Comp 388/488 - Game Design and Development

Spring Semester 2018 - Slides - week 14

Dr Nick Hayward

Games - Complete and Functional

check for unintentional features

- another consideration on our way to a game's completeness
  - *the hunt for unintentional features in gameplay and features*
- looking for a flaw in our game
  - *a player may exploit for an unfair or unwarranted advantage*
- player manipulation of a game is crucial to the experience
  - *in essence to simply win a game*
- unintentional features often occur to the detriment of the game itself
  - *often ruins the sense of play in the game*
  - *may prejudice one player over another in certain environments*
- game may not be considered complete whilst such issues persist
- many players enjoy tracking and recording such unintentional features
  - *may become a fun aspect of the gaming experience*
- well-known example arose in the game *Deus Ex*
  - *originally released in 2000*
- *Deus Ex* includes an explosive weapon, a grenade
  - *a player could attach this weapon to a wall*
  - *then use them as a make-shift ladder to climb walls*
  - *also able to climb to unintended places on the game's map*
- as a result, certain levels became considerably easier
  - *less challenging and interesting than originally intended*

Video - Gaming Issues

Deus Ex climbing

Games - Complete and Functional

unintentional vs intentional features

- unintentional issues may pose problems for game designers
  - *they may also become intentional features to many players*
- quirks and perceived issues may still become a benefit
  - *perceived as an asset to the underlying gameplay*
- again, consider the *spawn camping* problem
  - *many examples online of gamers who like this type of game feature*
  - *e.g. many Rainbow Six Siege players are in favour of this feature*
- consider MMORPGs and role of players
  - *often such games do not include a clearly defined ending*
  - *create a sense of community*
  - *foster a long term social setting for players.*
- for MMORPGs, many players dislike killing in the game
  - *malicious killing of other players discouraged*
  - *such player killers seen as detrimental to fun, harmony, enjoyment...*
- developers continue to modify such online worlds to discourage **player killers**
  - *various options in games such as Asheron's Call, EverQuest, Ultima Online...*

Python and Pygame - Game Example I

fun game extras - scale explosion images - basic scale

- still a lingering issue with these collisions and explosions...
- explosions are not reinforcing the gameplay for our shooter style game
  - *no differentiation in the relative size of an explosion*
  - *no semblance of feedback to our player*
- one option to this issue
  - *perhaps add standard scale transform to image for each explosion sprite object*

```python
# explosions
explosion_imgs = []

# iterate over explosion images in directory
for i in range(9):
    file = 'explosion{}.png'.format(i)
    # load image from os
    expl_img = pygame.image.load(os.path.join(img_dir, file)).convert()
    # set colour key for image
    expl_img.set_colorkey(BLACK)
    # append to specified list for explosion images
    explosion_imgs.append(expl_img)
```

- render a smaller, less overwhelming explosion for each collision

Python and Pygame - Game Example I

fun game extras - scale explosion images - relative scale - part I

- useful to be able to scale these explosions relative to the actual size of a given sprite object
  - *e.g. a smaller relative explosion image for a smaller mob object*
  - *or, a relatively sized explosion against the player's ship*
- update our class for the `Explosion` object
  - *dynamically modify each explosion image in the animation relative to a specified size*
- scale each frame of explosion animation to match the size of the collison object, e.g.

```
# create a generic explosion sprite - use for asteroids, player explosions &c.
class Explosion(pygame.sprite.Sprite):
    # initialise sprite
    def __init__(self, center, size):
        pygame.sprite.Sprite.__init__(self)
        # specify size for explosion sprite
        self.size = size
        # get initial image for explosion
        self.image = pygame.transform.scale(explosion_imgs[0], self.size)
...
```

- start by adding a parameter for `size`
  - *pass a variable size for each collision object*
- use this size to scale the initial image for the explosion animation

Python and Pygame - Game Example 1

fun game extras - scale explosion images - relative scale - part 2

- each frame of the animation will also require scaling of the explosion image, e.g.

```python
# change image as time progresses for explosion sprite
def update(self):
    # get current time
    now = pygame.time.get_ticks()
    # check if enough time has passed between animations
    if now - self.last_update > self.frame_rate:
        self.last_update = now
        # if enough time passed - add 1 to frame
        self.frame += 1
        # check if end of explosion images reached
        if self.frame == len(explosion_imgs):
            # kill if end of image reached
            self.kill()
        else:
            center = self.rect.center
            self.image = pygame.transform.scale(explosion_imgs[self.frame], self.size)
            # update rect for image
            self.rect = self.image.get_rect()
            self.rect.center = center
```

