Tuna Gül

Istanbul, Türkiye | tunagul54@gmail.com | linkedin.com/in/tunapro1234 | tunapro.xyz

AI · **Computer engineering** · **Robotics** · **Fun projects** · **Education**



I'm a third-year Civil Engineering student at Boğaziçi and I build things: AI agent systems, robots, and the software that holds them together. Most of my time right now goes into multi-agent systems. I wrote Blueprint, the tool that runs my own fleet of coding agents, and that fleet now does most of the work at Probot Studio, the robotics kit company I founded. Before agents, I wrote everything by hand, from a JSON parser in high school to firmware for competition robots.

I chose Civil Engineering at Boğaziçi on purpose. The people and the network here are worth more to me than what a CS degree somewhere else would teach me, and I have always learned software better by building it than in a classroom. I also love electronics and mechanics, so a lot of what follows ends up as a physical thing.

AI

Agents and orchestration

late 2025 – present

These days agents write most of my code.

Blueprint (bp) GitHub · bp.tunapro.xyz · Go

late 2025 – present

- A single Go binary that connects, watches and supervises long-running CLI agents (Claude Code, Codex, Hermes, OpenCode) on a server. You can drive it as a CLI, as an MCP server or through a local HTTP API.
- The hard part was messaging. When an agent is busy, or its user is halfway through typing, the message waits in a queue instead of interrupting. It only counts as delivered once it shows up in the receiving agent's own transcript.
- Agents on any two machines can talk to each other over peer-to-peer links (libp2p, with Ed25519 machine keys), directly or through a relay when the machines can't reach each other. My agents spread across four machines (a laptop, a home server, an AWS box and a desktop at home) and hand work to each other as if they were on one computer.
- 591 commits, 1,204 Go tests and 50+ signed releases. It runs my fleet every day, with about 50 agents open at once and 200+ registered.

Agents at Probot Studio

2025 – present

Probot is where I use all of this hardest. Separate agents own the website, the simulator, the lessons, the CAD parts and the shop, and they work together through bp. The details are under Robotics.

AI for PCB and CAD design

2026 – present

I'm still experimenting here, and I have built my own tools for it.

- **autoboard** takes a circuit written as Python code, simulates it with ngspice, exports it to KiCad, then routes it and runs DRC. A Gemini agent places the components, and a CLI tracks every stage of the pipeline. Example boards go up to an ESP32-S3.
- For mechanical parts, agents write the CAD as code (build123d) and work against Onshape through its API. My tools turn STEP files into previews and into parts for our browser robot builder.

Human simulation

Mar – Jun 2026

Can LLM agents stand in for people in an economics experiment? I spent spring 2026 finding out.

- **replicant** puts LLM agents with Big Five personalities (BFI-2-Expanded, r = 0.91 against Mini-IPIP) into live oTree experiments, alone or alongside humans. I used it to replicate Ertan, Page & Putterman (2009) and wrote it up as a 14-page paper.
- **edsl-test** compares 9 persona methods across 14 economic games against human data. That came to 10,920 agent decisions with 0.3% invalid, and 8 of 8 paper replications passed.
- **metron** gives psychometric tests to 199 agent persona templates and compares them with human norms. **awesome-llm-social-simulation** is a map of the field in about 500 linked notes, collected by research agents.

Boğaziçi AI UpForAChallenge, 1st place

Oct 2024

We won with an elevator scheduling simulator in Rust, built with Yiğit (his first Rust project). It models car physics with PID motor control and passenger flow, has a deterministic dispatcher to beat, and puts a neuroevolution controller on top. The PyTorch Rust bindings wouldn't compile on my Gentoo machine with LTO on, so I wrote the neural network myself. That turned out faster than fixing the toolchain.

Where it started 2020 – 2021

In high school I bought *Neural Networks from Scratch* and did backprop before I had learned the calculus for it. AI-Playground (2020) was my first network written from zero, committed with the message "Sanırım çalışıyor" ("I think it works"). In cygnus (2021) I rewrote the matrix math in C, with n-dimensional arrays kept off the heap by macros. I also tried running small networks on embedded boards, though most of that never reached GitHub.

Computer engineering

All written by hand, mostly to understand how things work underneath.

pro1238 GitHub · Python 2022

A Unix-like shell with its own command parser, running on a virtual filesystem I stored in my own database format. It has recursive ls, cd, mkdir, rm, history and recall.

vector-h GitHub · C89 2021

C doesn't give you a generic vector, so I wrote one entirely in macros. `vector_t(int)` gives you a typed container with push, insert, growth and merge sort. I wrote two implementations and unit-tested them against each other.

ppps GitHub · Python, C 2021

A preprocessor for C that runs Python code embedded in the source file and writes out the generated C. I published it on pip.

DBEX GitHub · PyPI · Python 2020 – 2021

In the middle of the AI and big data wave, I wanted to load big files without filling up RAM, so I wrote my own JSON parser. It works as a generator: values come out one at a time instead of the whole file sitting in memory, and a depth setting decides how much gets fully loaded. 97 commits, written in 11th grade.

Robotics

Probot Studio, founder 2024 – present

probotstudio.com · probot-core

Probot started as a robotics kit for high-school competitions and grew into a platform for learning robotics. It's probably the thing I care about most.

