Eren Işık

(+90) 505 548 35 96 | erenisik@std.iyte.edu.tr | linkedin/in/eren-işık-271914349 | github/Ursidae0

Eğitim

İzmir Yüksek Teknoloji Enstitüsü — İzmir, Türkiye
Bilgisayar Mühendisliği Lisans — GPA: 3.25/4 — *2023 – 2027*

Deneyim

Cengaver Rover — İzmir, Türkiye
Yazılım Ekibi Üyesi — TEKNOFEST Üniversite Rover Yarışması — *Eylül 2025 – Günümüz*

- Rover'ın sürüş yazılımını MicroPython'dan C'ye (Raspberry Pi Pico SDK) taşıyarak yoklama döngüsünü 1 kHz'lik gerçek zamanlı kontrol döngüsüyle değiştirdim; orijinal sistemde bulunmayan CRC-8 paket bütünlüğü, donanım watchdog ve Dead Man's Switch özelliklerini ekledim.
- USB CDC üzerinden özel bir ikili iletişim protokolü tasarladım; eşleşen ana bilgisayar tarafı sürücüsünü Jetson üzerinde çalışan bir ROS 2 düğümü olarak uyguladım.
- İletişim zaman aşımında yazılım devreden çıkarma, donanım watchdog sıfırlaması ve GPIO düzeyinde motor kesiminden oluşan çok katmanlı güvenlik sistemi kurdum. (bağlantı kaybında veya yazılım arızasında güvenli durma sağlar)

Projeler

Seyrek GPU Çekirdekleri (Makale Yeniden Üretimi) | GitHub | *CUDA, C++, Docker* — Aralık 2025 – Ocak 2026

- Sputnik CUDA kütüphanesini (Gale ve ark., SC'20) NVIDIA NGC konteynerı içinde, güncel Linux çekirdeği ve NVIDIA sürücü sürümleriyle uyumlu olacak şekilde yapılandırdım.
- SpMM ve SDDMM çekirdeklerini RTX 4070 Super üzerinde cuSPARSE temelleriyle kıyasladım; %99 seyreklikte yoğun cuBLAS'a kıyasla 7,8x hız artışı ve 4,3 TFLOPs'ı aşan SpMM verimi elde ettim.
- Makalede bildirilen %70,5 seyreklik başabaş noktasını (break-even point) yeniden oluşturdum (makale V100'de ∼%71 belirtmektedir); çekirdek verimliliğinin GPU nesilleri arasında ölçeklendiğini doğruladım.

NVIDIA Jetson Nano Üzerinde Sinir Ağı Çıkarımı | GitHub | *Python, Docker, CUDA* — Aralık 2025

- NVIDIA Jetson Nano (Tegra X1) üzerinde TensorRT ile optimize edilmiş üç model (PeopleNet, SSD-Inception-v2, SSD-MobileNet-v2) konuşlandırdım ve karşılaştırdım; PeopleNet gömülü GPU'da görüntü başına ∼650 ms ile en hızlı çıkarımı gösterdi.
- nvprof ile tüm modellerde SM verimliliğini profilledim; kaynaştırılmış evrişim çekirdekleri SM kullanımını %99,9'a kadar çıkardı, NMS sonrası işlem ∼%12,6 SM verimiyle ana darboğaz olarak belirledim.

Bilgisayarla Görme Proje Portföyü | GitHub | *Python, OpenCV, scikit-learn, scikit-fuzzy* — Aralık 2025 – Ocak 2026

- **Kalabalık Sayma** — ön plan maskelerinden morfolojik özellikler (kapama/açma) çıkararak kalabalık sayısını tahmin eden bir MLP eğittim; en iyi yöntem %10,64 MAPE elde etti; tahminlerin %86,27'si gerçek değerin %15'i dahilinde kaldı.
- **Retinal Kan Damarı Çıkarımı (Makale Yeniden Üretimi)** — fundus görüntülerinden retinal damar çıkarımı için Python'da (OpenCV, scikit-learn, scikit-fuzzy) MATLAB tabanlı hibrit FCM + denetimli işlem hattını yeniden kurguladım; DRIVE test seti üzerinde AUC 0,86 ve %92,6 piksel doğruluğu elde ettim; makalenin bildirdiği AUC 0,975 ile elde edilen değer arasındaki farkın standart ML kütüphanelerinde bulunmayan tescilli Kök Rehberli Karar Ağacı'ndan (Root Guided Decision Tree) kaynaklandığını saptadım.

Teknik Beceriler

Programlama: Python, C/C++, CUDA
Kütüphaneler: scikit-learn, OpenCV, NumPy, Matplotlib, scikit-fuzzy
Araçlar: Git, Docker, CMake, Bash, nvprof, Raspberry Pi Pico SDK (ARM Cortex-M0+)
Diller: Türkçe (Anadil), İngilizce (B2)