

Procedural Content Generation

2019-10-28



John von Neumann

Announcements

- Trajectory update
 - Meta: Game AI vs Academic, Graphs + Search
 - Physical Acts: Movement, Steering
 - Decide: FSMs, Plans, D&B Trees, RBS, BBs, Fuzzy
 - PCG: Model, Learn and Generate
- HW6: Behavior trees. Thoughts?
- Exam thoughts?
- HW7 due Sunday, Nov 10
- Free material here:
 - <http://www.gameapro.com/>
 - HTN through example:
http://www.gameapro.com/GameAIPro/GameAIPro_Chapter12_Exploring_HTN_Planners_through_Example.pdf

- Stolen terms (bits, space, scenarios): Procedural Content Generation for Games: A Survey
 - https://course.ccs.neu.edu/cs5150f13/readings/hendrikx_pcgg.pdf
- Search-Based Procedural Content Generation: A Taxonomy and Survey
 - https://course.ccs.neu.edu/cs5150f13/readings/togelius_sbpcg.pdf
- PCG in Games: A textbook and an overview of current research (2016)
 - <http://pcgbook.com>
- <http://pcg.wikidot.com/>

Free on website



About the book

Welcome to the Procedural Content Generation in Games book. This is, as far as we know, the first textbook about procedural content generation in games, aka PCG. As far as we know it is also the first book-length overview of the research field. We hope you find it useful, whether you are studying in a course, on your own, or are a researcher.

Content is king!

PROCEDURAL CONTENT GENERATION

Procedural Content Generation

- Use of computation instead of manual effort to produce elements of gameplay. Why?
 - Design aspects of the game
 - Save development cost (?)
 - Save storage or main memory (“infinite games”)
 - Adapt aspects of the game (player models)
- In what games have you seen this? What’d you think?
- What sorts of problems with PCG? What’s well suited?

What is Procedural Content Generation?

Generating game content (levels, narrative, quests, visuals, sound, etc.) algorithmically.

- Game **bits** (characters, visuals, sound)
- Game **space** (levels, worlds, puzzles)
- Game **scenarios** (story, quests, history)

Game Bits (characters, visuals, sound)

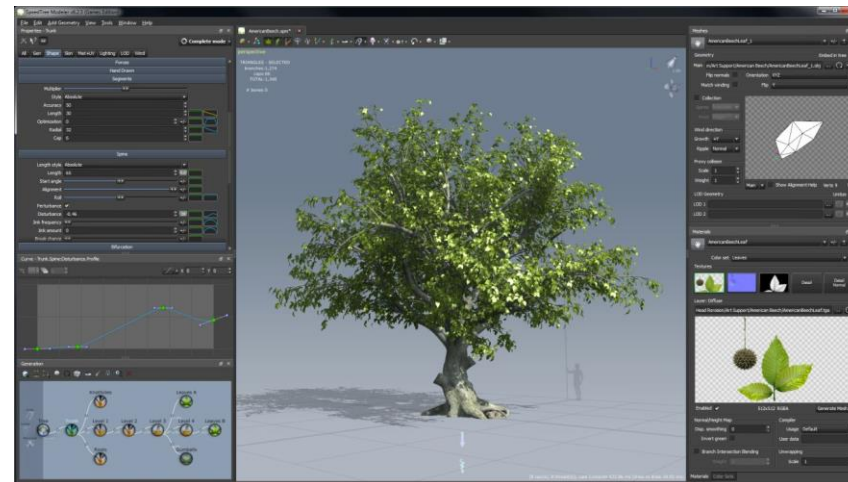


Borderlands series



No Man's Sky

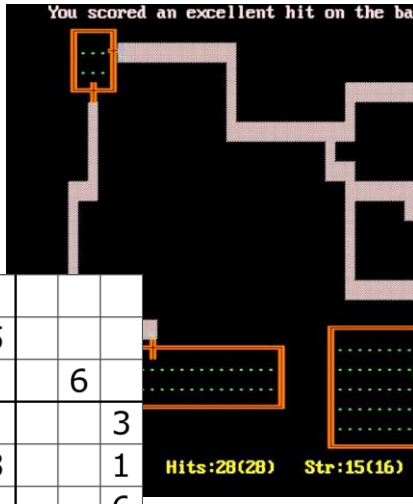
SpeedTree



Game Space (levels, worlds, puzzles)

Sudoku

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9



Rogue (1980)



Diablo 3



Bloodborne, Chalice Dungeons



Spelunky



Minecraft

Game Scenarios (story, quests, history)



Skyrim, radiant quests



Dwarf Fortress



Diablo 2, randomizing situations



Shadow of Mordor



MASS EFFECT™ ANDROMEDA

kotaku.com/the-story-behind-mass-effect-andromedas-troubled-five-1795886428

Why use PCG?

- Reducing cost of creating content?
 - Yes, when clever (SpeedTree), but can add complications (Mass Effect Andromeda)
 - Exponentially increasing costs (levels, maps, missions, etc.)
- CONTENT IS KING
 - If its interesting/personal/exciting to the player (Borderlands 2 guns), not if its bland oatmeal (No Man's Sky)
 - Computed content saves:
 - development costs
 - Storage and/or main memory
- Keep player playing/replaying
 - Adapting to player (Left4Dead AI director), new every time (Spelunky), or subarea (Bloodborne Chalice Dungeons)

Two Major Ways to use PCG

- Runtime PCG: customization, dynamic adjustment, replayability
 - Generate content live in game (No Man's Sky, Spelunky, Minecraft, Horizon Zero Dawn, etc)
- Offline/Design-time/Developer PCG: Speed up design of static content
 - generate, pick best, and save content
 - generate then edit
 - generate and incorporate (Speedtree)

PCG high-level Methods

- Rule systems
- Generative Grammars
- Search
- Constraint Solving

Comparing the Approaches

- **Rule Systems** allow for emergent output from simple rules. But hard to control output.
- **Generative grammars** can create high-quality output, but depend upon expert authoring of components and rules
- **Search** cuts back on authoring burden to just a heuristic and searchable representation, but these are unintuitive for many designers

How'd they do it?

- Minecraft: <https://www.youtube.com/watch?v=FE5S2NeD7uU>
- Galactic Arms Race: <https://www.indiedb.com/games/galactic-arms-race/videos/galactic-arms-race-promo>
 - <http://galacticarmsrace.blogspot.com/p/media.html>
- Spelunky: <https://www.youtube.com/watch?v=Uqk5Zf0tw3o>

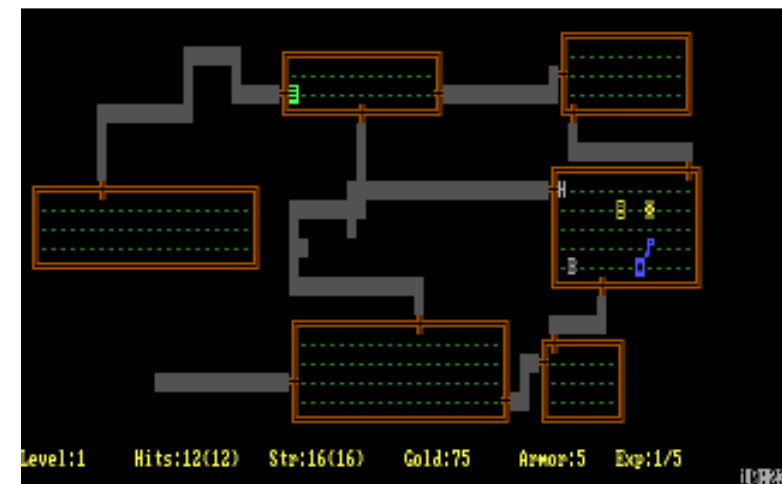
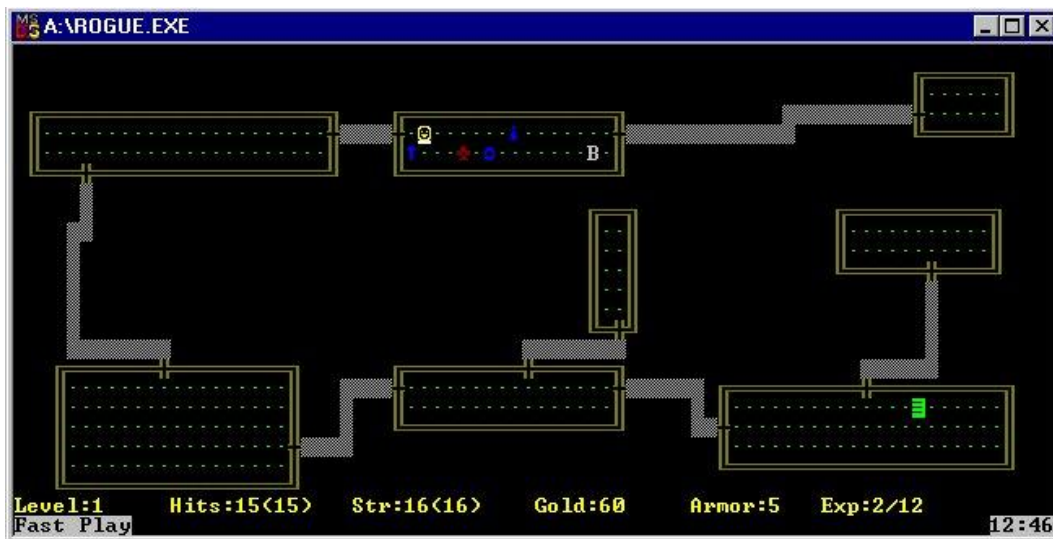
BRIEF PCG HISTORY

History of PCG

- Level generation was an early feature of games
- Games weren't professionally developed
- Memory and storage was limited
- Processors also limited
- Tradeoff between memory/storage and processor was important
 - Random generation was the answer

History

- Rogue (AI Design, 1983)
 - <http://www.abandonia.com/en/games/982/Rogue+-+The+Adventure+Game.html>
 - Granddaddy of all rpg dungeon-crawlers
 - Unlimited replay: dungeons + monsters PCG



History

- Elite (Acornsoft, 1984)
 - <http://www.iancgbell.clara.net/elite/>
 - Space trading (piracy, trade, military, etc.)
 - Original plan: 2^{48} galaxies (~282 trillion), 256 solar systems each
 - Final: 8 galaxies
 - Planet pos., commodity prices, name/local details
 - Model for Wing Commander, Grand Theft Auto, EVE online; influenced Second Life, World Of Warcraft...