- **The kit.** Motors, wheels, chassis and **probot-core**, the ESP32 C++ library that drives them (MIT licensed). We make the parts ourselves. Every team on the field at MEB İMFEST Tasarla-Geliştir ran on it, and it's still used at competitions.
- **Free tools on the site. Builder** lets you put a robot together in 3D in the browser, **Blocks** lets you program it with drag-and-drop blocks, and **Sim** lets you drive it in a simulator before you have built anything.
- **Lessons.** A pipeline of agents writes each lesson, then reviews it from three points of view: a student, a teacher and an expert educator. Slides and images are generated with AI, the narration is an AI voice, and the result becomes a lesson video. All of it runs on our own LMS, with classes and a teacher panel.
- **How it runs.** Most of the code, CAD and content is made by agents on bp (see AI).

FTC Zağanos #25577 2025 – present

software mentor in DECODE 2025–26, team lead in BIOBUZZ 2026–27

- **BIOBUZZ, this season.** A hierarchy of AI agents writes the robot software, and each phase has to pass its tests before the next one starts. Everything is public in boobuzz-docs.
- **What we built in DECODE.** About 70,800 lines of robot code in one season: a strategy engine, A* and DWA motion planning, EKF and Limelight-based localization, vision-based ball tracking, an auto-tuning framework that characterized the mechanisms by itself, and a simulator that ran the real Java robot code through JPype. It's all written up topic by topic in docs/miras.
- **The DECODE software document** (Dec 2025, 63 pages, English and Turkish), on the software of our shooter-turret robot.
  - The same code ran on the real robot and in a PyBullet simulator, so we debugged the whole logic stack before the mechanical build was done. Shots were solved with full ball physics integrated by RK4, optimized with Nelder–Mead, and exported to lookup tables the robot could use in real time. In Monte Carlo runs in simulation this hit 77% on average.
  - Robot pose came from Kalman fusion of odometry and AprilTags. Balls were tracked with a Hungarian-matching, Kalman and MHT tracker, and sensor offsets were calibrated with differential evolution. We also fed the turret's PWM absolute encoder through an analog input, which gave us reliable 360° turret control.

FIRST Robotics Competition

2020 – 2026

I started on FRC 7839, then founded team NF (FRC 9029), and we went to the FIRST Championship in Houston in 2025 and 2026. I extended our fork of the open-source YAGSL swerve library (nf-swerve-lib) to support VictorSPX controllers, so we could run swerve on the motor controllers we already had.

Early hardware

2020

MEGABOI is a handheld game console I built on an Arduino Mega with a touchscreen, six buttons and an ESP8266. I wrote Snake for it, and it kept running out of memory. My Arduino turned out to be a clone with less RAM than a real one, so I had to squeeze everything to make it fit. That project taught me how much optimization matters, and it was one of the big reasons I later installed Gentoo and put my own kernel together from scratch.

Fun projects

Recent things I had far too much fun building.

Lighthouse

2026 – present

- Agents search the internet for programs I could apply to, and then apply for me.
- Another set of agents finds hackathons and works on them for me. Each event gets a lead agent and a builder, plus judge and idea agents that argue for at most three rounds. Once they went 12 rounds and produced 98 ideas, and none of them beat the first one.
- Every morning at 06:00 it all comes out of my printer as a newspaper. 84 issues so far.

Necati @necatea_

Oct 2026 – present

An agent swarm built to maximize the amount of laughter in the world. It started as a WhatsApp bot in a group of friends and keeps evolving. Right now about 8 agents share a memory of which jokes landed, which bombed and why, and its default mode is staying quiet. A lot of my friends really love talking to it, and honestly, it's funny. On X it's learning to get funnier: a UCB1 bandit picks what to try next, and the reward is engagement adjusted for follower count. It has about one follower so far. We'll see where it goes.

Home book press

Some books I want aren't in print here, or take a month and 5,000+ TL to import, so I print and bind them at home. The simple pipeline finds the text body in a PDF, centres it and fits it to my paper and binding margins. The bigger one OCRs the book and typesets it from scratch in LaTeX, and agents translate the extracted text. Since it's my own press, I add blank pages so a book doubles as a notebook, and I 3D-print covers with a slot so my pens stop getting lost or broken.

Home server and local models

2025 – present

My home server is a friend's old laptop, on a static IP my dad pays for. It's a small machine, but it runs my agent fleet, mail, Gitea, Home Assistant, a 3D-printer agent and our Minecraft world. Before RAM prices exploded I also put 64 GB into my ThinkPad to run uncensored models locally (I use Arch, btw).

Education

Boğaziçi University, Istanbul B.S. in Civil Engineering, 3rd year — 2023 – 202∞ (expected)

AI Lead at CompEC, Boğaziçi's computing club. We give AI lessons every week.

Beşiktaş Atatürk Anadolu Lisesi, Istanbul — 2018 – 2022

Skills

Code: Python, Go, TypeScript/JavaScript, Rust, C/C++, Java

AI: Claude Code, Codex, Anthropic, OpenAI and Gemini APIs, MCP, multi-agent orchestration, PyTorch

Systems and hardware: Linux (Arch, Gentoo), systemd, Docker, Git, ESP32, Arduino, KiCad, build123d

Languages: Turkish (native), English (fluent)