Skulls of the Shogun AI Post-Mortem

Borut Pfeifer

Developer: Haunted Temple Studios
Publisher: Microsoft
Platforms: XBLA, Windows Phone, Windows 8
Release Date: ~~February 2012~~ *Fall 2012*

BACKGROUND


RADICAL
ENTERTAINMENT


SOE
SONY ONLINE
ENTERTAINMENT


EA


HAUNTED TEMPLE
STUDIOS


PLUSH
APOCALYPSE
PRODUCTIONS

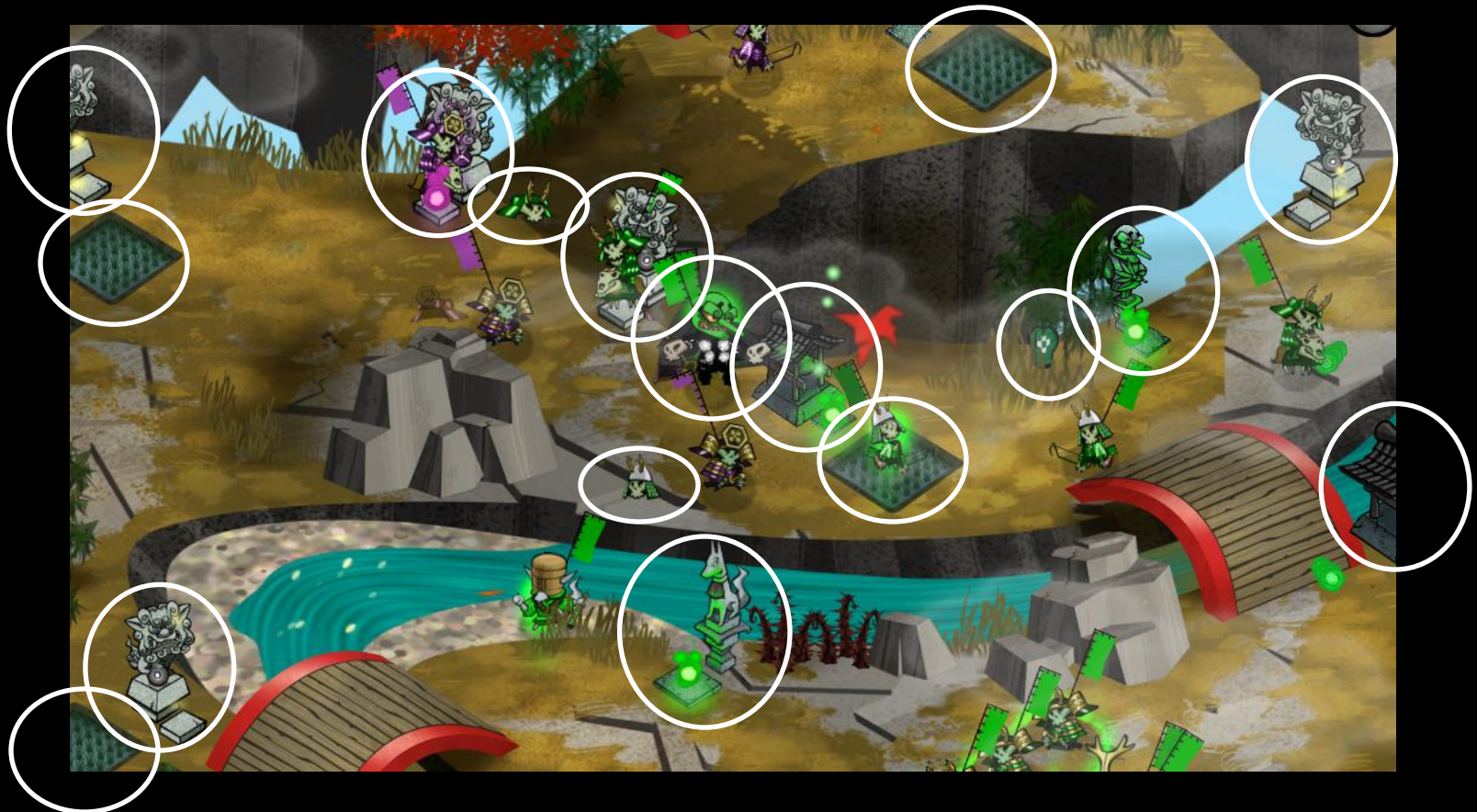
SKULLS OF THE SHOGUN

CONSTRAINTS

- Combination of RTS & TBS elements
- Turn based/board gamey
- No grid/analog world
- An action is a resource (5 orders per turn)
- Arcade-strategy: fast, no “waiting for AI”

