**Kamera ile Gerçek Zamanlı Araç Hız Tespiti: YOLOv8 ve Matematiksel Hesaplamaların Entegrasyonu**

**Özet**

Bu makale, bilgisayarlı görü ve derin öğrenme tekniklerini kullanarak gerçek zamanlı araç hız tespiti yapmayı amaçlayan bir sistemin tasarımını ve uygulamasını detaylandırmaktadır. Sistem, YOLOv8 (You Only Look Once) nesne tespit modeli ve matematiksel hesaplamaların entegrasyonu ile geliştirilmiştir. Makalede, sistemin çalışma prensipleri, kullanılan algoritmalar, matematiksel hesaplamalar ve elde edilen sonuçlar detaylı bir şekilde incelenmektedir. Ayrıca, sistemin performansı ve potansiyel uygulama alanları da tartışılmaktadır.

**1. Giriş**

Günümüzde, trafik yönetimi ve güvenliği, özellikle yoğun şehirlerde büyük bir sorun haline gelmiştir. Araçların hızlarının tespiti, trafik kurallarının uygulanması, kazaların önlenmesi ve trafik akışının optimize edilmesi açısından kritik öneme sahiptir. Geleneksel hız tespit yöntemleri, radar ve lazer tabanlı sistemler gibi pahalı ve kurulumu zor olan cihazlara dayanmaktadır. Bu çalışmada, bilgisayarlı görü ve derin öğrenme tekniklerini kullanarak daha ekonomik ve esnek bir çözüm sunmayı amaçlayan bir sistem önerilmektedir.

**2. Yöntem**

**2.1. YOLOv8 Nesne Tespit Modeli**

YOLOv8, nesne tespiti için kullanılan en son ve en gelişmiş modellerden biridir. YOLO mimarisi, tek bir evrişimsel sinir ağı kullanarak görüntüdeki nesneleri tespit eder ve sınıflandırır. YOLOv8, önceki versiyonlara göre daha yüksek doğruluk ve hız sunar, bu da onu gerçek zamanlı uygulamalar için ideal kılar.

Bu çalışmada, YOLOv8 modeli, "coco.txt" dosyasında tanımlanan sınıflar kullanılarak eğitilmiştir. Model, görüntüdeki araçları tespit etmek ve sınıflandırmak için kullanılmıştır. Tespit edilen araçların sınırlayıcı kutuları (bounding boxes) ve merkez noktaları (centroids) hesaplanarak hız tespiti için kullanılmıştır.

**2.2. Araç Takibi ve Hız Hesaplama**

Araçların hızlarını hesaplamak için, ardışık karelerdeki araçların konumları arasındaki mesafe ve zaman farkı kullanılmıştır. Her bir araç için, önceki karedeki konum (cx, cy) ve zaman (t) bilgileri saklanmıştır. Ardışık karelerdeki konum değişiklikleri ve zaman farkı kullanılarak araçların hızları hesaplanmıştır.

Hız hesaplaması için aşağıdaki formül kullanılmıştır:

Hız=MesafeZaman FarkıHız=Zaman FarkıMesafe​

Burada, mesafe, iki ardışık karedeki araç merkez noktaları arasındaki Öklid mesafesi olarak hesaplanmıştır:

Mesafe=(cx2−cx1)2+(cy2−cy1)2Mesafe=(cx2​−cx1​)2+(cy2​−cy1​)2​

Zaman farkı ise, ardışık karelerin zaman damgaları arasındaki fark olarak hesaplanmıştır.

**2.3. Matematiksel Hesaplamalar ve Gerçek Dünya Ölçeklemesi**

Hız hesaplamasında, piksel cinsinden mesafelerin gerçek dünya ölçeklerine dönüştürülmesi gerekmektedir. Bu dönüşüm, araçların sınırlayıcı kutularının boyutları ve kamera kalibrasyonu kullanılarak yapılmıştır. Araçların gerçek boyutları ve kamera açıları göz önünde bulundurularak, piksel cinsinden mesafeler metre cinsine dönüştürülmüştür.

Bu dönüşüm için aşağıdaki formül kullanılmıştır:

Gerc¸ek Mesafe=Piksel MesafesiO¨lc¸ek Fakto¨ru¨Gerc¸​ek Mesafe=O¨lc¸​ek Fakto¨ru¨Piksel Mesafesi​

Ölçek faktörü, araçların sınırlayıcı kutularının genişlik ve yükseklikleri kullanılarak hesaplanmıştır. Bu hesaplama, araçların gerçek boyutları ve kamera açıları göz önünde bulundurularak yapılmıştır.

**2.4. Açı Hesaplamaları**

Araçların hareket yönünü belirlemek ve hız hesaplamalarını daha doğru hale getirmek için açı hesaplamaları yapılmıştır. Bu hesaplamalar, araçların merkez noktaları arasındaki mesafe ve bu mesafenin x ve y eksenlerindeki bileşenleri kullanılarak yapılmıştır.

Açı hesaplamaları için kosinüs teoremi kullanılmıştır:

cos⁡(A)=b2+c2−a22bccos(A)=2bcb2+c2−a2​

Burada, aa, bb, ve cc üçgenin kenar uzunluklarını temsil etmektedir. Bu hesaplamalar, araçların hareket yönünü belirlemek ve hız hesaplamalarını daha doğru hale getirmek için kullanılmıştır.

**3. Sonuçlar ve Tartışma**

Bu çalışmada, YOLOv8 modeli ve matematiksel hesaplamaların entegrasyonu ile gerçek zamanlı araç hız tespiti yapılmıştır. Sistem, farklı trafik koşullarında test edilmiş ve başarılı sonuçlar elde edilmiştir. Elde edilen sonuçlar, sistemin doğruluk ve performans açısından tatmin edici olduğunu göstermektedir.

Ancak, sistemin performansı, kamera kalibrasyonu, ışık koşulları ve araçların boyutları gibi faktörlere bağlı olarak değişebilmektedir. Bu nedenle, sistemin daha geniş bir yelpazede kullanılabilmesi için daha fazla test ve iyileştirme yapılması gerekmektedir.

**4. Kaynakça**

1. Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.
2. Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.
3. Bradski, G., & Kaehler, A. (2008). Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media, Inc.
4. Hartley, R., & Zisserman, A. (2003). Multiple View Geometry in Computer Vision. Cambridge University Press.
5. Forsyth, D. A., & Ponce, J. (2012). Computer Vision: A Modern Approach. Pearson Education.
6. Szeliski, R. (2010). Computer Vision: Algorithms and Applications. Springer Science & Business Media.