History

- The Sentinel (Firebird, 1986)
 - http://www.youtube.com/watch?v=9V_pgo3vgil
 - “The first VR game”
 - 10,000 levels
 - Pseudorandom number generators (again)
 - landscape from the 8-digit number given at the completion of a previous landscape
 - Not all landscapes tested; possible to skip difficult levels by rebeating earlier one
- Consolidated list:
 - http://en.wikipedia.org/wiki/Procedural_generation#Software_examples

Sidebar: The random seed

- Pseudorandom number generator
 - Algorithm creates sequence of numbers that are “sufficiently close” to random
 - Sequence is completely determined by initial value(s);
- What happens if
 - Different seed every time the game is played
 - The same seed is reused

Problems, Solutions, Problems

- Early
 - Not enough space
 - Artists (and art assets) cost money + space
 - Shift costs to algorithm design + CPU
- Middle
 - Can “cheat”, ok to repeat objects
 - Artists create “seeds”, algorithms combine
- Now (spiraling costs of high-quality content)
 - “behind the scenes” → trees and vegetation
 - “search-based”, “experience-driven” → Player models
 - Flow (Dynamic Difficulty Adjustment, Drama Management)

Still not seen as efficiently + reliably producing content of sufficient quality for game content

Desiderata

- Fast, Reliable, High-quality
- Novelty, Structure, Interest
- Controllable (parameterized)
 - Geometric aspects (e.g. length of track)
 - Gameplay aspects (e.g. difficulty)
 - Adapted to player(s) (e.g. preference, type, fun)
 - Designer-centric as well

See: <http://www.marioai.org/levelGenerationCompetition.pdf> for excellent overview

This is not without risk.



The Story Behind Mass Effect: Andromeda's
Troubled Five-Year Development

<http://kotaku.com/the-story-behind-mass-effect-andromedas-troubled-five-1795886428>





Apps ▾

Packages ▾

Sales

Calculator

Forum

Stats ▾

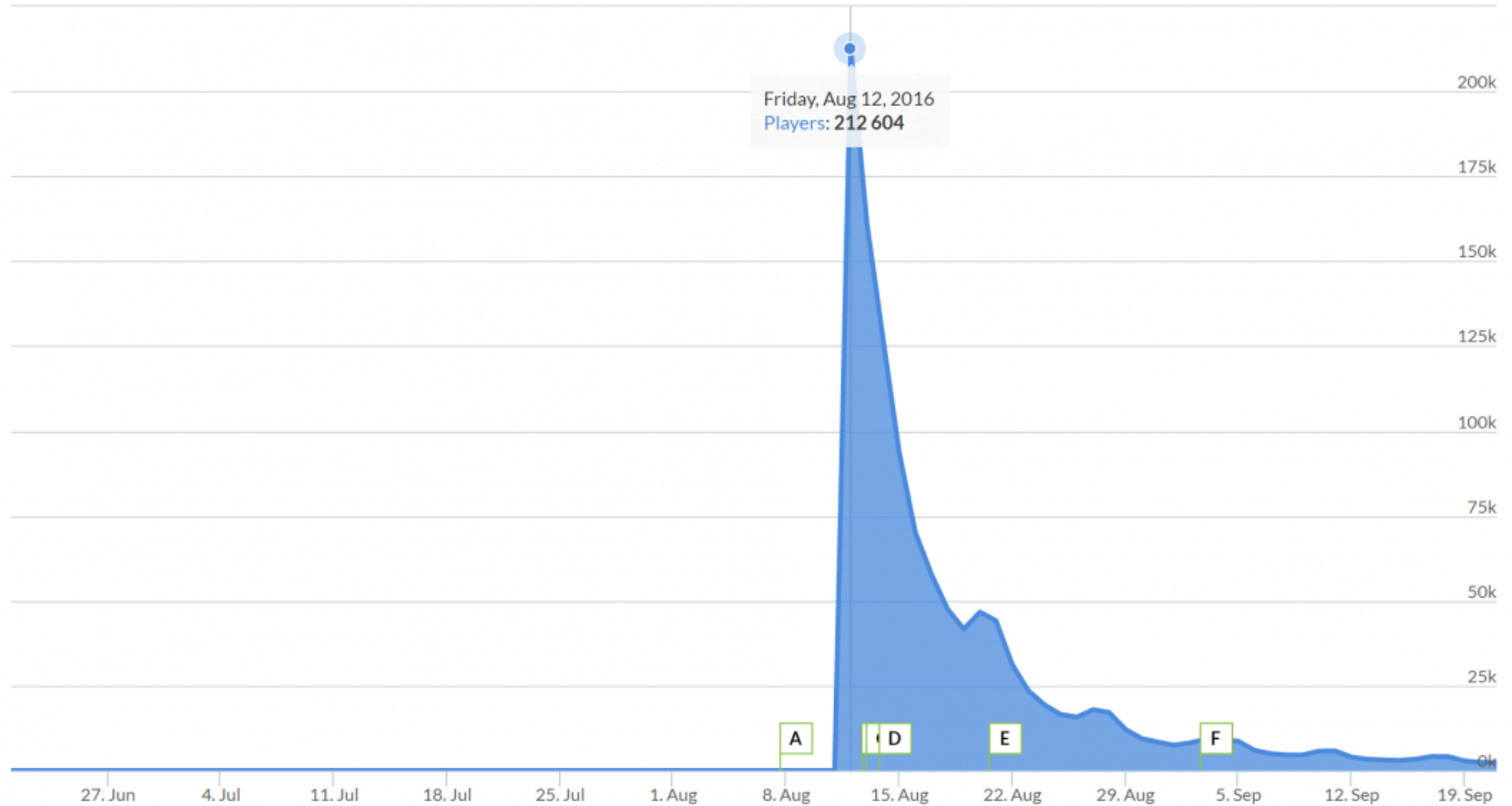
Other ▾

Search...



Sign In

Zoom 1w 1m 3m 6m 1y



<https://forum.facepunch.com/f/pd/bpxrt/Almost-Literally-No-Man-s-No-Man-s-Sky-s-player-count-is-dipping-below-1K-on-Steam/1/>

No Man's Sky

[Store](#) | [Hub](#)535
playing an hour ago657
24-hour peak212,321
all-time peakZoom 48h 7d 1m 3m 6m 1y **All**From Jul 31, 2016 To **Mar 27, 2018**[Compare with others...](#)

Month	Avg. Players	Gain	% Gain	Peak Players
Last 30 Days	568.8	-139.3	-19.67%	1,218
February 2018	708.2	-222.2	-23.88%	1,406
January 2018	930.3	+25.0	+2.76%	1,938
December 2017	905.3	-27.9	-2.99%	1,874
November 2017	933.2	-101.9	-9.84%	1,915
October 2017	1,035.1	-1,111.0	-51.77%	2,277
September 2017	2,146.1	-2,749.4	-56.16%	5,733
August 2017	4,895.4	+4,068.0	+491.61%	19,017
July 2017	827.5	+176.3	+27.07%	1,694

BUT THIS CAN WORK, AND HAS WORKED!