GAME MECHANICS - UNITS

GAME MECHANICS - RESOURCES



Game Mechanics - Terrain

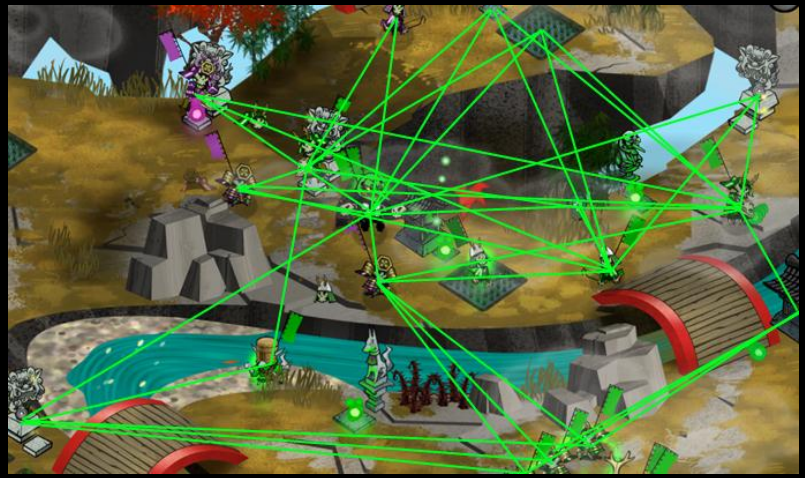
GAME MECHANICS - ADVANCED ABILITIES

1ST SKULL

Objective Evaluation

NxM problem:

- All units vs. all targets
- Infeasible, but ideal, right?
- Eval big list of unit-target pairs
- Utility theory based approach



Issues:

- Hard to compare strategic differences with so many pairs.
- Huge diff in utility between available/unavailable objectives.
- Deciding to defend required a lot more context
  (want to avoid duplicating this logic in each low level objective).

