∅	Players will be instructed to use a limited pass system to skip words they simply cannot type. After a requisite number of passes a word will be flagged as untypable, and will no longer be served to players.

<p>Lumped Words</p>	<p>vixquemquam → vix quem quam</p>	<p>Players will be instructed to type multiple words with spaces in between, as enforced by the LCS algorithm. These errors will be returned as a single entry with the previously lumped words split by spaces.</p>
<p>Length</p>	<p>RicinusAmericanusmajor&minor. → ∅</p>	<p>The game system will cap length of words served by the games. All words of over a certain pixel length will not be served to players</p>

Game #1 - Smörball (for Gamers)

Description

In *Smörball* you play the coach of the Harrisburg Melonballers, a fictional team that competes in the fictional and goofy amalgam sport of Smörball. Through shrewd tactics and excellent typing ability, you must coach, command, and direct your players to victory! Your team plays defense in every game of Smörball, and must prevent the opposing players from getting across the field to score in your endzone. As the coach, you must shout (type) commands to your players to tell them when to tackle their opponents. The fewer points the opposing team scores in each match, the better for you (and your accumulated winnings)!



Smörball is designed for players who regularly play casual mobile games and beyond, and players who are game-motivated rather than altruism-motivated. The game is intended to be played over multiple sessions, and will save progress locally on the user's computer.

Gameplay and Game Mechanics

Levels - Smörball is a level-based game. Each level represents one sports match in a national series, leading up to the final match of the season: the Smör Bowl. Over the course of the first 5-10 levels players learn the basics of gameplay, and over the following levels they are challenged to apply the game mechanics to succeed. The levels are intended to give the game-motivated players a sense of direction and to retain players at least through the course of all of the levels once, with the potential for replaying each of those levels multiple times. At the end of each level, players are scored on how well they did, and this score directly contributes to the currency they accumulate to spend on upgrades. This mechanic lends the game replayability, as players can revisit earlier levels to complete them with a better score.

Light Narrative - Over the course of the levels, *Smörball* tells the player a simple story of their team's victory, leading up to the last level, the Smör Bowl, and claiming the Sigrid Cup.

Enemies - During each level, *Smörball's* player (the coach) encounters the opposing team's athletes, whom she must stop from scoring by typing commands to her own athletes. Over the course of the game new types of opposing athletes are introduced, and the coach must adapt to learn how to best take them down. The various types of opposing athlete include: "fast forwards" (quick enemies), "walking-backs" (slow enemies), "tallstops" (standard enemies), "wide centres"

(strong, slow enemies), “wingers” (who switch lanes), and “cheerleaders” (who speed up other enemies). Most enemy types vary with speed, and number of hits it takes to tackle them.

Powerups - Over the course of the game several types of ‘powerups’ (special plays) are introduced, which can be used by the player at opportune moments to empower their athletes and more efficiently stop the opposing athletes. When to use these powerups is an important tactical decision that players will practice to become more adept at. These special plays include: The Rush (instantly tackles bigger opposing athletes), Up The Middle (sends an athlete all the way down one row), and The Sweep (sends athletes down every lane).

Upgrades - At the end of each level, players are given winnings based on how well they did. Each level has a maximum payoff (if the player manages a shutout), and the winnings go down for each point the opposing team scores against her. If replaying a level, a player can only receive the difference between her previous payoff and her new payoff, and only if positive (if on the first play she received \$5000 on the second play she played perfectly, worth \$6000, at the end of the level the player gets +\$1000).

After several levels of play, the player is introduced to the Clubhouse, in which they can use their winnings to purchase upgrades. Upgrades include: the ability to start each level with one of the powerups, more frequent powerup spawning, increased knockback, increased damage on long words, shorter cooldowns on mistakes, and more.

Variable Difficulty - Flow theory states that in order to keep players engaged, designers must set “challenges--tasks that are neither too difficult nor too simple” for their users’ abilities.¹ This poses a particular problem in *Smörball*, as players’ typing abilities can vary wildly. To compensate for variation in players’ typing speeds, *Smörball* will adjust its challenge in two ways:

First, opposing players are organized into “waves.” Each wave is composed of one or more enemy, timed to appear on the screen at the same time or in quick succession. Each of these waves occurs in sequence when the previous wave is dispatched, and not before. This timing ensures that gameplay never becomes boring through long spaces between enemies, and also never becomes too difficult by having multiple waves on the screen at once.