L4D



Credit: <http://dankline.wordpress.com/2011/06/29/the-ai-director-of-darkspore/>

Spore



Darkspore




<https://dankline.wordpress.com/2011/06/29/the-ai-director-of-darkspore/>



Last 2 Spikes	If Doing Well, Next is...	If Doing Poorly, Next is...
Hard, Medium	Hard	Medium
Hard, Skip	Hard	Medium
Medium, Hard	Medium	Skip
Medium, Medium	Hard	Skip
Medium, Skip	Hard	Medium
Skip, Hard	Medium	Skip
Skip, Medium	Hard	Medium

Source <http://dankline.wordpress.com/2011/06/29/the-ai-director-of-darkspore/>

Things we AI Directed:

1. Matchmaker
 2. Level Selection
 3. Spawn types
 4. Spawn placement
 5. Spawn amount
 6. Enemy health and damage
 7. Loot and DNA drops
 8. Health and Power drops
 9. Music
- Just a Utility
Targeting Function!
- 

Things we didn't AI Direct,
but we probably should
have:

1. Damage and Crit rolls
2. Quality of Loot Drops
3. Loot Pickup Rolls
4. Elite spawns
5. Elite affixes





Star Trek Online

Generates new races, new objects, star systems and planets for exploration

Players can save coordinates for a system, which becomes available for others





From Dust: Terrain generation

Controlled Procedural Terrain Generation Using Software Agents

<http://larc.unt.edu/ian/pubs/terrain.pdf>



Elder Scrolls II: Daggerfall (Bethesda, 1996)

<http://www.elderscrolls.com/daggerfall/>

Generated terrain

15,000 towns, villages, dungeons

750,000 NPCs

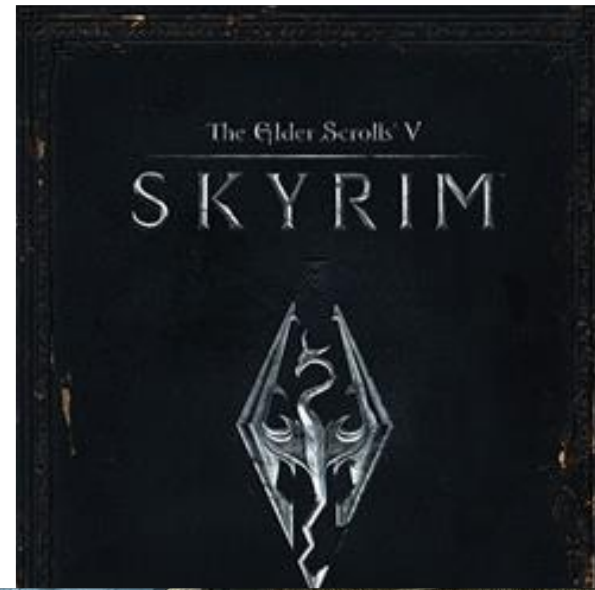


Fuel (2009)

Game world is size of Connecticut

The award for largest playable area in a console game was awarded (Guinness World Record)

Wander looking or people to race

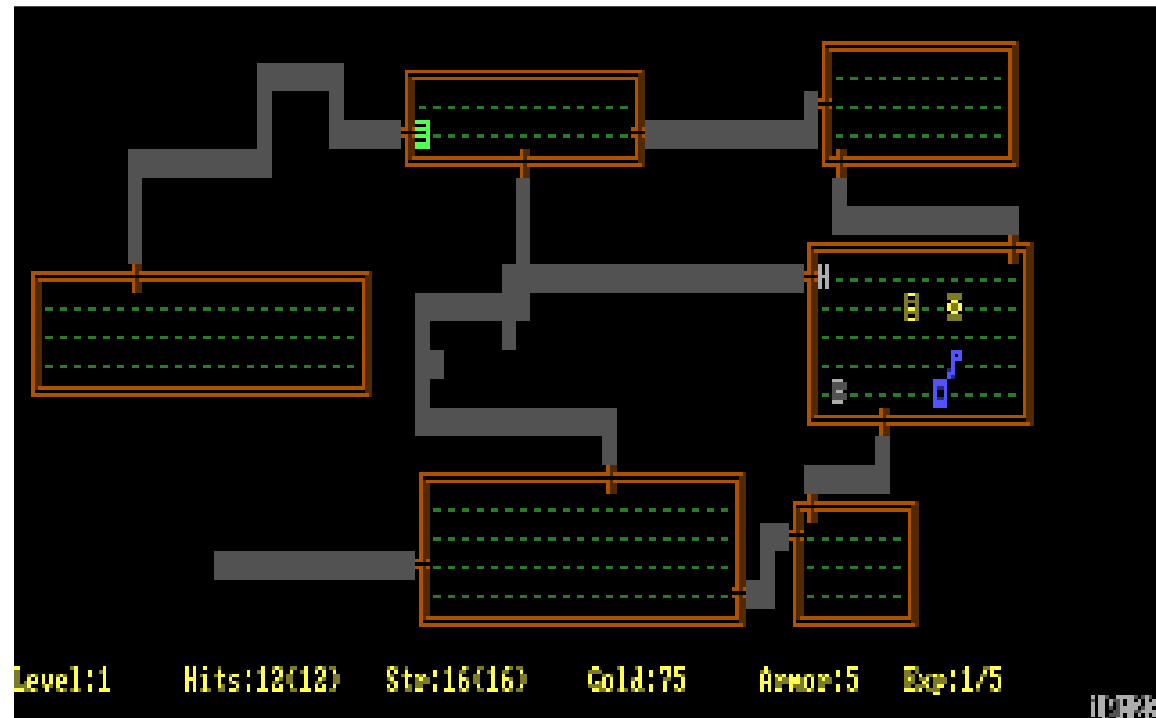


<http://www.terminally-incoherent.com/blog/2011/12/16/skyrim-radiant-quest-system/>

<http://www.joystiq.com/2011/11/09/howard-skyrim-offers-infinite-procedurally-generated-quests/>

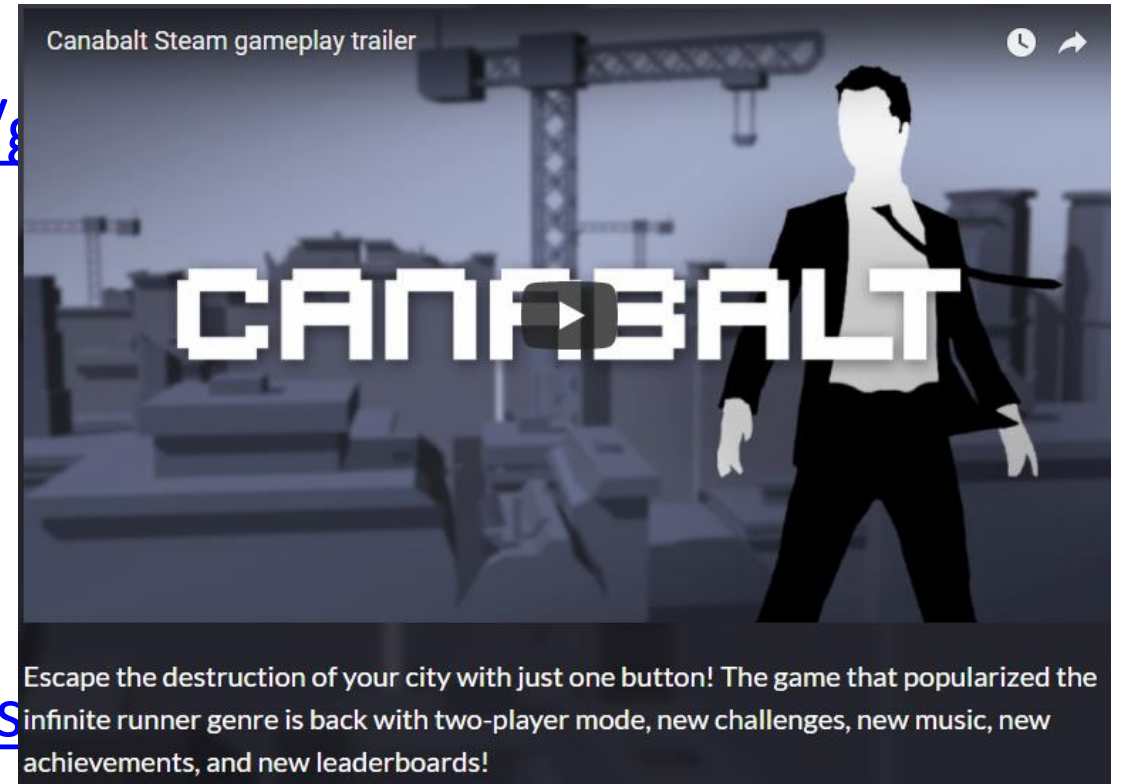
“Roguelike” Games

- <http://en.wikipedia.org/wiki/Roguelike>
- Rogue
 - <http://www.hexatron.com/rogue/index.html>
- Moria
 - <http://www-math.bgsu.edu/~grabine/moria.html>
- Angband
 - <http://rephial.org/play>
- Diablo



Silly Platformers

- Infinite Mario Bros
 - <https://openhtml5games.github.io/games-mirror/dist/mariohtml5/main.html>
- Canabalt
 - <http://canabalt.com/>
- Robot Unicorn Attack
 - <https://www.adultswim.com/games/web/robot-unicorn-attack>



Indy Level Generators

- Spelunky
 - <http://www.spelunkyworld.com/>
 - <https://youtu.be/7lBmiyGkQyg>
- Dwarf Fortress
 - <http://www.bay12games.com/dwarves/>



Academic labs

- Galactic Arms Race
 - <http://gar.eecs.ucf.edu/>
 - <http://galacticarmsrace.blogspot.com/p/research.html>
- Game Forge (here!), SMB level generation (Guzdial)
 - <http://gvu.gatech.edu/computer-watches-youtube-to-make-games>
- Challenge Tailoring + Contextualization (here!)
 - <https://research.cc.gatech.edu/inc/challenge-tailoring>



MOTIVATIONS AND MANIFESTATIONS

How'd they do it?