- as we output each frame of the explosion animation
  - *scale this image to match the passed `size` for the explosion object*

Python and Pygame - Game Example 1

fun game extras - scale explosion images - dynamic collision size

- different size mob objects will have a matching explosion animation
  - *update in the game loop, e.g.*

```
# add check for sprite group collide with another sprite group - projectiles hitting enemy objects - use True to delete sprites from ea
collisions = pygame.sprite.groupcollide(mob_sprites, projectiles, True, True)
# add more mobs for those hit and deleted by projectiles
for collision in collisions:
    # calculate points relative to size of mob object
    game_score += 40 - collision.radius
    # play explosion sound effect for collision
    explosion_effect.play()
    # get size of collision object
    col_size = collision.rect.size
    #print("collision size = " + str(col_size))
    # add animation for explosion images if collision
    explosion = Explosion(collision.rect.center, col_size)
    # add explosion sprite to game sprites group
    game_sprites.add(explosion)
    # create a new mob object
    createMob()
```

- same for the player's object...

resources

- notes = extras-part1-explosions.pdf
- code = objectexplosions2.py

game example

- shooter1.2.py
- add some fun explosions
  - *create sprite object for explosion*
  - *cycle through images to create explosion animation*
  - *add explosion for each collision*
- extra explosions
  - *explode a player's ship for a collision*
- scale explosions
  - *rescale and size explosions in game window*

Video - Shooter 1.2 - Part 3

scale explosions

Games - Complete and Functional

Asheron's Call

- in *Asheron's Call*, which finally finished on 31st January 2017
  - *designers originally created a system of allegiance and fellowship*
  - *new players were given the chance to swear allegiance to another player*
  - *might receive protection, money, weapons &c. in the game*
  - *this relationship became bi-directional with each player gaining...*
- further mods introduced to *Asheron's Call*
  - *e.g. prevented a player from directly attacking another playe*
  - *also modified the underlying story for this game*
  - *provided players with a share of magic and protection of Dereth*
- some players founds this too limiting - they missed being *player killers*
  - *they saw the game as overly boring, lacking in challenge &c.*
- further modification was added by the developers
  - *allowed players to voluntarily convert to player killer status*
- happy medium achieved in this game for many players
  - *through testing and feedback*
  - *by default, players were free from the threat of being killed*
  - *player killers could engage each other*
- a great social community grew up around this game
  - *until its servers were finally closed down earlier this year*

Video - Gaming Issues

Last moments of Asheron's Call

Games and development

quick exercise - part 1 - Reminder

Choose at least one of the following games,

- Asheron's Call
- Asteroids
- Deus Ex
- Journey
- Mario Kart

or use your own game idea and concept.

Then consider the following questions:

- what is the minimum you consider necessary for this game to be functionally complete?
  - *in effect, ready for initial testers and players?*
  - *as a designer and developer, which aspects of the game would you leave open to change during testing?*

quick exercise - part 2

Choose at least one of the following games,

- Asheron's Call
- Asteroids
- Deus Ex
- Journey
- Mario Kart

or use your own game idea and concept.

Then consider the following questions:

- how do you integrate these changes into your game before publication?
  - *i.e. chosen aspects of the game left open to change during testing...*
- can you identify unintentional features and issues that might arise from knowledge of similar games?

Games - Complete and Functional

a dead end

- another type of flaw or issue that may occur in our game
    - *a perceived dead end in gameplay, functionality...*
- a noticeable difference when compared with unintentional features
    - *do not allow players to gain an advantage or possible exploit in the game*
- a *dead end* is something we need to fix as quickly as possible
- developers of adventure games commonly encounter this issue
    - *Zork a bench mark example*
    - *many such as Sierra's King's Quest, Space Quest...*
- for some players, dead ends have become a nostalgic recollection
    - *they became a part of the expected gameplay for original adventure titles*
- we may start to consider a game as complete and functional
    - *internally complete*
    - *player can operate a game without compromising gameplay or functionality*
    - *considered an objective and subjective question*
    - *game is not complete - room for improvement*
    - *lingering issues, flaws, dead ends...*

