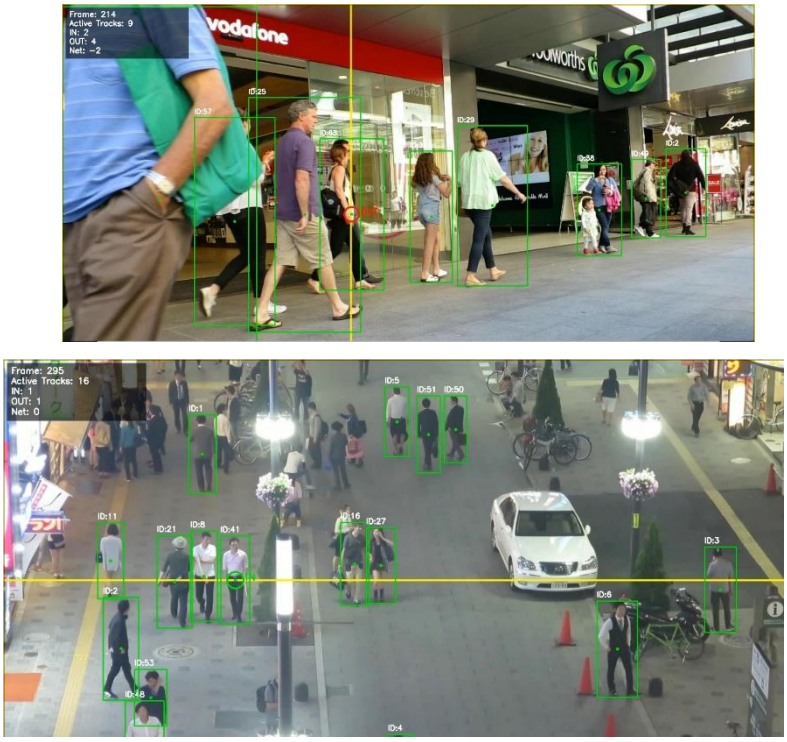
Study Case 2 - Hazır Modellerle Video Analizi Takip (Tracking) + Olay Mantığı (Counting) Yaklaşım Özeti

Bu çalışma, MOT17 veri seti üzerinde nesne tespiti, takibi ve belirli bir hattan geçiş yapan kişilerin sayımı (giriş-çıkış analizi) amacıyla kurgulanmıştır. Projede, gerçek zamanlılık ve CPU verimliliği hedefleri doğrultusunda YOLO11 modelinin iki farklı varyantı, iki ayrı takip (tracking) algoritmasıyla eşleştirilerek karşılaştırmalı bir yapı kurulmuştur. Bu kapsamda hız öncelikli sahneler için **YOLO11n + BoT-SORT** ve doğruluk odaklı senaryolar için **YOLO11s + ByteTrack** kombinasyonları seçilmiştir.

Takip aşamasında karşılaşılan örtüşme (occlusion) ve kimlik değişimi (ID switch) gibi temel zorlukları aşmak için BoT-SORT algoritması, Re-ID (yeniden tanımlama) özelliklerini ve Kalman filtresi tahminlerini optimize eden özel bir yapılandırma dosyası (botsort.yaml) ile desteklenmiştir. Özellikle kalabalık sahnelerde takibin kopmasını engellemek adına kayıp izlerin hafızada tutulma süresi (track_buffer) artırılmış ve Re-ID benzerlik eşikleri sıkılaştırılmıştır. Buna ek olarak, post-processing aşamasında devreye giren TrackIDStitcher modülü, kopan izleri mekânsal ve zamansal yakınlıklarına göre analiz ederek parçalanmış track segmentlerini tek bir ID altında birleştirmiş, böylece veri bütünlüğü korunmuştur.

Olay mantığı (Counting), video düzlemine yerleştirilen sanal çizgiler ve vektörel çarpım (cross-product) tabanlı bir algoritma üzerine inşa edilmiştir. Çizgiyi geçen her bir nesnenin yönü, hareket vektörünün sanal hat ile olan geometrik ilişkisine göre "IN" veya "OUT" olarak sınıflandırılmaktadır. Çift sayımı (double counting) önlemek amacıyla, LineCrossingCounter sınıfı içerisinde her nesne kimliği için bir soğuma süresi (min_frames_between_crossings) ve pozisyon geçmişi tanımlanmıştır. Nihai çıktılar; standart MOT formatında takip dosyaları, detaylı olay logları ve analiz sürecini kolaylaştıran görsel overlay videoları olarak sunulmaktadır.