- Minecraft: <https://www.youtube.com/watch?v=FE5S2NeD7uU>
- Galactic Arms Race: <https://www.indiedb.com/games/galactic-arms-race/videos/galactic-arms-race-promo>
 - <http://galacticarmsrace.blogspot.com/p/media.html>
- Spelunky: <https://www.youtube.com/watch?v=Uqk5Zf0tw3o>

Why industry considers PCG

- Development cost savings
- Replayability (e.g. easing permadeath)
- Customization/adaptation
- Dynamic difficulty adjustment
 - Flow theory
 - C.f. [http://en.wikipedia.org/wiki/Flow \(psychology\)](http://en.wikipedia.org/wiki/Flow_(psychology))
 - <http://jenovachen.com/flowingames/designfig.htm>
 - [http://en.wikipedia.org/wiki/Flow \(video game\)](http://en.wikipedia.org/wiki/Flow_(video_game))

Design time PCG

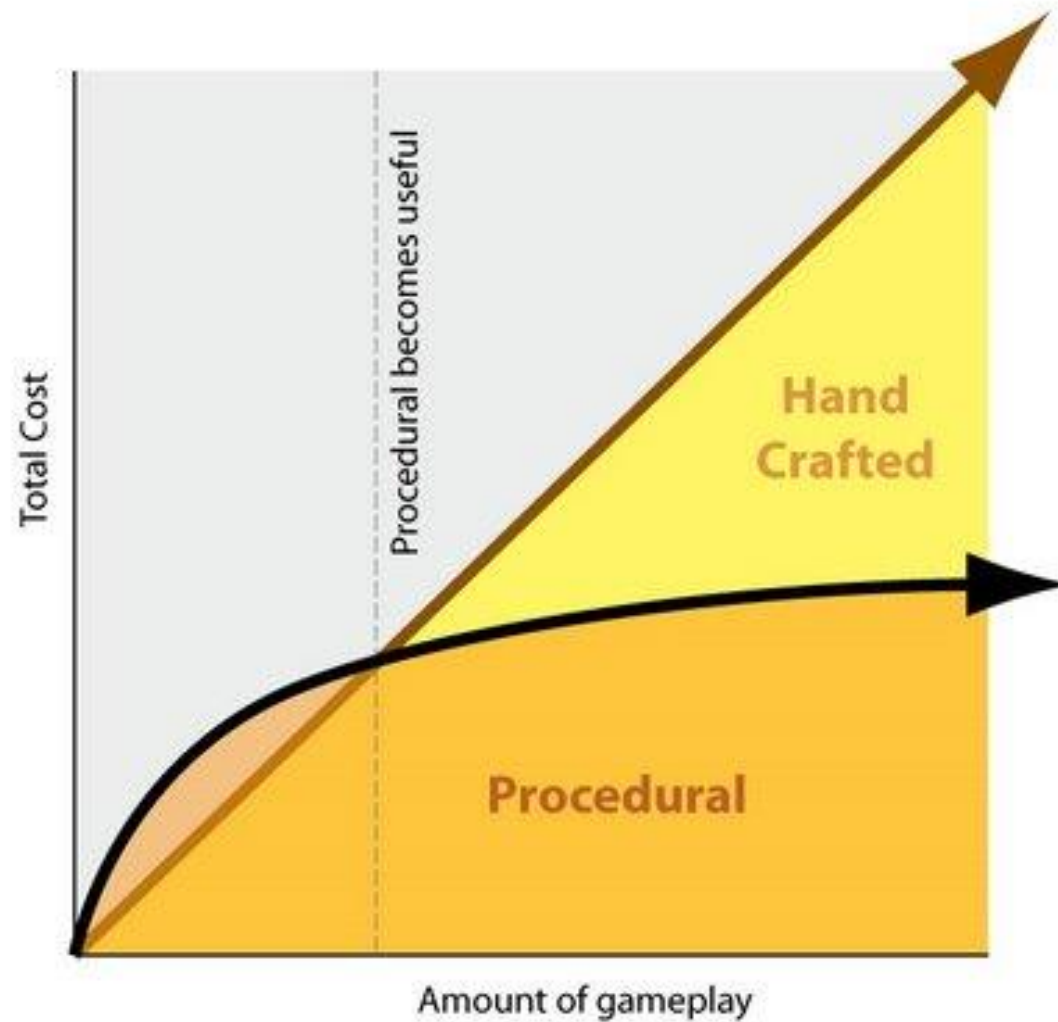
- Graphical asset construction is also consuming development time
 - # of unique objects in the world
 - Players expect non-repetitive
 - Game dev times now 100s of man-years with huge design teams
- Cost savings therefore a strong motivation
- But there are risks:
 - Procedures can do stupid things that destroy the player experience
 - Game design process exists to maximize control of design

Design time PCG

- Typically a mix of designer provided assets and procedural use of assets
 - Speedtree
 - CityEngine:
 - Star Trek Nemesis
 - Golden Compass
 - TV Commercials
 - Terragen



Economies of scale



Types of content

- Throw away
- Reusable
- Handcrafted anything is expensive and hard to modify
- The more handcrafted throwaway content, the less agile development, and more time spent on things the players don't appreciate
- PCG doesn't have to be human-level; just "good enough"

Run time PCG

- Players are different
- Preferences for pace and playstyle
 - Moderate challenge levels
 - Help avoid getting stuck
 - Adapt to player's tastes
 - Detecting player exploits
- When to use run-time PCG
 - When decisions can only be made at run-time
 - When you just can't pre-compute due to storage/memory limits
- Optimization problem
 - What is the set of content that delivers the optimal experience to the player given individual differences?
 - Example: rubber banding

Player Modeling

- What is optimal in PCG?
- Need a player model
 - Tells you something about the player
 - Makes predictions about player
- Challenges in player modeling:
 - What do you model?
 - Where do you get the model?
 - How do you use the model?

Promises

- Instancing in-game entities (especially appearance)
- NPC histories
- Puzzles that are harder or easier
- Puzzles/levels with more than one solution
- Story generation
- Pacing
- Different, multiple endings
- Meaningful choices
- Extending game lifetime

Pitfalls / Risks

- PCG can be NP-hard for anything non-trivial
 - But you can't always make players wait
- Thorough testing of run-time PCG is impossible
 - Best you can do is statistical sampling
- Offensive material
- Bad content
- Algorithm crash
- Meaningless activities
 - Easy to create quests, but if they don't connect to the larger game, no one cares

Hard problems in PCG

- Procedures structuring player's experience
 - Narrative
- Mental models of players
- Statistical player models
- Avoiding (dealing with) occasional catastrophic failure
- Algorithm safety
- Healing a broken level
- Social factors and full expressiveness with story/language generation
- “Deep understanding” vs. “generate & test”
- Personalized generated content
- Destiny
 - It's easy to have infinite levels, but dungeon crawling gets boring without a sense of progression/achievement
- **How do you know you are generating something interesting?**

WAYS AND MEANS

PCG Survey Paper

Check this out after class!

https://course.ccs.neu.edu/cs5150f13/readings/hendrikx_pcgg.pdf

Game Development is Changing

- Previous model
 - Designers did design on paper
 - Designers handed over to developers
 - Developers might iterate with developers on what is possible
- Current model
 - Developers build engines
 - Developers build tools for designers
 - Designers use tools to make characters, levels, cut-scenes, etc.
 - Developers help with things the tools can't do
 - Sometimes tools are developed concurrent with design (for new features)

PCG For *Real*

- Difficulty: encode design knowledge in algorithms to produce consistent, **quality** content
- Create types of game experiences impossible without PCG
 - Borderlands wouldn't work (as well) without gun cycle
 - Spelunky wouldn't work with static levels

PCG high-level Methods

(for game bits and space)

- Rule systems
- Generative Grammars
- Search
- Constraint Solving

From AI Game Programming Wisdom 2 (Rabin, 2004)

CASE STUDY: EMPIRE EARTH (SHOEMAKER'S MAPMAKER)

Stainless Steel Studios

- Dedicated exclusively to designing RTS games
- RT 3D game engine: TITAN 2.0
 - Windows, complete source (c++), 40+ person years
 - 3 million units sold ([got cash?](#))
- Empire Earth
 - Released Nov 2001
 - “... a new high-water mark for realtime strategy fans.”
GameSpy’s Dave Kosak
 - Same lead game designer (Rick Goodman) as AOE
- Empires: Dawn of the Modern World

Birds-Eye



Fancy 3D View



If You Were Asked...

- Make a system to generate maps
 - Tile-based
 - RTS
 - Multi-player
- What qualities should these maps have?



These Maps Must be _____?