Video - Complete and Functional

Dead End Dancer in King's Quest

Games - Systems and Balance

concept of balance

- balance is a concept in game design that is regularly mentioned
  - *but often misunderstood or poorly applied...*
- a common underlying issue with this concept of balance for game design
  - *we may abstract this concept*
  - *but it still needs to be applied in specific cases...each and every time*
- to begin balancing the design and development of a game
  - *begin by ensuring that it meets the specified player experience goals*
- we're checking the breadth and scope of such goals
  - *i.e. have they been met relative to the game's complexity?*
- also checking for any unnecessary or undefined results
- such checks and balances may also be influenced by the players themselves
  - *e.g. single player versus multi-player options*
- for *multi-player* options, we may need to ensure
  - *a game's resources, gameplay, and goals are even as the play begins*
- for *single player* options, we may simply consider
  - *a balance between a player's skill level and the game's challenge*
- issues of balance can become problematic for designers and developers
  - *consider requirements of such goals relative to our own preferences, desires...*
- very nature of trying to balance many divergent elements within a complex system
  - *has potential to create many headaches for designers and developers*

Game designers

Designer example - Rob Pardo on design and balance

- recurrent use of iterative design for each game title and series at Blizzard
  - *key aspects of game design for Pardo...*
- iterative modification of game variables
  - *a key factor to the success of their designs*
  - *helps strike a balance in the way the game performs and plays*
- this process continued for their titles right up to the initial release
  - *new spells in WarCraft III just before public release*
- holes and options still remain open in public beta
  - *allows suggestions from testers, pro gamers to be integrated*
- testing and iterative design continues long after a game's initial release
  - *StarCraft testing and development continued for 2 years after release*
- this included perceived imbalances in the game
  - *patches or updates &c. to reflect loopholes and glitches*
- any of these issues were discovered and shared by the gamers
  - *then required updates to re-balance the game*
- Pardo was also proactive in creating a new role at Blizzard
  - *for analysing and monitoring online player behaviour and usage*
- this so-called *game balance designer*
  - *might check statistics and patterns recorded for a given game*
  - *then start to test adjustments for applicable part of the game*

Games - Systems and Balance

symmetry

- initially, symmetry in gaming is a simple concept
- give each player the same resources and conditions as they game begins
  - *along with information about the story, gameplay &c.*
  - *a game should be symmetrical...*
- classical examples of initial symmetry include draughts, chess...
  - *many turn-based examples include initial symmetry*
- symmetry is a particularly useful concept
  - *we may modify as necessary to create interesting and fun games*
- a few changes here and there to such perceptions of symmetry
  - *the nature of a game may be easily changed and updated*
- for symmetrical games such as draughts...
  - *still the potential for loss of symmetry*
  - *e.g. who gets to move first?*
  - *such a game may become asymmetrical quickly*
- may negate any perceived advantage of moving first
  - *e.g. chess limits first move to a pawn or knight*
- such moves are rarely game changing
  - *potential still remains for challenge for an expert player*
- a similar option to maintain and persist symmetry
  - *we may introduce a concept for chance elements in a game*
- benefit of reducing the potential for one player to dominate gameplay
  - *may reduce unintentional effects of starting first in a symmetrical game*
- chance elements may include, e.g.
  - *random options, scaled variants, emergent systems...*
- trying to ensure there is reduced potential for biased gameplay

Games - Systems and Balance

asymmetry

- we may also offer our players asymmetry, e.g.
  - *varying attributes, abilities, resources...*
- also vary a game's rules, and its underlying objectives
  - *to fit different players' roles and requirements*
  - *game has switched to become asymmetrical in nature*
- a perceived fundamental characteristic of such asymmetry in games
  - *need to maintain a balance of fair gameplay for each player*
- racing games, such as Mario Kart, are great examples
  - *e.g. variant attributes, skills, and perfomance for each kart and character*
  - *creates balance for a player relative to skill levels and experience*
- each player should still retain the potential to win the game
  - *regardless of the variant, asymmetrical factors...*
- asymmetry becomes a useful option for us as designers and developers
  - *e.g. creating games that model behaviour, stories, and gameplay on real life examples*
  - *such examples will commonly be asymmetrical*
- vast majority of video games are asymmetrical
  - *e.g. RTS games such as Command & Conquer*