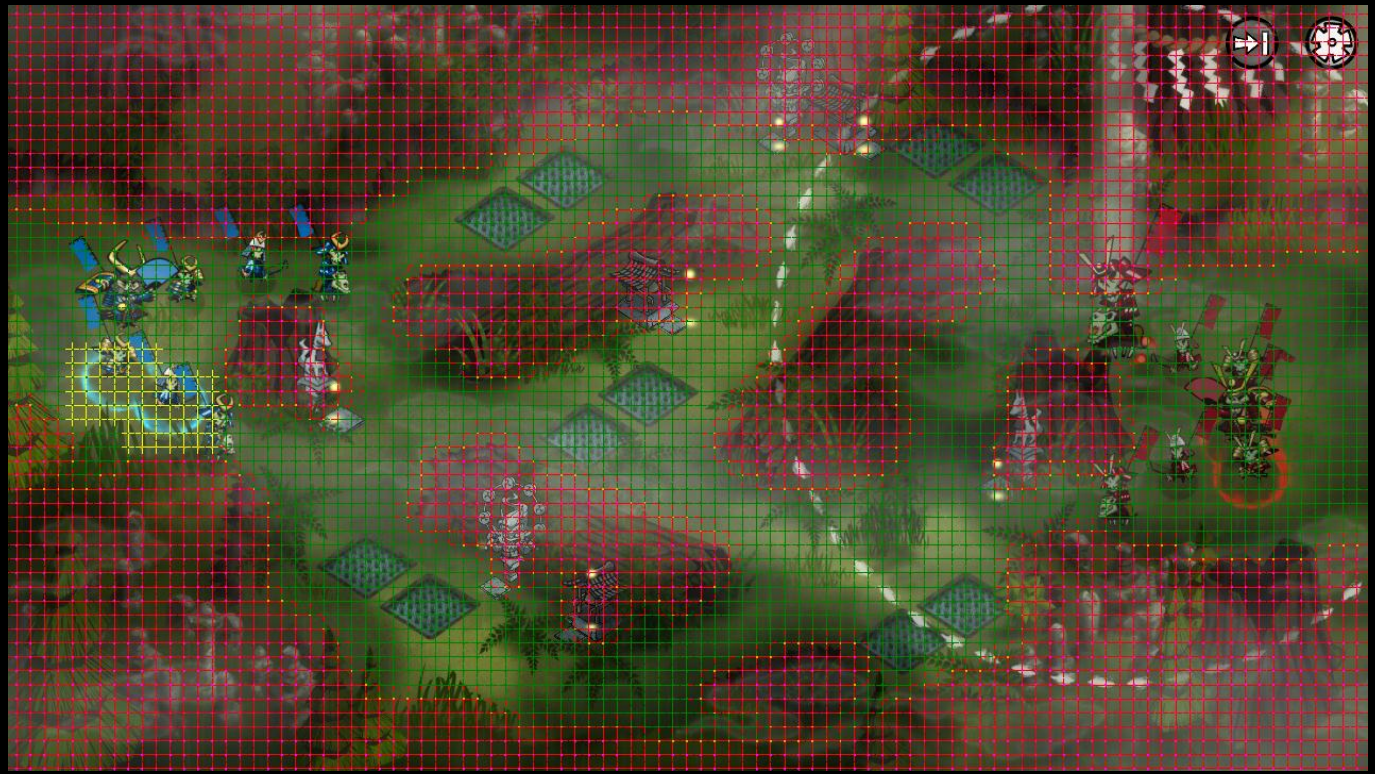
Pathfinding

Pathfinding in analog space

– Use grid even though inexact.

– Needs dynamic obstacle avoidance.

– Can’t avoid inaccuracies.



1ST PASS LESSONS

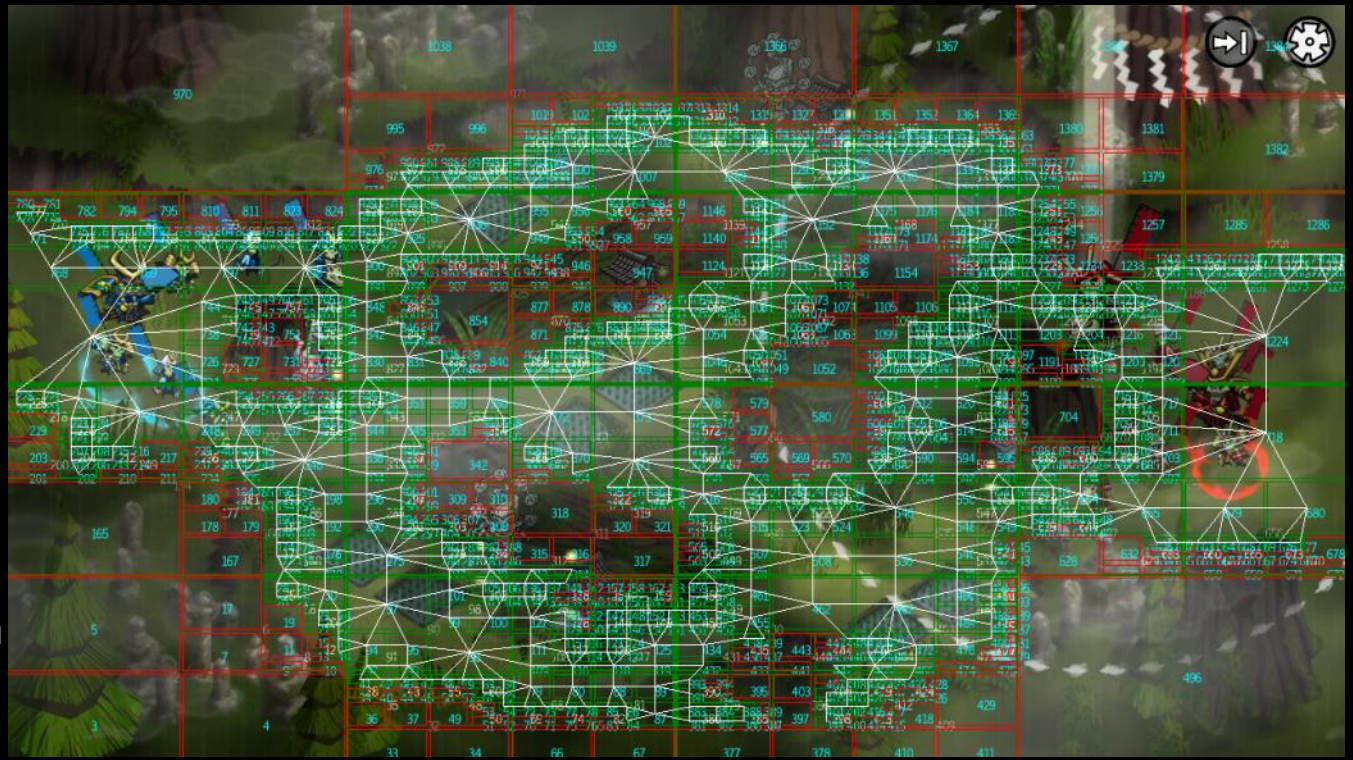
- Must organize objective comparisons.
- Architecture must be very *fault-tolerant.*
- Target distance key in comparing objectives.
- Must know all pair distances, ***fast***, at turn start - can't avoid NxM problem here.