Second, to account for difficulty *within* each wave, *Smörball* uses a hybrid automatic and manual difficulty system. During the first level, the game measures players’ typing and gameplay proficiency and sets a recommended difficulty: very easy, easy, normal, hard, or very hard. This difficulty can then be changed manually by the player via the menu screen. During gameplay, timing of opponents’ arrival on the screen within any given wave is adjusted based on the game

¹ Csikszentmihalyi, M. *Flow: The Psychology of Optimal Experience*. Harper Perennial, London, 1990.

difficulty, with “very easy” providing long delays between opponents and “very hard” providing very small delays between opponents.

State of the Prototype

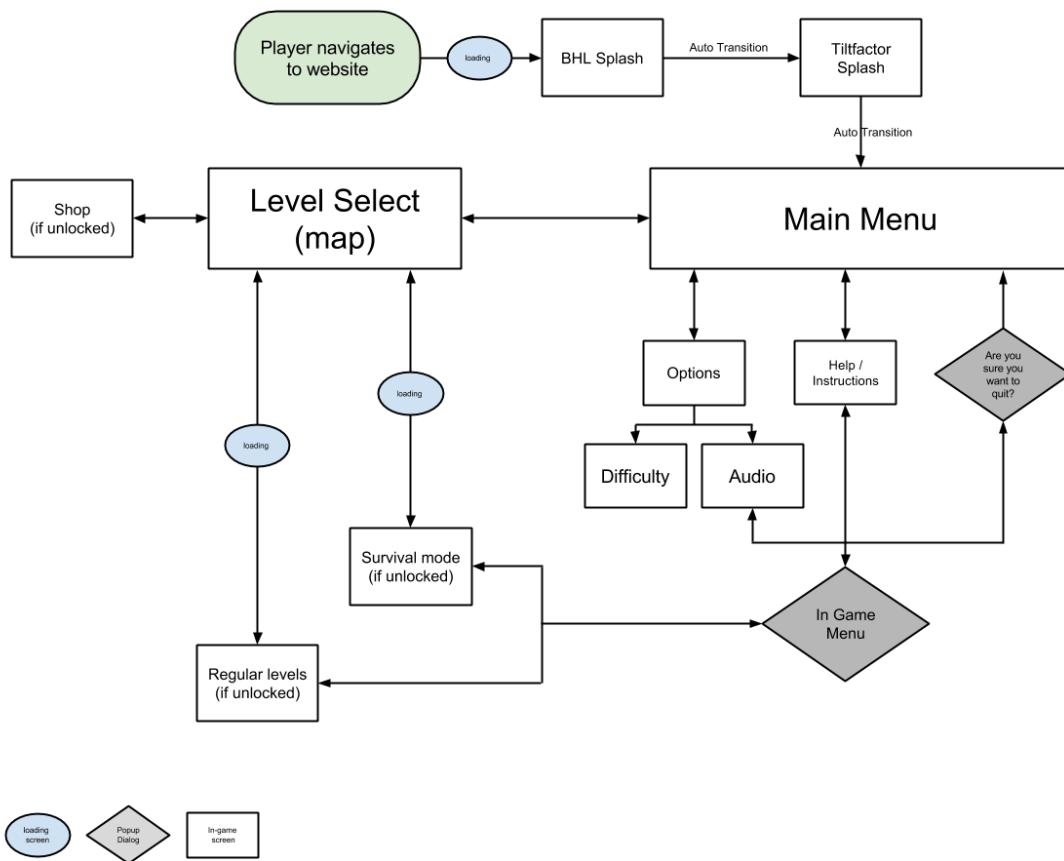
The current prototype implements the rough mechanical functionality of *Smörball* with placeholder graphics and no narrative for playtesting purposes. The prototype demonstrates the core functionalities of the game: unlockable levels, core gameplay within those levels, different enemy types, powerups, and variable difficulty. The currently playable levels serve as proofs of concept for the final progression of levels. The final levels will likely be longer and much more numerous than the test levels.

The current prototype can be found here: <http://tiltfactor1.dartmouth.edu/test/bhl-game1>. Please note that this is a live development build, and because it is constantly being updated may contain bugs and change wildly (or even stop functioning) from one visit to the next.



The Tiltfactor team is experimenting with different art styles to decide which to pursue for the final game. Above is a sample of potential opposing *Smörball* players sporting a mishmash of sports equipment and uniforms, illustrated in different styles.

User Experience



The chart above maps the user's top-level navigation through *Smörball's* game and menu screens.

Game #2 - Beanstalk (for non-Gamers)

Description

In *Beanstalk*, players tend a small plant that must be spoken to in order to grow. As the player types words correctly, leaves and flowers sprout from the beanstalk. After 8 leaves have sprouted, the stalk grows a new segment with no leaves, and the count resets. Should the player make a mistake while typing a word, all the leaves they have typed so far on this segment wither and fall off. The plant starts off as a seedling, growing among blades of grass, and over the course of the game grows past bushes, trees, skyscrapers, clouds, and eventually becomes a massive beanstalk swaying among the stars. When the player reaches the stars her beanstalk drops another seed, and she can start a second stalk.

