Nikita Agafonov

📧 email, 🌐 nikita.works, 📂 linkedin, 🖥️ github

Summary

More than a decade of experience building complex, interactive web applications, including 2D and 3D graphics. I like to push the boundaries of what's possible with modern technologies and I like when things work fast! In spare time lift weights and do no-gi grappling.

Top Skills

UI and Front End: JavaScript, TypeScript, CSS, SASS, jQuery, Svelte, React, NextJS.
2D and 3D graphics on the web: WebGL, WebGPU, glsl, wgsl, Three.js, Pixi.js, canvas2D.
Desktop apps and networking: NodeJS, Electron, Websockets, WebRTC, WebAssembly.

Corporate Experience

Trillion AR, August 2024 - January 2025
Three.js Developer

- Developed realistically looking, physically based shader for real-time rendering of gemstones. Shader itself became an extension of physical material which enabled gemstones to be fully integrated part of three.js ecosystem. Utilization of the three-mesh-bvh library enabled real-time path-traced internal reflections for gemstones.
- Improved performance of the 3D viewer with automated generation of InstancedMesh, which increased rendering speed 3-6x for jewelries with hundreds of gems.
- Improved post processing step of the pipeline such that it works without invisible renders and effects passes, which boosted frame rate further.
- Improved resource disposal which allowed reliable embedding of 3D viewers for customers, especially for cases with multiple viewers on the page that switch dynamically.
- Developed a scene editor which allowed to customize jewelleries by designers and artists. The interface enabled selection of various jewelry elements and setting them up right in the browser.
- Developed an efficient video => tensorflow => shaders pipeline using native webGL and tfjs api. It allowed for faster execution by being able to skip unnecessary cpu⇔gpu communications.
- Developed demo pages for new customers to showcase the tech.

Immersive Studios, January 2024 - March 2024
Creative Developer (part time collaboration)

- Implemented GPU driven particles system animated by 3D Perlin noise. It features an interesting looking effect, making it an eye catching, mesmerizing experience. The result can be seen live.
- Utilized three.js based mesh sampling utility, which allowed the extraction of 3D points on the surface from the mesh for initial particles positioning.
- Optimized the performance of the application by tuning visual effects, shaders and implementing flexible resolution rendering techniques. This allowed it to achieve high frame rate performance on a wide variety of devices including mobile.
- Wrote an article on technical details which covered how presented effects were achieved. Article is published on medium and devto.
- Contributed to open source by sharing the code behind the particle system with all effects. Can be found in github repo.

Open Science Tools, September 2022 - July 2024
Software Developer

- Developed an electron based desktop application as a part of partnership with a 3rd party startup. The app acts as a command center allowing it to connect to multiple cameras and orchestrate the process of psychological assessments observing participants from different angles.

- Implemented a multi-window desktop app architecture which keeps UI and the state of all windows synchronized.
- Implemented a system which provides access to connected cameras, including out of the box support of Sony SRG-X400 PTZ camera and records their video streams progressively. Implemented full control of PTZ features from the UI which allows easy manipulation of the camera.
- Implemented custom WebGPU based rendering of the video feed with live timing rendered on top. Implemented custom text rendering solution with font texture atlas and on the fly text mesh generation.
- Implemented UI with Svelte, it features multiple tiles which can be dragged around, resized and arranged as desired. Implemented section resizing controls which allow it to adjust layout itself manually. Implemented custom in-app windows which behave similarly to native OS windows.
- Implemented websocket based connection layer to PsychoPy which enables real time communication with it and thus piloting assessments real-time.

PsychoJS contributions

- Developed brand new features and maintained current ones for psychojs - a web counterpart of PsychoPy tool based on pixi.js. It acts as a web renderer for projects / experiments built by researchers in the area of psychophysics and neuroscience and allows running those experiments online.
- Implemented multi-touch support for mobiles. Implemented GIF based animation support for the WebGL ecosystem. Implemented grating stimuli with glsl.
- Delivered various per-customer requests, including peer-to-peer remote control of the experiments, custom particle systems in Pixi, embedded youtube player and many more. Performed most of coding in vanilla JS.

Survey platform

- Developed a custom surveying platform which allows importing surveys from Qualtrics - a popular surveying solution in the field of psychology and many others. Solution is built on top of SurveyJS library thus providing all the same features plus some custom ones.
- Implemented .qsf format support - file format which describes Qualtrics surveys and allows them to be exported. Implemented a solution which translates Qualtrics surveys to our system by converting .qsf into a structure suitable for SurveyJS.
- Implemented custom UI components to enable broader support of Qualtrics UI elements.
- Implemented most of the code in vanilla JavaScript and plain CSS.

Freelance, December 2021 - September 2022
Three.js and web graphics developer

- Delivered a set of algorithmic and architectural optimizations for a startup with 3D apartment visualization and design product.
- Improved rendering performance by 50% through code refactoring and optimizing in-app search algorithms to O(N) complexity. Some of the internal components and classes were rewritten to reuse resources, which allowed faster scene boot.
- Implemented custom camera animations using quaternions instead of Euler angles, enhancing the overall user experience.
- Fixed lots of ridiculous bugs. The project's 3D portion was built in three.js while the UI was implemented in vue.

Real time sky and clouds rendering, January 2021 - September 2021

- Personal project which appeared from the work done at Vidina. See Personal projects section below for details.

Vidina, November 2020 - December 2020
Graphics programmer (short contract)

- Built a Jupiter-like atmosphere simulation shader, which was planned to be a part of VR experience.

- Implemented a set of shaders including 2D fluid dynamics with the help of gpu gems article and a volumetric renderer of the cloud-like substance.
- Shader is driven by the audio with data being extracted using web audio api. The result can be seen on vidina's youtube channel.