Örnek Overlay Çıktıları



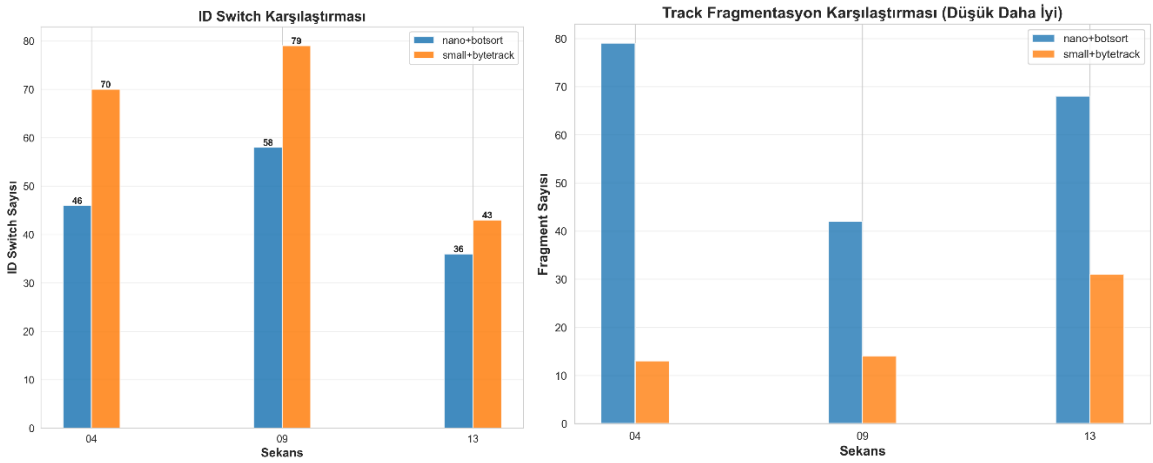
Şekil 1-2. Örnek Overlay Çıktıları

Seçilen Metrikler, Tablolar ve Yorum

Tablo 1. Tahmin ve Gerçek Veri Karşılaştırılması

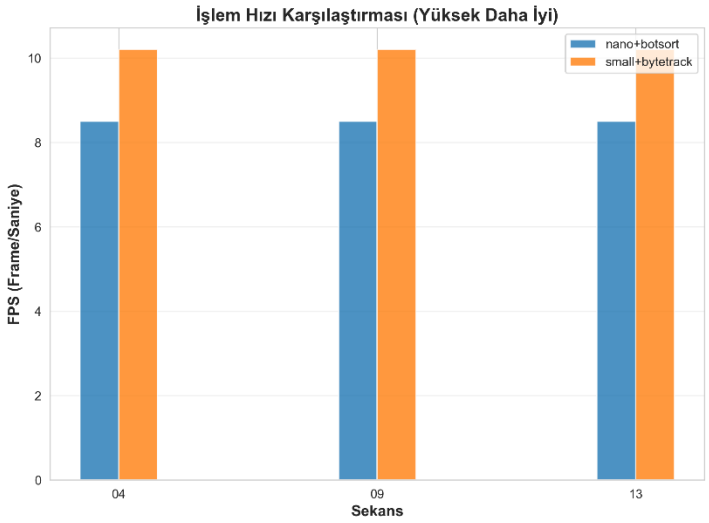
Sekans	Gerçek (GT)	Tahmin	Doğruluk	Başarı (F1)
04 nano+botsort	13	12	%92	%96
04 small+bytetrack	13	13	%100	%100
09 nano+botsort	15	14	%93	%96
09 small+bytetrack	15	14	%93	%96
13 nano+botsort	45	25	%55	%71
13 small+bytetrack	45	32	%71	%83

MOT17-04 ve 09 sekanslarında her iki modelin de %90'ın üzerinde doğruluk sağladığı görülmektedir. Ancak MOT17-13 sekansında başarının %55-%71 bandına düştüğü, bu sekans kalabalık yapısı nedeniyle projenin en zorlu testi olarak değerlendirilmiştir.



Şekil 3-4. ID Switch ve Track Fragmentasyon Grafikleri

BoT-SORT algoritmasının ByteTrack'e kıyasla sergilediği ID switch başarısı, botsort.yaml konfigürasyonundaki stratejik seçimlere dayanmaktadır. Özellikle *match_thresh: 0.8* ile getirilen sıkı eşleşme kriteri ve *with_reid: true* ile aktif edilen görünüm tabanlı kimlik doğrulama, nesnelerin birbirine yakın olduğu anlarda yanlış eşleşme riskini minimize etmiştir. MOT17-13 gibi kalabalık ve hareketli sahnelerdeki stabilite ise *track_buffer: 60* ve *gmc_method: sparseOptFlow* parametreleriyle güçlendirilmiştir. Kayıp izlerin 60 kare boyunca hafızada tutulması kısa süreli örtüşmelerde (occlusion) takibin kopmasını engellerken, küresel hareket telafisi (GMC) kamera sarsıntılarından kaynaklanan konum kaymalarını dengelemiştir. "nano+botsort" kombinasyonunda fragment sayısının daha yüksek olması, nano modelin küçük nesneleri algılamada daha sık kopma yaşadığından kaynaklanmaktadır.



