# CS354R Final Project Technical Document

Shanth Koka

## Game Design

### High Concept – Voxel-Based Endless Tower Defense

### Game Pitch

You, the player, will have to collect and use scrap to defensively place blocks and turrets. Every night the robots come to steal the power in your base. How long will you survive?

### Core Technology

When it comes to the core technology, I have a lot of options that fit well with the game concept:

- A* Navigation: for enemy pathfinding
- Flocking AI: for enemy wave attacks
- Procedurally Generated Terrain: for a cool randomized world
- Physically Based Particle System: for juicy effects such as interacting with blocks, enemy attacks, turret attacks, etc.

These are just the few I've picked out from the project requirements page. My goal is to have at least one of these core technologies implemented within the final game. It is very likely that I will implement A* Navigation as it fits nicely with voxel worlds and it seems like the way to go for enemy pathfinding.

### Alternative Core Technology

I've been following VR ever since the first Oculus Quest. It is one of those technologies that I will never get tired of and I want to develop for it. My Quest 2 is sitting 3 hours away in Houston but I'm dead set on getting it and somehow adding a VR element to this game. In the rare case that none of the main core technologies are implemented, VR will be my substitute core technology. The VR element would allow the game to be played in a tabletop scenario, or from the view of the player.

### Supplemental Systems

- Generic Node/Component System
- Navigation System
- Inventory System
- XR Handler

### Game Engine – Godot 3.5.1

## Software Architecture and Plan

Generic Spatial Node Superclass with a Component System (1 hour)

- The main reason for this is to easily be able to add repetitive functionality to multiple nodes at once
- Every node that extends from this superclass would specify a list of components they opt-in to
- Possible components: Animated, PlayerControlled, Networked, StateMachine, GlobalIdentifier, Inventory, etc…

Game Manager (1 hour)

- Persists across scenes, handles global state of the game

Level Manager (3 hours)

- Handles state of a level/tower defense game
- Keeps track of time & spawns waves of enemies

Navigation Manager (3 hours)

- Computes A* pathfinding from voxel A to voxel B given movement constraints (height, flying, etc..)
- Should account for breaking blocks vs pathfinding around them

Keyboard/Mouse Player Controller (Component) (2 hours)

- Already built out, to be rewritten

Health (Component) (1 hour)

- Generic Health Component
- Emits death signal
- Allows others to change health

Inventory (Component) (3 hours)

- Stores list of items and quantities
- Player's inventory should display itself (this may be another component?)

Item (3 hours)

- Specifies the interaction action for specific items
- Ex) block right click = place, scrap right click = repair, etc..

Generic State Machine (Component) & State (1 hour)

- Already built out, to be rewritten

Enemy AI (4 hours)

- Relies on state machine
- Uses Navigation Manager for pathfinding to player base

Player Turret AI (4 hours)

- Relies on state machine
- Targets and shoots enemies

## Division of Labor

I am working on this project solo which means I am responsible for delivering all parts of this project. The bare minimum result would be being able to place down 1 type of block, 1 type of turret, and fight waves of 1 type of enemy with A* Pathfinding. I've added time estimates for each portion of the project mentioned above. In total it is looking to be 26 hours worth of work for the current plan. Of course, this doesn't account for the VR element and anything else I may add.