Rampart — based off the 90s arcade game.

- one of my personal all time favorite games!!
- three phases
    1. Build
    2. Attack
    3. ReBuild

Game Design: Chains, Loops, and Player Motivation

Game Design Exercise about Chains: Structure, Motivation, and Depth

GDD:

Rampart — Game Design Document (GDD)

A modern re-creation of the 1990 Atari strategy / puzzle / shoot-'em-up hybrid

1. High-Level Overview

1.1 Game Summary

Rampart is a hybrid strategy/action game where players **defend territory by building walls**, **placing cannons**, and **engaging in timed combat.** Gameplay proceeds in repeating cycles of:

1. **Build/Repair Phase** – Patch breaches or expand territory using Tetris-like blocks.
2. **Deploy Phase** – Place cannons inside fully walled areas.
3. **Combat Phase** – Fire cannons at ships (single-player) or enemy walls (multiplayer).

The goal is to **maintain at least one fully enclosed castle** after each Build Phase; failure ends the game.
Rampart_(video_game)

---

2. Core Gameplay Loop

Loop Structure

1. **Initial Auto-Fortification**
   - Game generates a starter wall around a central castle.
2. **Player Cannon Placement Phase**
   - Player places cannons inside their enclosed wall. Space = cannon availability.
3. **Combat Phase (Timed)**
   - The player fires at moving ships (single-player) or opponents (multiplayer).
   - Ships/enemies fire back, damaging walls.
4. **Build/Repair Phase (Timed)**
   - Random Tetris-like wall pieces appear.
   - Player rotates/moves pieces to repair breaches or expand territory.
5. **Castle & Cannon Scoring Phase**
   - Any *fully enclosed* castle is retained and generates new cannons.
   - Unenclosed castles + cannons are *lost*.

6. **Repeat** until defeat or victory.

---

3. Game Modes

3.1 Single-Player

- Objective: complete **6 fixed levels**.
- Enemy: fleets of AI-controlled ships.
- Win condition: survive all rounds while maintaining at least one enclosed castle.
- Lose condition: fail to enclose at least one castle during Build Phase.

3.2 Multiplayer (1v1)

- Map divided by a river.
- Players bombard each other’s walls.
- Victory: higher score when both survive OR one player loses their final castle.

---

4. Detailed Systems

4.1 Player Territory & Castles

Castles

- Primary resource: defines territorial origin.
- Each fully enclosed castle yields **1 cannon** each cycle.
- Home Castle yields **2 cannons** if space allows.
- Multiple castles can be claimed by expanding wall territory.

Territory Rules

- Territory must be completely enclosed by contiguous wall segments.
- Breaches = loss of territory AND loss of cannons in that area.
- Craters and debris remain until covered by pieces.

4.2 Wall System

Tetris-like Pieces

Pieces are the core challenge. They:

- Vary in size and shape.
- Are often larger than the holes they must fill, creating tension.
- Cannot overlap structures, debris, ships, or land/water transitions.
- Must complete the enclosure before the timer ends.

Wall Rules

- A valid wall must form a continuous, unbroken loop around one or more castles.
- Any gap, even a single tile, invalidates the area.
- Overbuilding is allowed, but leftover protrusions can obstruct future repairs. Rampart_(video_game)

4.3 Cannons

Cannon Placement

- Must be placed **inside** enclosed territory.
- Must be on land (not water).
- Limited by available open tiles.
- Home Castle gives two cannons; other castles give one each.

Combat Properties

- Fixed firing arc straight outward (no diagonal turning in original).
- Fire rate: discrete timed intervals.
- Ammo: unlimited.
- Cannons can be destroyed by enemy fire.
- Power-ups (SNES/MS-DOS variants): some versions introduce enhanced cannon types.

4.4 Combat Phase Mechanics

Single-Player

- Enemy ships move along predetermined paths near your coast.
- Players must **lead their shots** due to slow projectile speeds.
- Objective: sink a target number of ships before timer ends.
- Ships fire back, causing craters.

Two-Player

- Each player fires at opponent walls and cannons.
- Objective: create breaches and destroy enemy cannons before Build Phase.

4.5 Build/Repair Phase Mechanics

This is the game’s strategic heart.

Key Features

- Timed phase (shorter in higher levels).
- Random Tetris-style pieces appear one at a time.
- Player positions and rotates pieces to fill breaches or expand loops.

- Must completely enclose at least one castle to survive.

Consequences of Failure

- No surrounded castle → game over (or eliminated in multiplayer).

4.6 Scoring System

Score Earned By

- Sinking ships.
- Capturing bonus squares (when applicable).
- Maintaining enclosed territory.
- Completing levels (single-player).

Score Lost By

- No specific penalties; score simply stagnates if territory/castles are lost.