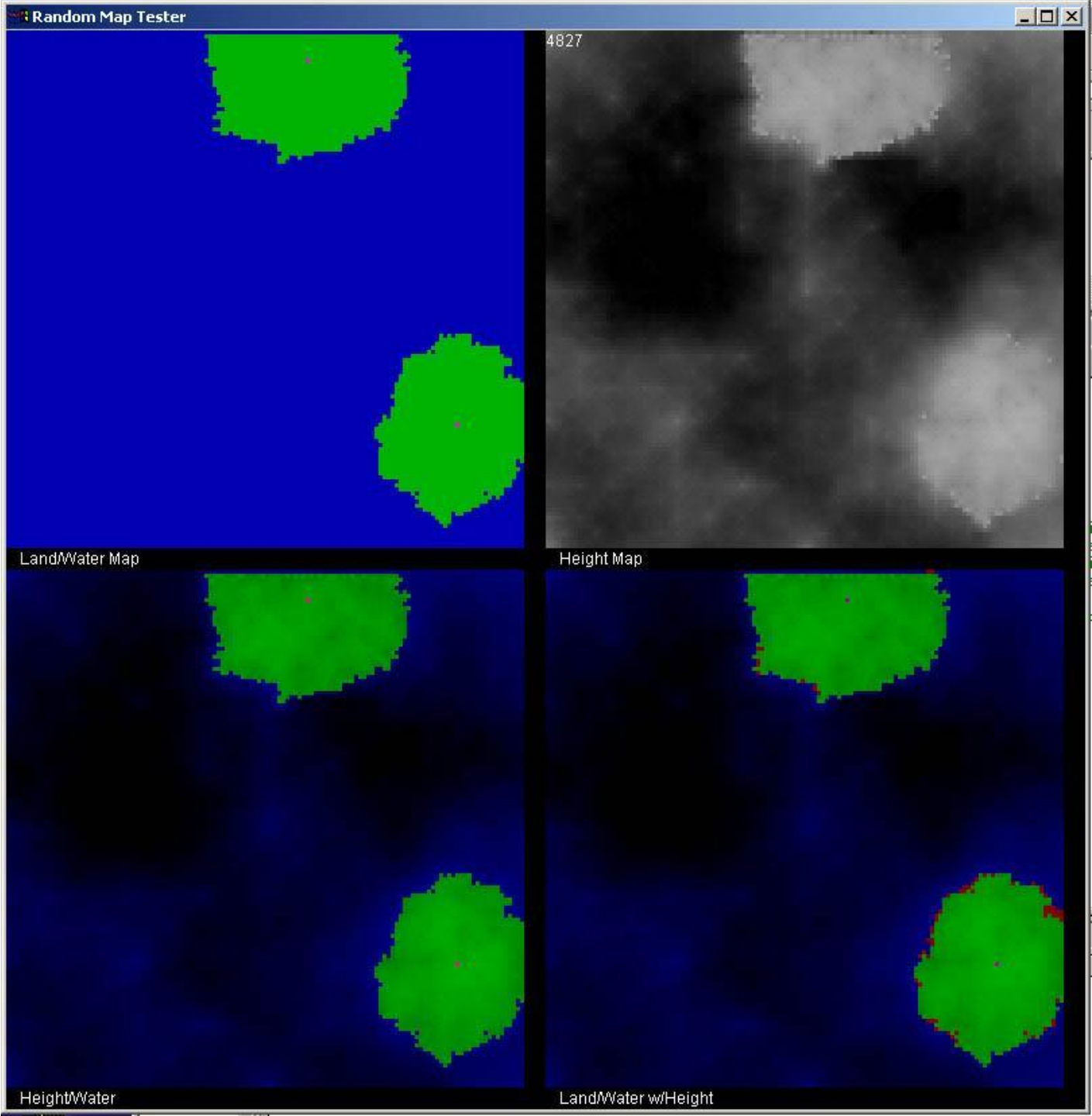
- Fair
 - Land per player
 - Resources per player
 - No strategic advantage
- Predictable
 - Similar by type
 - Allow for diff. outcomes
- Reproducible
 - Seed & type & size
 - Number of players
- Varied
 - Size
 - Type
- Relatively Quick
 - Many in-game systems not running
 - Impatient to start

Shoemaker's Mapmaker

- 1 .dll & 50+ script files
- Different .dll can be specified
- 7 scripts for each map type
- Inputs
 - # players, # teams
 - Size
 - Climate
 - Type
 - Random # seed

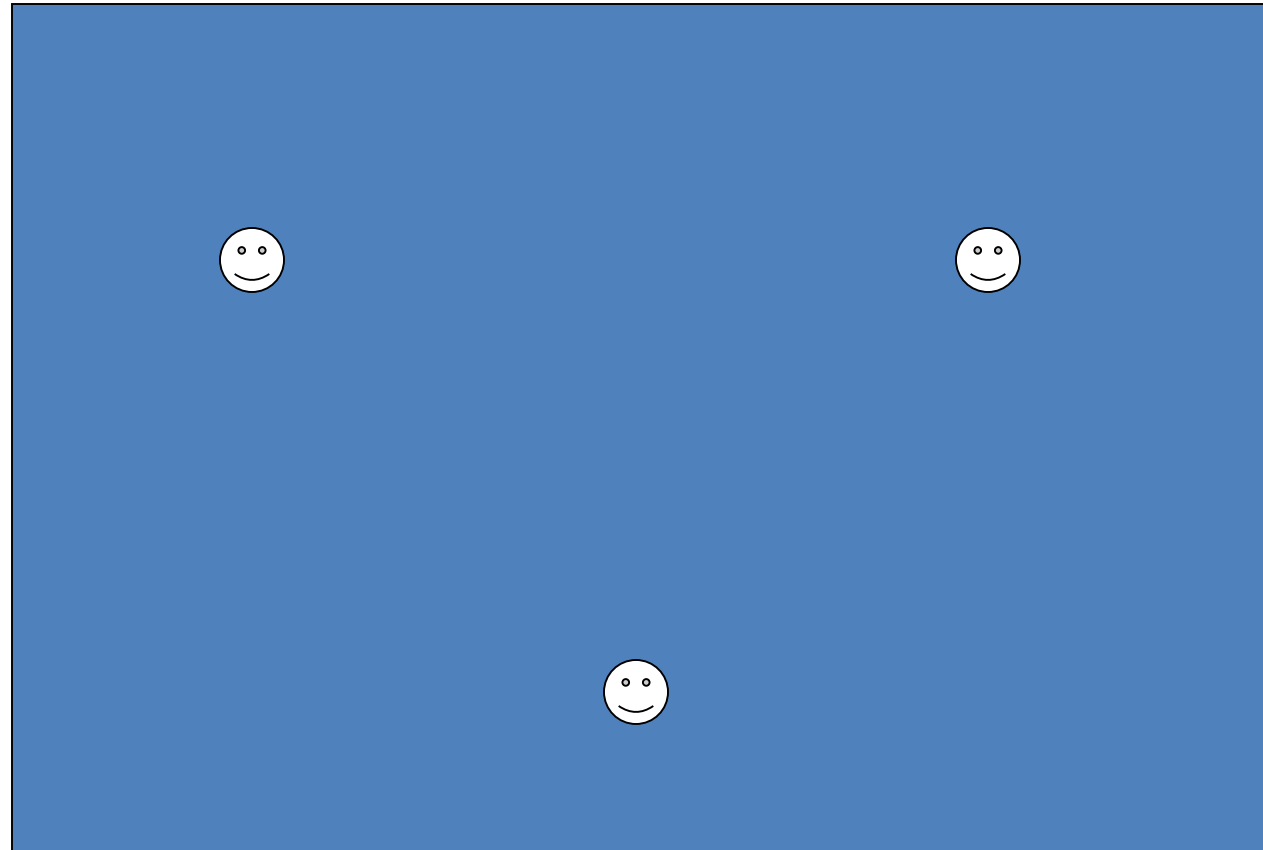
Intermediate Results

- Visualization tool created for 2D arrays
 - Land/Water
 - Height
 - Height/Water
 - Combined
- Map engine integrated late into game



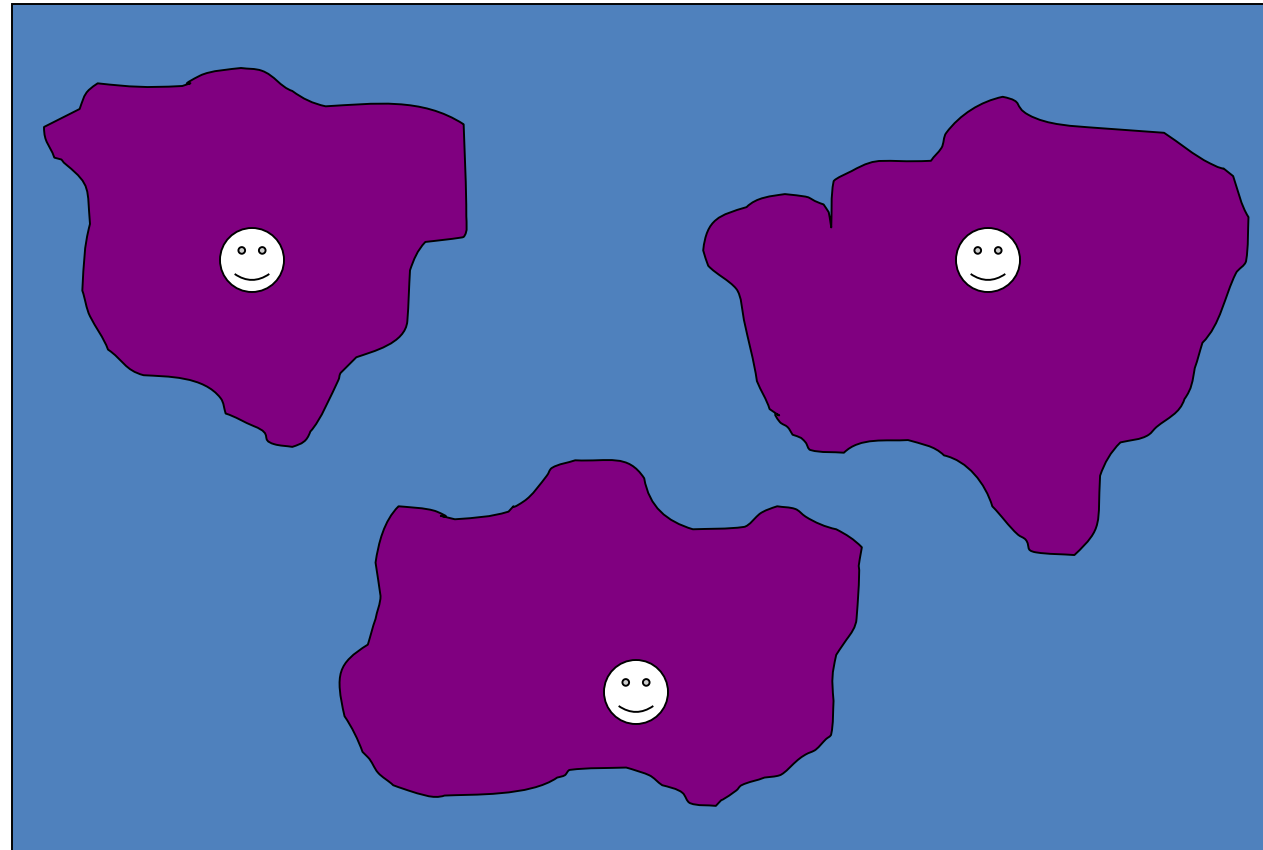
Solution Overview

- Step 1: Place players (blank map of H₂O)



