+1 (203) 560 7060

ashah19921@gmail.com

Vancouver, Canada
Connecticut, US

ANITA SHAH

GAMEPLAY/AI PROGRAMMER

LINKS

- Website
- Github
- Linkedin
- Itch.io

LANGUAGES

- C
- C++
- C#
- Rust
- Java
- Python

ENGINES

- Unreal
- Unity
- Bevy
- Bedrock

EDUCATION

University of British Columbia
2017-2020
- BSc in Computer Science
- BCS in Robotics/AI
  GPA: 4.20 (4.33)

Bowdoin College
2010-2014
- BA in Neuroscience
- BA in Anthropology

TECHNICAL EXPERIENCE/SKILLS

- Source Control Build Tools
- Git
- Perforce
- Confluence
- Multi-threading
- Object-Oriented Programming
- ECS Programming
- Camera Systems and Algorithms
- Combat, Control, Core gameplay systems
- Performance Optimization
- Testing, Debugging, Profiling systems and services
- Networking: Server Authoritative, Client Predictive
- Data Structures and Algorithms
- AI Architecture
- Behavior Trees
- Navigation Mesh Generation
- Hash Grids
- Influence Maps
- Tactical and Strategic NPC Behaviors
- RTS and Group movement
- Navigation and Pathfinding
- Physics and Collision
- 2D/3D Mathematics
- Linear and Matrix Algebra
- Differential Equations
- Console and PC Development

GAMES & PROJECTS

▶ AI in Self-Driving Cars

C++ & Unreal Engine

Implemented an AI system and algorithms for car entities in UE4 such that they learned how to navigate various tracks and terrains. For this design, a custom neural network with forward propagation was coupled with genetic evolution for backwards parameter tuning. The best results were generated utilizing 8 sensory raycast inputs with measured distance between the car and nearest solid surface along the vector ray. Custom backward propagation algorithms are currently being implemented to analyze the efficacy of genetic algorithms vs. stochastic gradient descent and back propagation.

▶ Aquamare!

Rust & Bevy Engine

Designed and Implemented a single player PC/mobile game in Rust on the Bevy Engine. The game utilizes jump and gravitational acceleration systems on the player entity in a similar manner to flappy bird's mechanics. In addition an NPC opponent summons enemy units that can project shield blocks. Similar to space invaders, the player can shoot projectiles to destroy the enemy units and shields. The game was developed in Rust to optimize parallel programming within ECS architecture.

WORK EXPERIENCE

Blackbird Interactive - Vancouver, CA.

AI & Gameplay Staff Programmer / 2019 - Present

A member of the core gameplay team working on Minecraft Legends, a AAA title in conjunction with Mojang and Microsoft since 2019. Helped develop unit/structure control mechanisms (direct, lure, advanced direct, build cursor), input handling for player controls, haptic systems such as rumble, presentation events for animations, gameplay camera systems for different game modes.

Main developer for custom navigation and collision systems in conjunction with short and long range pathfinding techniques for procedurally generated and dynamically changing landscapes. Developed asynchronous pathfinding tasks in conjunction with a pathfinding service capable of handling hundreds of A* pathfinding requests within a tick due to the RTS nature of the game.

Optimized long range pathfinding request handling ten-fold. Optimized ability to update paths to changing landscapes every 15 seconds versus the previous 90 seconds for more accurate NPC pathfinding. Optimizations were accomplished utilizing custom data structures, algorithms, and reduced memory allocation/deallocation.

Created custom influence map data structures utilizing KD-Trees, 3D Navigation Meshes (generated with delaunay tetrahedralization algorithms), and 2D hashgrids. Coupled with services and data parsing to allow for designer driven AI archetypes that could be dynamically assigned and removed from NPC characters in game via scripts. Custom group navigation methods were created to allow for unit formations and unit AI behaviors to maintain formation compositions.

Centre for Entertainment Arts

Course and Workshop Instructor / 2022 - Present

Non-tenured professor for the “C++ for Unreal” and “AI for Games” courses offered under CEA’s Advanced Game Development Program at Langara College and Kwantlen Polytechnic University. Major topics for C++ in Unreal include developing unique gameplay systems (for example re-creating BOTW climbing mechanics by overriding Unreal’s custom physics pipeline), custom controls and custom camera systems (created RTS-like camera systems, third-person camera systems, first-person camera systems, and custom blends and transitions between the different cameras), dynamic UI elements via navigational maps and custom tick components, audio and animation support, world-interaction and buff systems. Major topics for “AI for Games” include AI concepts, data structures, and algorithms in C++ such as custom pathfinding algorithms (A* algorithms and flow field pathfinding), custom influence map data structures and propagation for more dynamic NPC behavior (when coupled with UE4’s native EQS queries and behavior trees), custom neural network data structures in UE4 for autonomous and learned behavior.

Centre for Entertainment Arts

Industry Mentor / 2022 - 2022

Acted as an industry mentor for CEA’s Advanced Game Development program at Kwantlen Polytechnic University. Provided coding, architecture, and debugging advice for student final projects built in Unity/C# and Unreal/C++. Tutorials on networking with deterministic lockstep, Unity VR, Computer Vision for Unity, and Unreal 5 were provided. Conducted weekly mentoring sessions and playtests. Architecture and code optimization aid was provided upon request.

Sensory, Perception & Interaction Research Lab

Research Programmer / 2017 - 2020

Worked at the SPIN lab at the University of British Columbia developing novel silicon-based soft membraned robotic sensors. Constructed the carbon capacitive sensors and developed circuitry and arduino algorithms for 2D touch recognition, multi-touch recognition, hovering recognition, and pressure measurement. Optimized sensor sampling algorithms to take advantage of ATmega328P’s interruption protocol to increase sampling rate from 10 to 100 Hz. Increased speed of analog reads by 200% utilizing direct board manipulation rather than built-in libraries.

Built a pneumatically driven robot utilizing the developed silicon sensors and created body mappings using arduino gathered capacitance samplings and a C++ generated proprioception map. Utilized MATLAB, forward kinematics, inverse kinematics, and known concepts of pneumatic actuation to create custom predictive joint and end effector calculations for the generated robot proprioception maps.