---

5. Level Progression

5.1 Single-Player Campaign

6 levels total. Each level increases:

1. Enemy ship count
2. Ship fire speed
3. Combat damage spread
4. Build Phase difficulty (more complex breaches, piece RNG pressure)

Each level may introduce:

- New map shapes
- Additional castles to expand into

- Different shorelines affecting firing arcs

---

6. Visual & UI Design (for modern rebuild)

HUD Elements

- Timer (Build Phase / Combat Phase)
- Cannon count
- Castle count
- Enemy ships remaining (single-player)
- Territory highlight / valid enclosure visualization
- Piece preview (next Tetris piece) — optional modern QoL

Map Tiles

- Land
- Water
- Castles
- Walls
- Craters/damage
- Debris/ruins
- Bonus tiles (when applicable)

---

7. Controls & Input

Modern Implementation

- Gamepad, keyboard/mouse, or touchscreen.
- Must support:
  - Fast tile placement

    - Rotation buttons
    - Cannon aiming & firing
    - Quick cycling between Tetris pieces (optional modern QoL)
    - Multiplayer input mapping

Original used a trackball; modern input will be more flexible.
Rampart_(video_game)

---

8. Technical Notes for Rebuild

Recommended Tech Stack

(You can modify this based on your dev team.)

- **Engine:** Unity, Unreal, Godot, or custom WebGL engine.
- **Networking (Multiplayer):** Photon, Unity Netcode, or Nakama.
- **Pathfinding / Physics:** simple grid-based tile system.
- **Rendering:** 2D tilemap or minimalist 3D.
- **Data Structures:**
    - Grid arrays for tiles (16-bit per tile recommended)
    - List structures for ships & trajectories
    - Procedural wall generation algorithms
    - Real-time action loop for combat physics
    - Deterministic wall validation check (graph-based flood fill)

---

9. Modern Enhancements (Optional)

If you want a contemporary, improved version reminiscent of the original while enhancing play:

Quality-of-life

- Undo last piece placement (limited uses)
- Ghost-preview piece before dropping
- Next piece preview or even a “bag” system
- Customizable build timer (casual mode vs hardcore)

New Features

- Additional castle types
- Upgradable cannons (range, fire rate, explosive shot)
- Enemy ship variants (fast ships, tank ships, bomb galleons)
- Procedural maps
- Campaign storyline
- Online multiplayer
- Replay system and spectate mode

---

10. Win / Lose Conditions

Win

- Single-player: completing all 6 levels.
- Multiplayer: opponent fails to enclose any castle OR time expires and you have the higher score.

Lose

- Failing to enclose at least one castle during Build Phase.
- Losing all cannons and territory without the ability to rebuild.

---

11. Game Entities

11.1 Player Entities

- Castle
- Cannon
- Walls
- Build pieces
- Territory ownership marker

11.2 Enemy Entities (Single-Player)

- Standard ship
- Fast attack ship (higher damage)
- Large ship (more HP, powerful cannons)

11.3 Environmental Obstacles

- Craters
- Debris
- Rivers
- Shorelines
- Bonus tiles

---

12. AI Design (Ships)

Ships follow deterministic or semi-randomized behavior:

Behaviors

- Path along coast
- Periodically stop to fire
- Evade cannon fire with slight lateral adjustments
- Retreat when damaged (optional modern enhancement)

AI difficulty increases per level by:

- Firing faster
- Moving faster
- Targeting castle centers instead of random wall regions
- Using coordinated volleys

---

13. Multiplayer Design

Modes

- **Classic Duel:**
  - Same map
  - One wall set each
  - River divides land
- **Modern Modes (optional):**
  - 2v2 teams
  - Free-for-all islands
  - Draft mode (choose cannons & castle perks)

Sync Requirements

- All cannon shots must be deterministic
- Wall validation must be host-authoritative
- Ship variants disabled or mirrored for fairness

---

14. Development Milestones

Phase 1 — Core Systems

- Tilemap + wall placement
- Build phase logic

- Cannon placement + constraints
- Combat system with projectiles
- Territory validation system
- Single castle survival logic

Phase 2 — Game Loop

- Full round cycle
- Win/lose conditions
- All six levels for single-player

Phase 3 — UI + UX

- HUD
- Menus
- Visual clarity of walls/tiles

Phase 4 — Multiplayer

- Basic networking
- Sync of cannons, ships, walls
- Scoring + win conditions

Phase 5 — Polish & Enhancements

- Visual FX
- Sound/music
- Optional modern features

---

15. Deliverables for Dev Team

What this GDD gives them

- Complete mechanical breakdown

- Full game loop
- All game systems and entities
- Input + UI considerations
- Level rules
- AI logic
- Multiplayer architecture
- Tech stack recommendations
- Roadmap for implementation