Beanstalk is designed for altruism-motivated players whose game experience ranges between 'zero experience' and 'some experience in the past, but no recent experience.' The game is intended to be played over multiple sessions, and will save progress to the user's account (if signed in), or locally on the user's computer (if not logged in).

Gameplay and Game Mechanics

No-Stress Gameplay - *Beanstalk* is an extremely light game with no time constraints. Players are simply visually rewarded for contributing their time to improving BHL's collections.

Mobile and Desktop - Because of lack of time constraints and small screen real estate, *Beanstalk* can be played on phone and tablet browsers as well as desktop browsers.

Competition - For players who are so inclined, *Beanstalk* features leaderboards to compare the height of their plants with other players', as well as other metrics like the fastest-growing plants in the past week, and fewest missed words, etc.

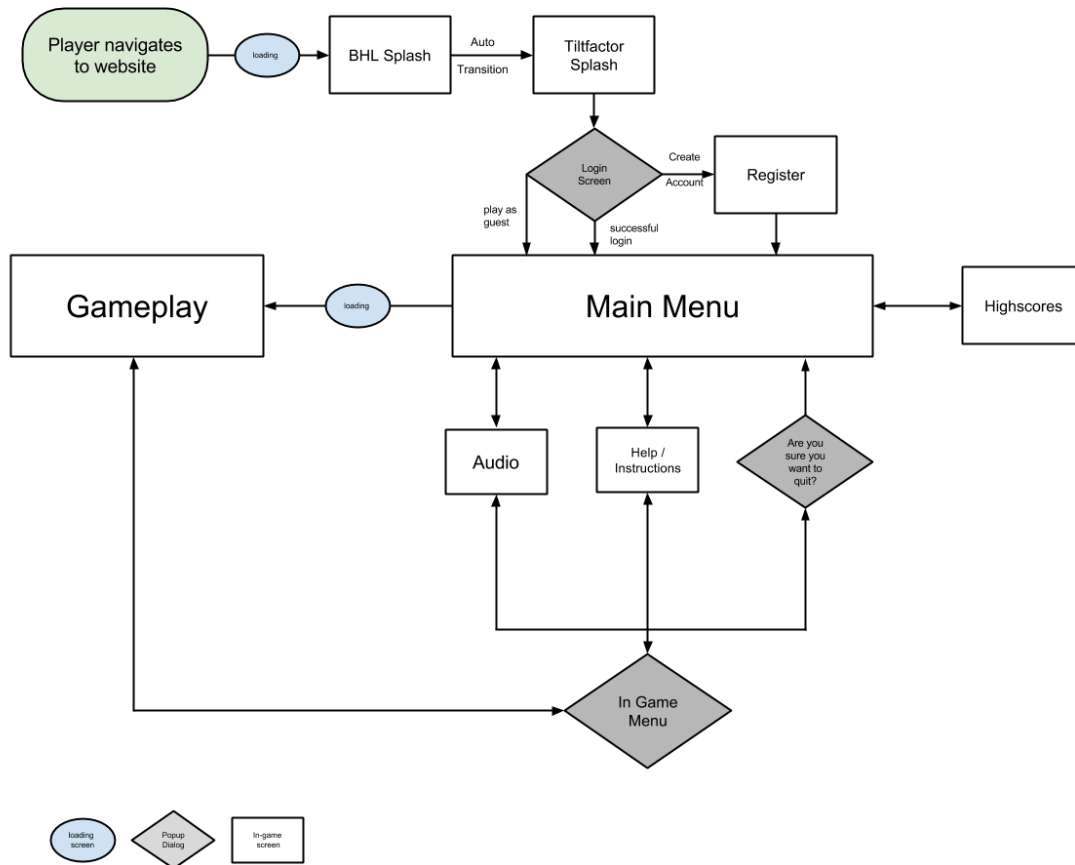
State of the Prototype

The current prototype implements the rough mechanical functionality of *Beanstalk* with placeholder graphics for playtesting purposes. The prototype demonstrates the core functionalities of the game: a growing plant, and a changing background.

The current prototype can be found here: <http://tiltfactor1.dartmouth.edu/test/bhl-game2>. Please note that this is a live development build, and because it is constantly being updated it may contain bugs or change wildly from one visit to the next.

Over the course of the next few weeks Tiltfactor's illustrators will be mocking up more finalized graphics for *Beanstalk*.

User Experience



The chart above maps the user's top-level navigation through *Beanstalk's* game and menu screens.