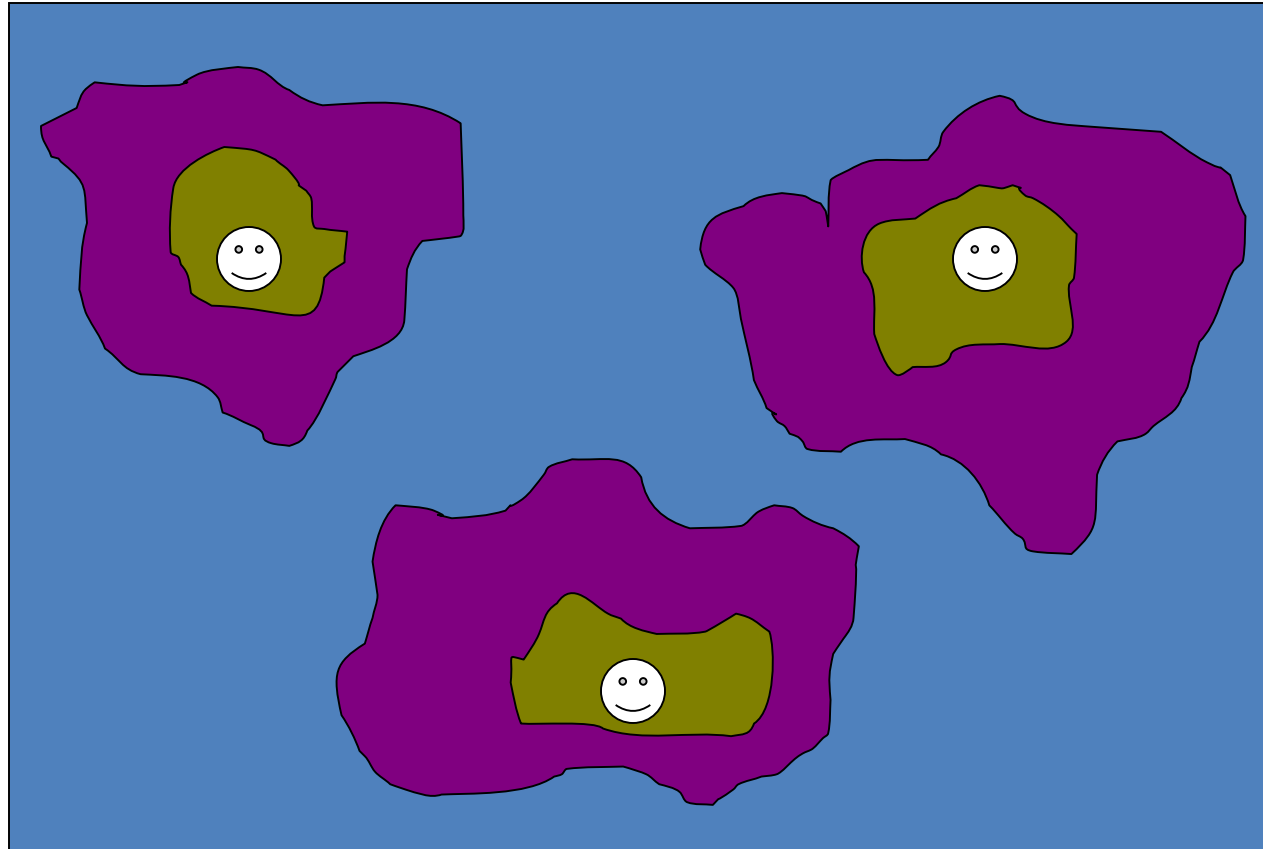
Solution Overview

- Step 2: Grow player land



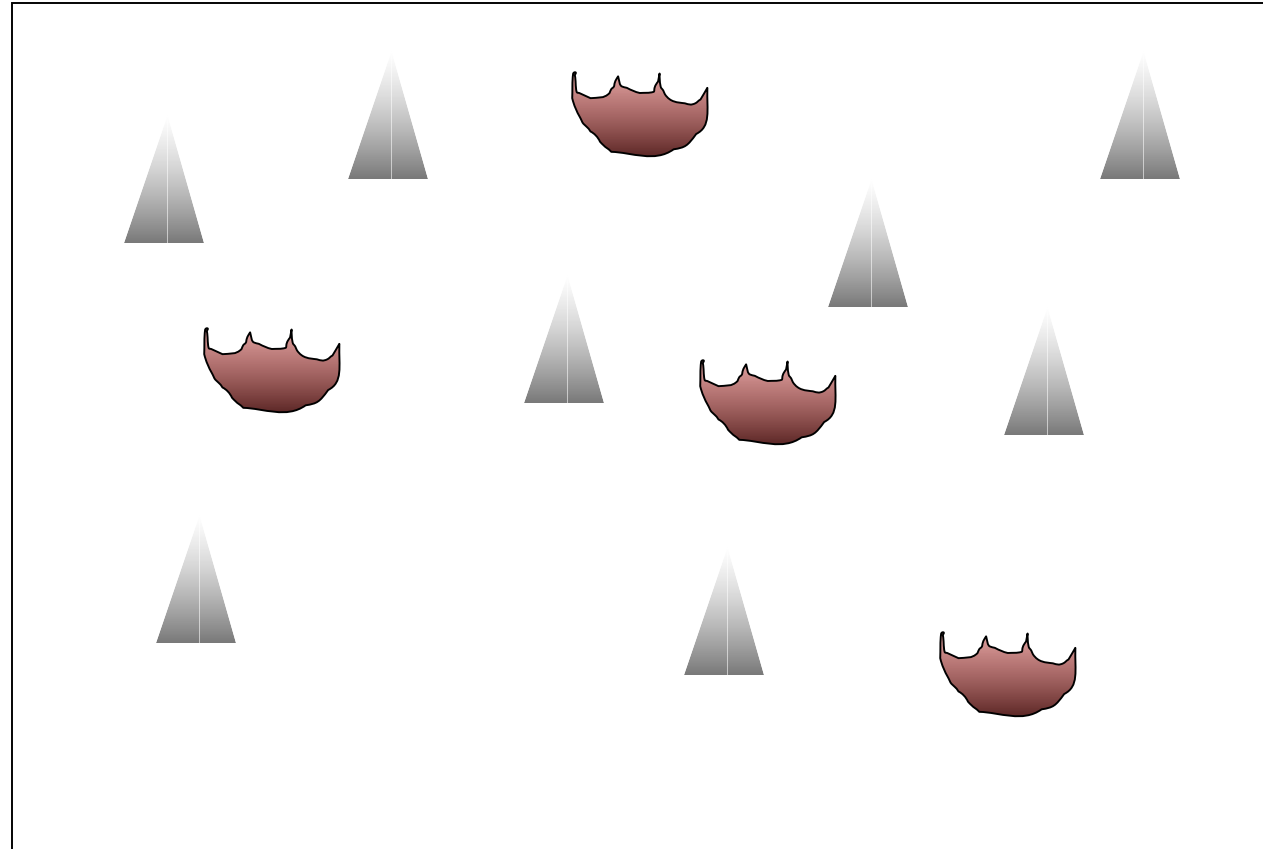
Solution Overview

- Step 3: Add flatlands to map



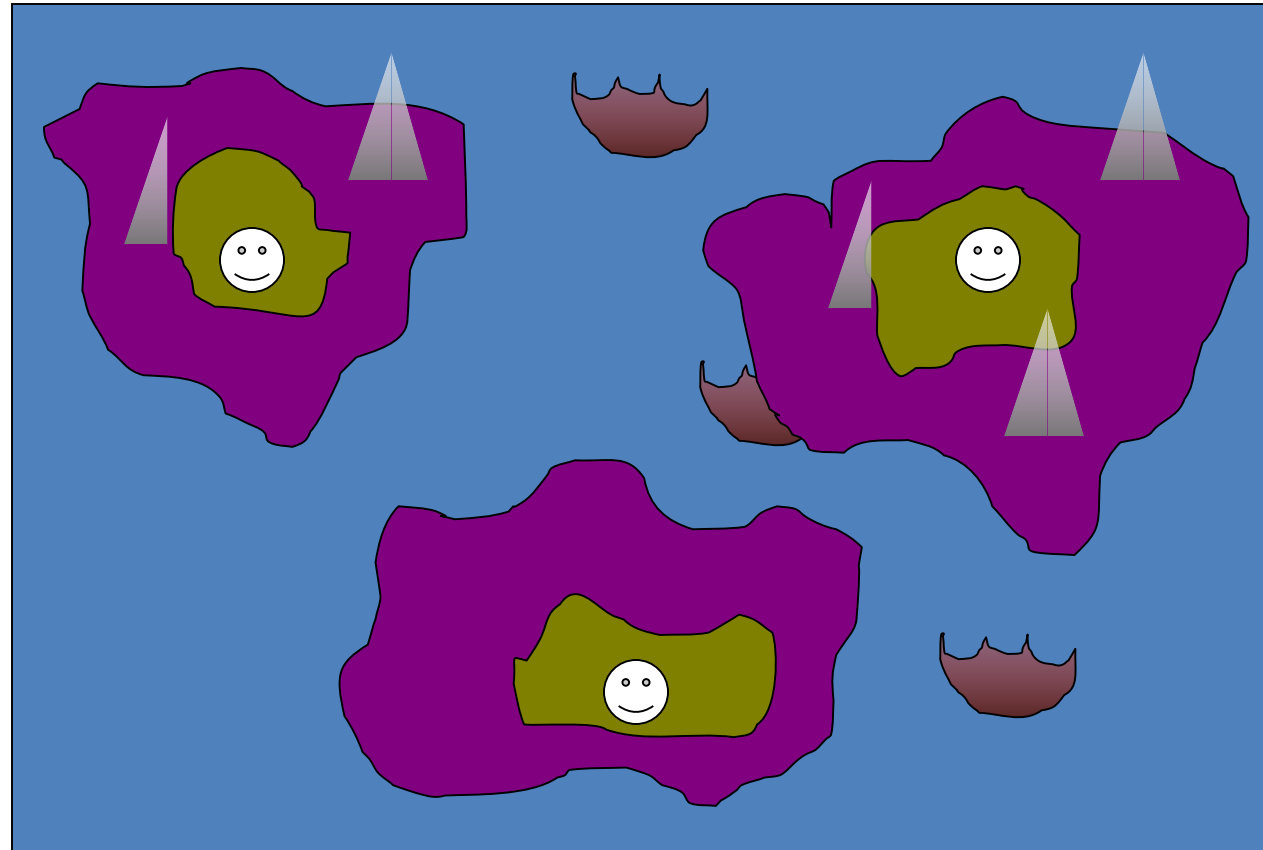
Solution Overview

- Step 4: Generate elevations map



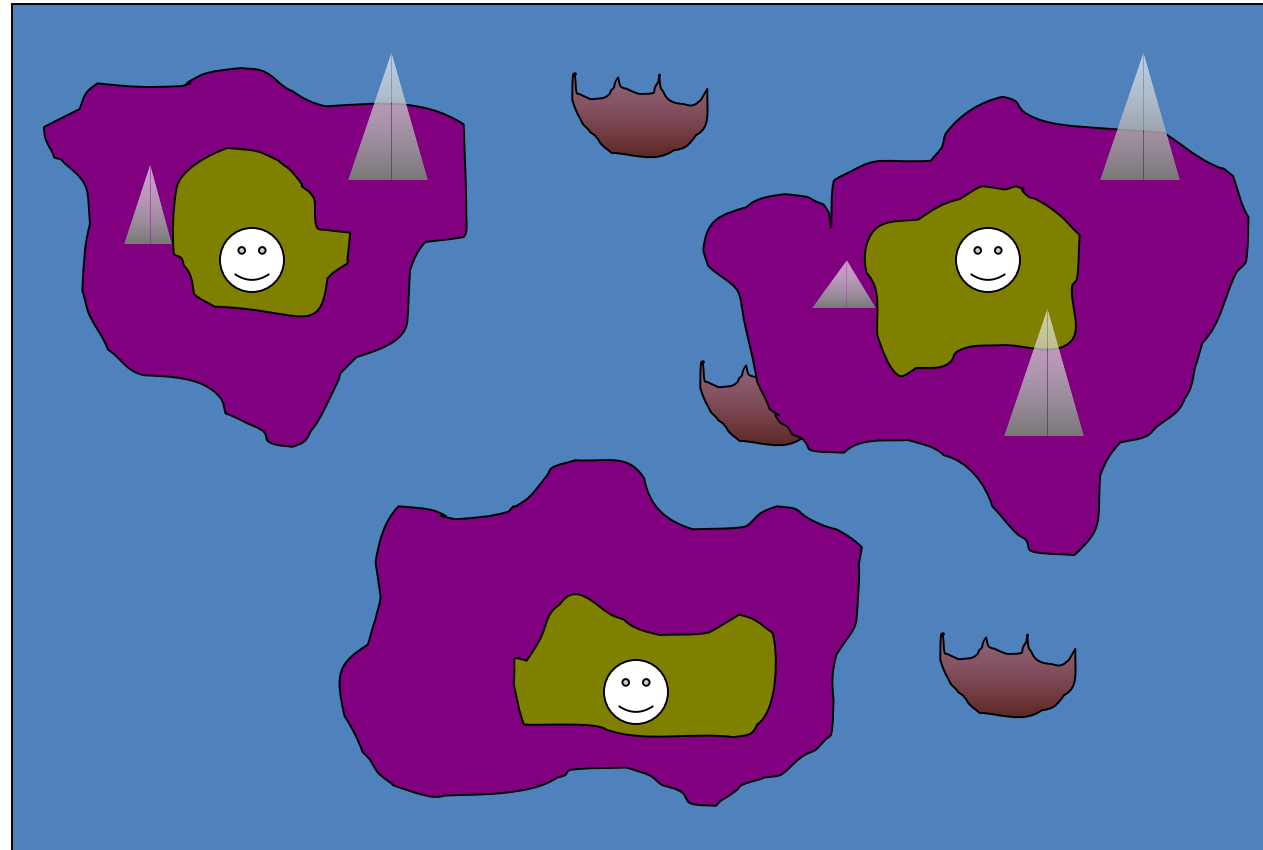
Solution Overview

- Step 5: Combine the two (obeying rules)



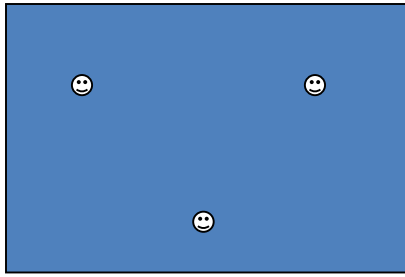
Solution Overview

- Step 6: Fix cliffs and other anomalies

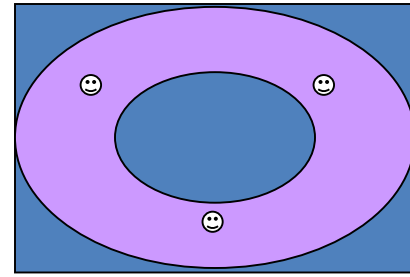


Solution Overview

- Step 7: Distribute resources to players
- Step 8: Place trees
- Step 9: Paint map with terrain textures
- Step 10: Place initial units for players
- Step 11: Place wildlife
- Steps 9 through 11 won't be covered.



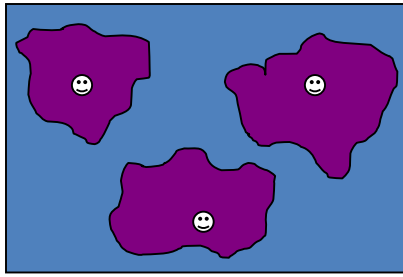
Step 1: Details



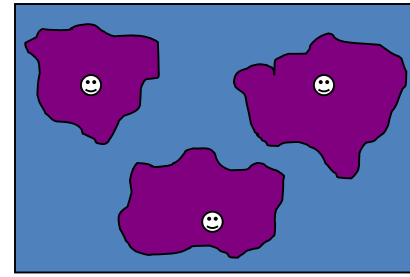
- Placement directly effects land growth
- How would you do this?
- Algorithm:
 - Inscribe one large disk onto map
 - Assign one random loc. per player w/in disk
 - Repeatedly push closest of player points apart until distribution sufficiently optimal
 - “optimal” is a map attribute
 - Specifies how evenly players must be placed

Step 1: Results

- Predictable player locations. Desirable?
 - Keeps maps balanced
 - Tunable through map attributes
- Assign player to points *after* placed. Why?
 - Teams adjacent to each other
 - Enemies somewhat symmetrically opposed
- What about colonization? “Dummy” players...
 - Placed after real players, on in/outside of disk. Why?
 - Dummies’ land grown after real players. Why?
 - Give the dummies resources too.



Step 2: Details



- Land is a resource in RTS games!
 - Allocate fairly
 - Must appear “natural”
- Clumps, modifiable script attributes
 - clumpSize
 - clumpNumber
 - Chaos level
- World starts as H₂O
- What’s a clump?