Sabbatical, October 2019 - November 2020
- Working on personal projects in graphics, experimenting with new tools, ecosystems and languages.

EPAM Systems, July 2017 - September 2019
Senior Software Engineer
3D Knowledge map
- Developed a 3D knowledge map visualizer and its administration system.
- Improved 3D model loading performance by 10x by optimizing search algorithms and utilizing buffer geometry data type of three.js (back then, they weren't default).
- Built an administration app from scratch using vanilla JavaScript and SASS for styling. UI features user management components, excel like data management and many more.
- There were two front-end engineers on the project with me leading the development.

CMS for TV studios
- Participated in the development of CMS for TV broadcasting studios.
- Developed new UI features, improved performance of the existing UI elements and fixed tons of bugs.
- Project was written in AngularJS 1.7 with elastic search on the backend.

E-commerce project
- Developed front-end for E-commerce project, cosmetics store.
- Implemented UI components for product purchasing. Optimized performance of the shopping items list loading.
- Project was written using Vue 2.1 and SCSS.

Orion Innovations (ex. MERA), March 2011 - March 2016
Software Engineer
Contact center for tech company in Calgary
- Onboarding of a new customer, a telecom software company from Calgary. I was one of the main developers to conduct initial customer onboarding, establish team relationships and bring the knowledge and tech back to our HQ.
- Went to a business trip to Calgary, where I've spent a couple of weeks getting familiar with the team and the product. Conducted knowledge transfer to our team back home.
- Conducted regular SWE duties, such as bugfix and small new features development to get familiarized with the codebase.

Contact center for local ISP
- Designed front-end architecture of the application from the ground up. The resulting system was essentially an in-house front-end framework which relied on jQuery, vanilla JS and SaSS.
- Implemented drag-n-drop based IVR (interactive voice response) tree editor which allowed the user to easily set up a voice response menu. The editor also featured zoom-in-out which made it convenient to design large trees.

Tool for bulk user update on remote system with no API
- Developed a multithreaded console based app written in C# using the .net framework.
- The app made it possible to update the remote system acting as a human user and generating appropriate html form submissions.

Mobile app for interactive presentations (slides)
- Designed the architecture of the app.
- Implemented custom smooth scrolling with inertia (at the time only iOS had that).
- Implemented smooth menus which accurately follow the thumb and respond respectively to the speed of gesture (snappy or gradual).

- Implemented resource efficient approach to present slides which enabled high frame-rate animations on presentations no matter the size, even with thousands of slides.
- Implemented websocket based chat, which also used efficient messages rendering and allowed to have thousands messages scrollable seamlessly.

Avaya sustain

- Worked as a part of Avaya service sustain for one of the largest IP-telephony systems at the time.
- Conducted call-trace capture analysis, mostly SIP based, to identify potential sources of the issues.
- Bug Fixes in two gigantic codebases used by the system - a mixture of C and SL-1, proprietary language. Telephony setup analysis, task triage.

Personal Projects

Grok Telegram bot, December 2024

- Utilized xAI grok API and telegram API to implement chat bot which is now a member in our friend's group chat.
- Developed an additional tooling enabling grok to search the internet, to be able to set reminders, ask users questions for more info. By default grok API only provides access to LLM and a Vision model without internet search capabilities.
- Custom system prompt is set for grok to have a character of T-800 from Terminator 2 so his replies are similar to those from the movie. There's also a simple function to evaluate current message history and save the essence to the system prompt which allows it to gradually evolve and remember instructions we told him.

3D cellular automatons, October 2023 - September 2024

- WebGPU based 3D cellular automatons computed and rendered real time. No dependencies, everything is built from the ground up.
- Developed path-traced renderer which allows to render high amounts of cells while maintaining interactive framerate. It works much faster than the default instancing approach if using triangles. Compute shader to calculate next grid state.
- Grid state itself is tightly packed in uint32 which allows larger grids to be sent to the GPU for compute.
- Maximum size of the grid is only limited by the GPU memory users have and its compute capabilities. Code is available in the repo.

Real time sky and clouds rendering, January 2021 - September 2021

- Developed a physically accurate sky and clouds simulation based on rendering techniques used by AAA game studios like Epic Games and Rockstar Games.
- Reverse-engineered D3D11 code samples and ported the techniques to WebGL, enabling real-time, high-fidelity atmospheric rendering in the browser.
- The project utilizes the same or very similar techniques to the ones used in Unreal Engine 4 (the latest tech in the 2021).
- The project is built with vanilla JS and all the shaders implemented with glsl. Few videos of it are available on my youtube channel.

2D Space Game, March 2016 - June 2017

- 2D space exploration game with in-house game engine built from the ground up with Pixi.js as a renderer.
- Implemented 2D collision detection and resolution algorithms which support processing of any convex hulls.
- Developed AI navigation system which uses obstacle avoidance algorithms instead of path-finding, which allows for more resource efficient NPC navigation. Implemented decision trees, which allow NPCs to behave differently based on their conditions (health, armor, etc.). Implemented PID controller based steering which enabled smooth trajectory alignment, gradual velocity gain and overall realistic behavior.
- Implemented custom glsl shaders for visual effects, like blasters, mining rays and force fields.
- Implemented resource allocation system which allows reuse entities created during run-time, so less GC load.
- The game features infinite, procedurally generated space and a system which allows it to fetch resources from a sector of space gradually as the player travels through it. There are no loading hang-ups ever. Built custom 3D models for ships in blender. Can be seen on my website.

Education

Master's degree, computer science. Nizhniy Novgorod State Technical University named after R.E. Alekseev (NSTU), 2008 - 2014.