Games - Systems and Balance

create a balance

- for most games we create and complete
  - *aiming for a balance between challenge and player's skill level*
- naturally vary from player to player
  - *for most instances we're clearly aiming at a middle ground*
  - *creating a median skill level*
- already seen examples of *classes* in **Diablo**
  - *different players may assume varied roles in this game*
- consideration of player skill levels in **Civilization**
  - *uses varied levels of difficulty*
  - *includes certain defaults for properties and values*
  - *e.g. cash reserves vary from chieftain to emperor*
- **Civilization** uses varying skill levels
  - *helps to customise properties and attributes in the game*
- balancing a game for *median* skill level requires extensive testing
- testing each game to see where the balance lies for such properties and attributes
  - *customary to start with more experienced, hard-core players*
  - *start at perceived highest skill level*
  - *helps set high mark for the game's skill levels*
  - *then test and set beginner, lowest skill level*
- use high and low boundaries for skill levels
  - *becomes easier to test varied properties and attributes*
  - *keep testing until median is established and set*
- skill levels need to be considered relative to a game's varied stages
  - *customary to incrementally increase difficulty*
  - *whilst reducing difference between skill levels for players*
  - *scale for skill levels starts to shrink...*

Games - Systems and Balance

balance options

- possible to consider balance in a game as a constituent part of the underlying logic
- as a game progresses, we may establish certain conditions
  - *to allow the game to incrementally modify player skill levels*
  - *adapt a game to match a player's skill level*
  - *e.g. as they improve and advance through various challenges modify game*
- examples include *Tetris*, *Gran Turismo*, *Mario Kart*...
  - *Tetris modifies speed of block falling to match a player skill level*
  - *speed becomes a coefficient of difficulty and challenge*
- racing games show subtle modifications to such skill levels and perceived difficulty
  - *Mario Kart introduces a semblance of self-balancing to the racing system*
  - *helps create a fair sense of challenge relative a player's skill level*
  - *a proportional representation between speed and skill for the player and the computer*
  - *e.g. as a player gets faster, computer controlled cars will speed up*
  - *if a player crashes or slows dowm, other cars may slow down*
  - *ensures there is some gameplay left for a particular level or track...*
- balancing creates a sense of challenge
  - *whilst maintaing a semblance of fun and achievement...*

Games - Systems and Balance

modular options for balance

- to help us develop a balanced game consider various parts that constitute the game itself
  - *sub-components that combine to form the game*
  - *rare to design a game as a single, monolithic unit*
- we can start to consider balance in smaller units
  - *customarily relate to smaller, inter-related subsystems*
  - *subsystems that coalesce to form our game*
- we may consider our game as a series of discrete functional units
- by clearly defining each unit
  - *helps us identify its functionality and requirements*
  - *its relations to other units in the game*
- consider a common RPG (role-playing) game as a group of subsystems
  - *e.g. as combat, movement, resources...*
- each subsystem forms a part of the overall game system
  - *may also present obvious issues as we try to balance the system*
  - *e.g. one module or subsystem that is interconnected with another*
  - *changes to one may precipitate an unexpected cascade in another*
- we may start by isolating each subsystem
  - *abstracting their usage and implementation from the whole*
- testing and configuring each subsystem
  - *trying to ensure functional independence from the overall game*
  - *crucial for developing a large scale game...*
- we're following many standard practices for object-oriented programming
- by clearly defining the I/O for each functional unit
  - *able to more effectively analyse and monitor each unit*
  - *i.e. as we balance and maintain the overall game system*

Games - Systems and Balance

balance and focus

- balance may also be derived from a clearly defined sense of purpose
- a game's focus and goals helps set clear requirements and parameters for I/O
  - *whilst helping to define the modular components for the development*
- identification of purpose helps assign a clear usage and structure
  - *to a game's development of underlying modular components*
- consider why you have certain components in your game system
  - *what is a component's purpose?*
  - *is this purpose unique to the component?*
  - *will the game work without this component?*
  - *...*
- a component's purpose needs to be
  - *clear, well-defined, and logical*
  - *suitable for the type of game being developed*
- each component in a game should have a purpose
  - *where possible no component should have more than one primary function*
- e.g. start by considering a game's mechanics
  - *how to dissect them into fundamental parts for the game's requirements*
  - *what is the purpose of such mechanics?*
- by clearly defining such constituent parts we're trying to avoid
  - *a development scenario with a mix of rules and subsystems*
  - *a mess of tangled rules, ideas, options &c.*
  - *e.g. different conflicting ideas, concepts &c.*
- if we then need to modify an aspect of the mechanics
  - *perhaps update or remove an element*
  - *we only need to modify one aspect of the gameplay*
- balancing a game's mechanics, and gameplay by association
  - *becomes more systematic and methodical rather than trial and error*
- this pattern of balance and focus also helps promote
  - *incremental development, modification, and testing*