Step 2: Clumps

- Grow until
 - size == clumpsize
 - Create new clump some dist away at random, chaotic angle
 - Can't grow any larger
 - Make new clump of *limited* size
- Continue until number == clumpNumber
- Two growth methods, step & completion

Step Clump Growth

- 1st tile is players starting location
- Choose one *valid* random edge per iteration

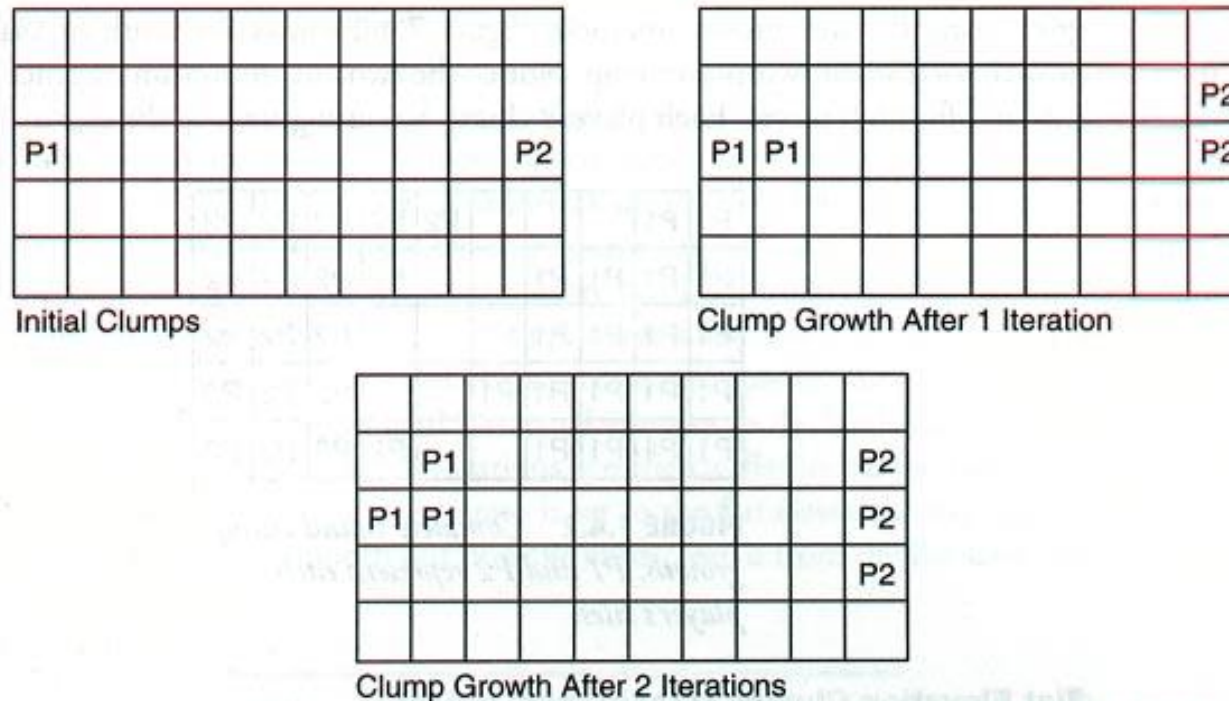


FIGURE 7.4.1 Clump growth using the step method. P1 and P2 represent each player's tiles.

Valid Clumps

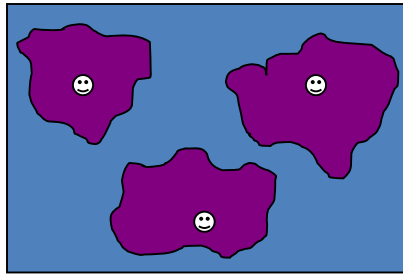
- Different types have different rules
 - Don't take owned land
 - Don't make peninsulas
- Island Clumps:

P1	P1				P2	P2	P2	P2	P2
P1	P1	P1	P1			P2	P2	P2	P2
P1	P1	P1	P1				P2	P2	P2
P1	P1	P1	P1	P1			P2	P2	P2
P1	P1	P1	P1			P2	P2	P2	P2

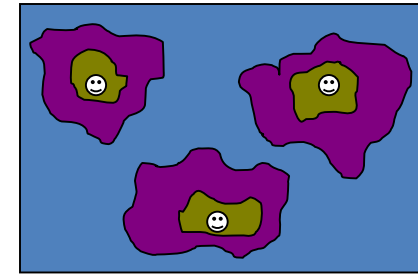
FIGURE 7.4.2 *Complete island clump growth. P1 and P2 represent each player's tiles.*

Step Clump Discussion

- Why is stepwise growth desirable?
 - Fair
 - each player has equal chance to grow
 - Preventative
 - Hinders one's land "surrounding" another's
 - Avoids "starvation"
 - Opportune
 - Makes good on the importance of placing players!
 - Remember the dummies!



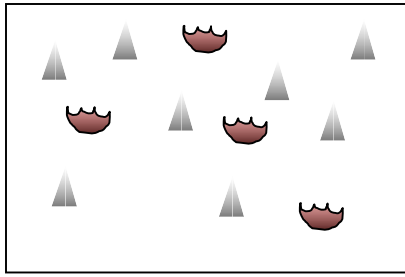
Completion Clumps



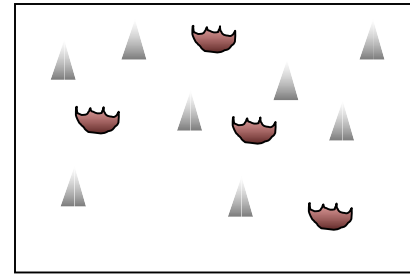
- Why bother, steps are awesome?!
- How about [step 3](#)?

P1	P1				P2	P2	P2	P2	P2
P1	P1	P1	P1			P2	P2	F	F
F	F	F	P1				P2	F	F
P1	F	F	P1	P1			P2	F	P2
P1	P1	P1	P1			P2	P2	P2	P2

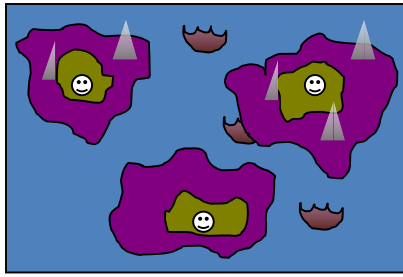
FIGURE 7.4.3 Complete flat clump growth. P1 and P2 represent each player's tiles. F represents a flat tile.



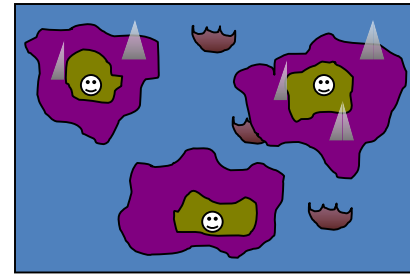
Step 4: Details



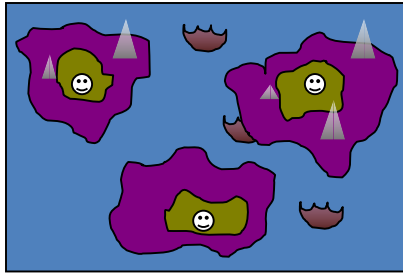
- 2D height map array
- Fractals (esp. diamond sq.)
 - Quick & easy
 - Realistic looking elevation
- Once again, map attributes
 - Min, max, initial elevation
- What about “noise”?
 - Filter used to smooth out jagged artifacts



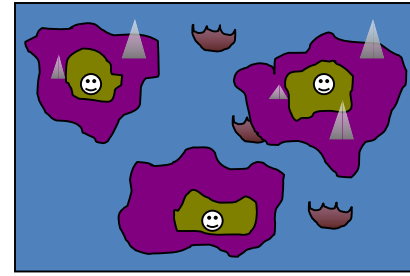
Step 5: Details



- Remember “[obeying rules](#)”?
- Algorithm:
 - if(landNH2O[i][j] == FLAT) ignore heightMap
 - if(landNH2O[i][j] == LAND && height[i][j] == WATER)
 - Raise value in height[i][j] to preserve natural look
 - if(landNH2O[i][j] == WATER && height[i][j] == LAND)
 - Lower height[i][j] to landNH2O[i][j] level
- Why not negate in the last two cases?



Step 6: Details



- We've got anomalies again!
- Wouldn't negating have worked?
- Re-apply filter

Step 7: Resource Allocation

- Each player gets same amt. of each w.r.t. other players
- Randomly placed within rings centered on players' starting locs.
- Validation
 - Some resources can't be placed on terrain features
 - Resources restricted in min. distance to other resources
 - Resources restricted in min. dist to start loc
- Once again, map attributes
 - Amount of each resource
 - Size of ring

Step 8: Place trees

- Plentiful; Restrict pathfinding
- Can the clump class be re-used?
- Map attributes
 - Number forests/player
 - Size of forests
 - Number of clumps/forest
- Some areas of map marked as forest free
- Player-owned land does not restrict growth

Scripts

- Language to manipulate map attribs
- Basic text parser and data structures
- Things like:
 - #resources per player
 - Max altitude in height map
 - Player land allocation
 - Climate (terrain/forest/animal texture set)
- Small changes yield map variations

Scripts: Pros

- Allow for incremental map type devel.
 - New attribs for new maps
 - Add attribs to parser
 - Matching code added to map generator
 - Offload some tasks from programmers to designers
 - Modability/extensibility/variety
- Next steps...
 - Island map w/teams: allow land for team-members to merge
 - Tweak elevations: island maps become canyon maps
 - Tweak clump size: island maps become river maps

Scripts: Cons

- Unfortunately:
 - One master script per type of random map
 - One script per size of type of map
 - In each script, subsections for each qty. of players (# berries? # gold?)
- You can imagine:
 - Scripts can become hard to manage
 - Manuals needed to explain types & attribs
 - Additional tools to edit attribs across files
 - Designers are not programmers
 - Programmers must create map “templates”
 - Designers simply “tweak”

Buyer Beware!

- Can't explicitly place particular terrain features
 - “super-tile” design approach
 - Fairer/more visually appealing
- Designers are not programmers
- Chokepoints are possible & unfair
- Resource placement can still yield advantages