2ND SKULL

Estimating Strategic Distances

Est. distances for strategic decisions:

- Use quad tree w/fewer nodes
- More inaccurate
- Use estimates when assigning objectives
- Still need full path when starting action



Decision Making

- Single list of objectives/targets
- Split by type:
  - Resources
  - Attack
  - Defense
- High level strategy modes:
  - Build up resources
  - Attack
  - Last ditch (losing badly)

Decision Making Algorithm

- Run pathfinding on quad tree for all pairs
- Rank objectives in each category
  - ex: prioritize enemies on periphery as easier to kill
- Find available units for every objective
  - in range this turn
- Run high level strategy check
  - Easy to manage, 1-2 pages of if statements
- Assign 5 orders to top objectives by strategy
  - resources strategy = 3 resource objectives, 2 attack, 1 defense
  - Fallback to other available objectives, then unavailable.

2nd Pass Lessons

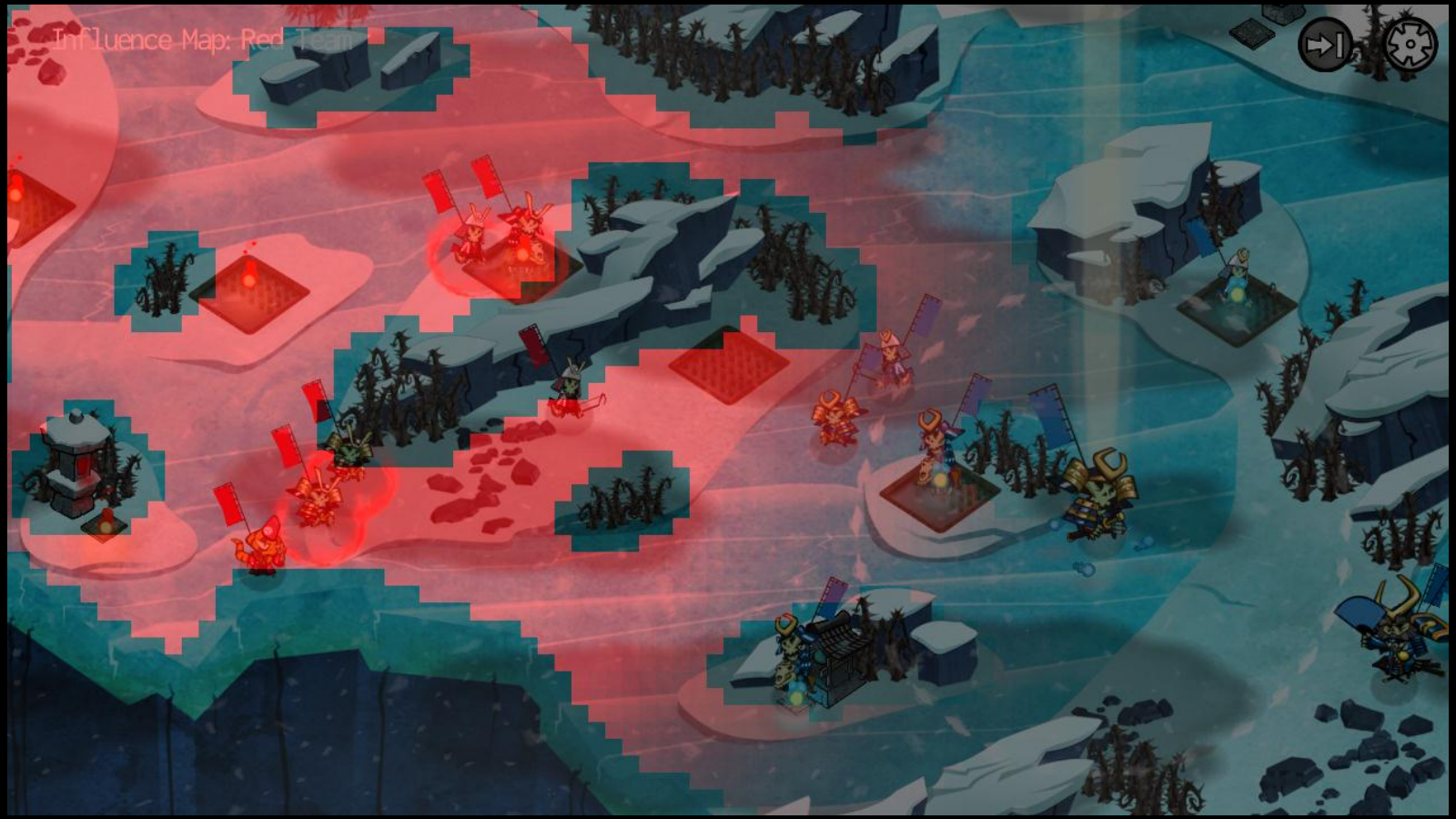
- Comparing objectives more manageable.
  - Lost some unimportant comparisons.
  - (When Unit X is better than Y for 2 objectives, but is way better fit for lower priority objective.)
- Quad tree introduces error:
  - Before objective starts, look for real path (dynamic obstacles can still cause failure)
  - Re-eval objectives on failure: looks “human”, because it tries one thing, finds it can’t succeed, tries another.

3RD SKULL

Still Missing Elements

- Need to prioritize unavailable objectives sometimes: ***Reinforcements!***
  - Simple set of tests in high level strategy analysis
  - < 3-4 avail objectives for > 2 turns, if extra units
- Need more tactical info for objectives comparisons: ***Influence maps!***
  - Besides avoiding counterattacks or prioritizing available targets, why is one target better than another?
- Difficulty modes, advanced spell use

Influence Map - Red Army



Influence Map - Blue Army

Influence Map – Blue Army Resource Appeal

Influence Map – Blue Army Threat Level

Influence Map Usage

- Easily plugged in to objective eval.
- Influence used for base objective score.
- Each objective multiplies that score.
- Maybe 3-7 cases per objective to modify score (easy to manage).

Oh Yeah...

Advanced AI:

- Ledges: Positioning to knock off, avoiding, forming walls
- Advanced spells: Simple, conservative scoring (costs rice).
- Difficulty: Categorize behaviors for novice vs. expert.

Things the AI still doesn't do:

- Use Spawn Oni spell (barbarian unit – too chaotic!)
- Not very zen like (doesn't follow master level strategies).

Learning is Fun!

- *A*lways *B*e *A*rchitecting
  - Never spend too much time up front - you'll be wrong.
  - Apply prototyping methodology - throw stuff out.
  - Leave spaces - code w/flexibility in mind, so you won't have to.
- *Apples to Apples*
  - Split up decision making to compare similar utilities.
  - Push common analysis (e.g. defense) to higher level to take advantage of broader strategies.
- *Fault Tolerance* at every level
  - Shit happens, your game model will never be accurate enough.
  - Always try to deal w/error 1 level above, possibly pass it higher up.

More Info

Skulls of the Shogun - Fall 2012

XBLA, Windows Phone, Windows 8

borut@skullsoftheshogun.com