Games - Systems and Fun

choices

- *fun* balanced against a game's sense of challenge or conflict
  - *helps provide required hooks in a game*
  - *often simply emotional attachment to a game*
- strive to captivate players
  - *helps promote a sense of connection and interest in a game's outcome*
- Sid Meier famously noted,

> *Games are a series of interesting choices (decisions)...*

- often derided as overly simplistic
  - *still a semblance of truth to this sentiment...*
- as a player progresses through a game, they are constantly making choices
  - *some big, others small*
  - *together they help a player make sense of the gaming world*
- as game designers and developers
  - *strive to provide a sense of consequence and meaning to these choices*
- real world experiences also help shape our perception of such choices
- if there is little sense of consequence or feedback to a choice or decision
  - *we start to question its validity and merit*
- such examples start to become a distraction
  - *definitely something we want to avoid in most games*
- start by trying to inform a player
  - *an awareness of potential consequences of decisions and actions*
- e.g. consider introducing a simple dilemma that challenges the player
  - *helps them consider certain choices more carefully*
- calculation of a choice relative to its potential outcome
  - *useful way to challenge our players throughout a game*
- often subtle in nature
  - *it's still a useful option for maintaining interest in gameplay*

quick exercise

Choose at least one of the following games,

- Asheron's Call
- Asteroids
- Deus Ex
- Journey
- Mario Kart

or use your own game idea and concept.

Then consider the following questions:

- what are the various opportunities for challenge and play present in your chosen game?
- what are examples of individual challenges in this game?
- are there any repeating challenges or dilemmas in this game?
- how do these choices or challenges help create a sense of fun in the game?
  - *and, as a consequence, act as a hook for the player*

Games - Systems and Fun

considering choice

- adding choice to a game will often improve competition, challenge...
  - *may also present a hook for our players*
- adding choice to general gameplay without the potential for consequence
  - *may simply remove any chance of player engagement*
- to increase the potential for this player engagement
  - *choice should present opportunity to change or modify a game's direction*
- each choice should present the player with a possibility
  - *a positive or negative outcome*
  - *e.g. to advance player to the end of the game...*
- this becomes the common *risk and reward* strategy
- Meier's comment on *interesting choices or decisions*
  - *encapsulated this concept of a series of choices*
  - *choices that flowed throughout a game*
- in contrast to decision making in books and movies
  - *a player may interactively experience such choices for themselves*
- need to ensure that we provide the right game environment
  - *one that permits such choices and decisions by the player*
- start by simply deciding types of player decisions
  - *e.g. decisions a player must make in a particular game*
  - *perhaps based on puzzles, motor kills, perception...*

Games - Systems and Fun

meaningful decisions

- regardless of the choices offered in a game
  - *need to ensure decisions are meaningful and relevant*
- focus initially on the main objective of the game
  - *then structure your game to help your player achieve this end goal*
- review your game and its choices
  - *check for minor or tangential decisions*
  - *if present, revise game and choices*
- may need to reconsider these decisions and choices
  - *so that they matter to the context of the game*
- a balance should also be struck between the types of choices offered
  - *with the simple intention of creating balance in your game*
  - *e.g. recurring action based choices may get tiresome and annoying*
- consider the narrative structure with its abyss and summit
  - *acts as a good indication of variation in story and gameplay*
- decisions and choices may often follow a similar pattern

Video - Meaningful Decisions

three questions

Mario Kart competition

SNES Mario Kart competition

- playoff between groups in the class
- find the best representative player...

Resources

Demos

- pygame - fun game extras
- objectexplosions2.py
- pygame - Game 1 Example
- shooter1.2.py

Games

- Asheron's Call - Wikipedia
- Witness the last moments of Asheron's Call...
- Call of Duty
- Command & Conquer
- Deus Ex Wiki
- King's Quest
- Rainbow Six Siege
- Space Quest
- StarCraft
- free download
- World of Warcraft

Game notes

- Pygame
- extras-part1-explosions.pdf

References

- Bogost, I. *Persuasive Games: The Expressive Power of Videogames*. MIT Press. Cambridge, MA. 2007.
- Huizinga, J. *Homo Ludens: A Study of the Play-Element in Culture*. Angelico Press. 2016.
- Poundstone, W. *Prisoner's Dilemma.* Touchstone. New York. 2002.

Videos

- King's Quest, Dead End Dancer - YouTube
- The last moments of Asheron's Call - YouTube
- Three Questions - Monty Python and the Holy Grail - YouTube