📍

# geo-sleuth: Başlangıç Rehberi

Bir fotoğrafı yapay zeka ajanına ver, nerede çekildiğini sor. Ajan ipuçlarını okuyor, harita ve yükseklik verisiyle hesap yapıyor, kameranın yerini bir hata payıyla ve kanıt görüntüsüyle veriyor.

Açık kaynak · MIT | ⭐ 4.400+ GitHub yıldızı | 6 ajanla çalışır | Kendi bilgisayarında çalışır

## 🔗 DOĞRULANMIŞ LİNKLER

- ⭐ GitHub (resmi kaynak): **github.com/Oldcircle/geo-sleuth**
- 🧰 uv (Python paket yöneticisi): **docs.astral.sh/uv**
- 🤖 Çalıştığı ajanlar: Claude Code, Codex, Cursor, Gemini CLI, OpenCode, GitHub Copilot

## 🧩 VİDEODAKİ ÖRNEK NASIL ÇÖZÜLDÜ

Pirinç tarlasında bir fırın, arkada bir viyadük ve bir dağ. Yazı yok, konum bilgisi (EXIF) silinmiş.

`27.335 köprü parçası` → `171 yer` → `14.372 kamera noktası` → `22` → `3` → `1` → `±2 m`

1. **Fotoğrafı okudu:** viyadükteki direkler elektrik hattı taşıyor, yani elektrikli demiryolu. Pirinç biçilmiş ama çimen yeşil, yani henüz don yok.
2. **Bölgeyi taradı:** bölgedeki tüm demiryolu köprülerini OpenStreetMap'ten çekti, her 400 metrede bir noktadan 360° ufku yükseklik verisiyle hesapladı.
3. **Silüeti eşleştirdi:** aday noktalardan görünen dağ çizgisini fotoğraftakiyle karşılaştırdı, en iyi üçünü fotoğrafa bindirip birini seçti.
4. **Köprü ayaklarını saydı:** 17 ayağa 17 yön çizgisi çekti; aralıkları eşit yapan tek nokta kameranın yeri oldu.

## ⚡ KURULUM

Gerekenler: Node.js 22.20+, Python 3.10+, uv, curl, Google Chrome ya da Playwright Chromium

```
npx skills add Oldcircle/geo-sleuth
# sorduğunda ajan olarak Claude Code'u seç

# uv yoksa (Mac / Linux):
curl -LsSf https://astral.sh/uv/install.sh | sh

# Chrome yoksa:
uvx playwright install chromium
```

## 🧭 KULLANIM

1. Claude Code'u aç, fotoğrafı ekle.
2. İlk seferde ajana yeteneğin **doctor.py** kontrolünü çalıştırmasını ve eksikleri düzeltmesini söyle.
3. Projenin örnek isteği:

```
find where this photo was taken
```

Sonuç: kamera konumu ± hata payı, kameranın baktığı yön, uydu kanıt görüntüsü ve güven derecesi.

## ⚠️ BİLMEN GEREKENLER

- ⏱️ Videodaki örnek yaklaşık **72 dakika** sürdü, yarısı hesaplama beklemesi.
- 🎯 Sonuç garanti değil. ±2 m, yöntemin o fotoğraftaki ilk çözümü; README farklı model ya da kararlarla başka sonuca varılabileceğini söylüyor. Kör denemelerde 87 m (32 dk, "Çin'de" ipucuyla) ve 3 m (53 dk).
- 💳 Claude Code ücretsiz planda yok; en ucuz seçenek Pro (aylık 20 $, yıllık ödemede ayda 17 $, vergi hariç).
- 📦 Python paketlerini ilk çalıştırmada uv kendisi kuruyor; ilk açılış uzun sürebilir.

## 🔒 SORUMLU KULLANIM VE KENDİ GİZLİLİĞİN

✅ Yalnız kendi fotoğraflarında ya da izin aldığın fotoğraflarda kullan. Projenin kuralı: bulunmayı kabul etmemiş kişileri bulmak için asla.

📍 Konumu kapatmak yetmiyor. Arkadaki dağ, köprü, tabela ya da gölge bile yeri ele verebiliyor. Ev ve iş çevrende çektiğin fotoğrafı paylaşmadan önce arka plana bir bak.



Bilgiler 29 Ekim 2026'da projenin GitHub sayfası ve README dosyasından doğrulandı. Rakamlar README örnek 1'den.