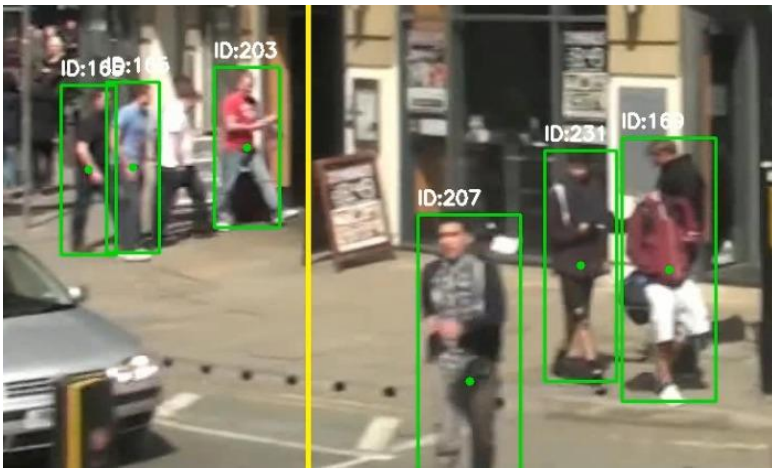
Şekil 5. İşlem Hızı Karşılaştırma Grafiği

Hız öncelikli sahnelerde YOLO'nun nano modelinin daha avantajlı olacağı düşünülmesine karşın YOLO11s + ByteTrack kombinasyonunun (~10 FPS), YOLO11n + BoT-SORT (~8.5 FPS) kombinasyonundan daha hızlı çalıştığı gözlemlenmiştir. Bu durum, BoT-SORT içerisindeki Global Motion Compensation (GMC) ve Re-ID özelliklerinin CPU üzerindeki hesaplama yükünün, model boyutundaki farktan daha baskın olduğunu göstermektedir.

Karşılaşılan Zorluklar

Şekil 6. Tekrarlayan Sayım Hatası

Bir kişinin camdan yansıyan görüntüsünün dedektör tarafından ayrı bir nesne olarak algılandığı ve takip algoritmasının bu yansımayı orijinal kişiyle aynı kimlik altında (ID: 28) eşleştirdiği görülmektedir. Bu durum, "False Positive" algılamaların takip sürekliliğini bozmasına ve özellikle yansımalı yüzeylerin bulunduğu geçiş hatlarında tekrarlayan sayım (double counting) riskine yol açmaktadır.



Şekil 7. Örtüşme Problemi Örneği

Kalabalık sahnelerin en büyük zorluğu olan "Occlusion" (örtüşme) problemi yaşanmaktadır. Fiziksel olarak çok yakın yürüyen veya üst üste binen iki farklı kişinin dedektör tarafından tek bir sınırlayıcı kutu (ID: 169) içerisinde gruplanması, sistemin bir kişiyi "miss" (kaçırma) yapmasına neden olmuştur.

İyileştirme Yapsaydım Ne Denerdim?

Donanım Seviyesinde Optimizasyon ve Model Ölçeklendirme: Çıkarım (inference) performansını artırmak amacıyla modellerin OpenVINO veya ONNX Runtime gibi donanıma özgü kütüphanelere dönüştürülmesi hedeflenebilir. Bu tür bir optimizasyon stratejisi ile elde edilecek FPS (Frames Per Second) artışı, donanım darboğazını aşarak daha yüksek doğruluk ve öznitelik çıkarma kapasitesine sahip olan YOLO11m veya YOLO11l gibi üst segment modellerin gerçek zamanlıya yakın senaryolarda CPU üzerinde koşturulmasına imkan tanıyabilir.

Bölge Bazlı Dinamik Sayım ve Durum Takibi (ROI Corridors): Mevcut tek boyutlu sanal hat metodolojisi, nesne merkez noktalarının çizgi üzerindeki anlık salınımlarına (jitter) ve duraksamalara karşı duyarlılık göstermektedir. Nesnenin sadece bir koordinat eşiğini geçmesi yerine, tanımlanmış bir alan dizisi boyunca mantıksal bir rotayı tamamlaması şartının aranması, özellikle kalabalık geçiş noktalarında karşılaşılan "false-positive" sayım hatalarını ve tekrarlayan kayıtları minimize edebilir.