Erciyes Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
Cilt 37, Sayı 3, 2021

Erciyes University
Journal of Institue Of Science and Technology
Volume 37, Issue 3, 2021

Gerçek Zamanlı Fare Takip ve Analiz Yazılımı Geliştirilmesi

Hüseyin Güneş *1

*1 Balıkesir Üniversitesi Mühendislik Fakültesi Mekatronik Mühendisliği, BALIKESİR

(Alınış / Received: 06.11.2020, Kabul / Accepted: 09.10.2021, Online Yayınlanma / Published Online: 30.12.2021)

Anahtar Kelimeler
Fare Takip,
Davranış Analizi,
Gerçek Zamanlı,
Görüntü İşleme

Öz: Davranışsal araştırmalarda, önce hayvan davranışlarını kaydedip daha sonra videoları izlemek ve gözlemleri analiz etmek bireysel yorumlama farklılıkları, dikkatsizlik veya yanlılık gibi nedenlerle analizlerde hata olasılığını arttırmaktadır. Bunu azaltmak için geliştirilen video takip sistemleri ile manuel kayıt almaya kıyasla daha güvenli ve tutarlı sonuçlar elde edilmektedir. Ancak mevcut video takip sistemlerinin de birtakım eksiklikleri/kısıtlılıkları (birden fazla nesneyi takip edememe, anlık takip edememe ya da analiz edememe vb.) bulunmaktadır. Bu nedenle bu çalışmada davranış deneylerinin takip ve analizinde kullanılabilecek bir yazılım geliştirilmiştir. Yazılım ile deney anında canlı olarak kameradan alınan görüntüler çeşitli görüntü işleme yöntemleri kullanılarak analiz edilmektedir. Analizler sonucunda araştırmacı tarafından belirlenen alanlarda, farenin ne kadar süre geçirdiği, bu alanlara giriş çıkış frekansları, farenin hızı, toplam kat ettiği mesafe vb. birçok veri anlık olarak hesaplanabilmekte ve düzenli olarak kayıt altına alınabilmektedir.

Development of Real-Time Mouse Tracking and Analyzing Software

Keywords
Mouse Tracking,
Behavior Analysis,
Real Time,
Image Processing

Abstract: In behavioral research, recording animal behavior first and then watching and observing the videos increase the probability of errors in the analysis due to reasons such as individual interpretation differences, carelessness or bias. With the video tracking systems developed to reduce this problem, safer and more consistent results are obtained compared to manual recording. However, the existing video surveillance systems also have some deficiencies / limitations (not being able to track more than one object, not following them instantly or analyzing them, etc.). Therefore, in this study, a software that can be used in the follow-up and analysis of behavioral experiments has been developed. With the software, images taken from the camera live during the experiment are analyzed using various image processing methods. As a result of the analysis, how much time the mouse spent in the areas determined by the researcher, the input and output frequencies, the speed of the mouse, the total distance traveled and many similar data can be calculated instantly and recorded regularly.

***İlgili Yazar, e-mail:** hgunes@balikesir.edu.tr

1. Giriş

Psikoloji, fizyoloji, farmakoloji, sinirbilim gibi pek çok bilim dalı tarafından davranışın nörobiyolojik temellerinin araştırılması ve öğrenme-bellek, anksiyete, stres ve otizm, depresyon gibi psikiyatrik hastalıkların deney hayvanlarında modellenmesi ile ilgili çalışmalar uzun yıllardır yürütülmektedir. Örneğin, farelerin ve sıçanların davranışlarını değerlendirmek, nöropsikiyatrik hastalıklar hakkındaki anlayışımızı ilerletmek ve yeni ilaç ve aşıların gelişimdeki etkilerini test etmek için kullanılmaktadır [1]. Nöropsikiyatrik hastalıkların hayvan modelleri sosyal davranışlarda bozulma ile karakterizedir. Bu nedenle kemirgenlerde sosyal davranışların değerlendirilmesi bu tür çalışmalarda oldukça önemlidir [2].

Kemirgenler ile ilgili davranışsal araştırmalarda canlıların videolarını kaydetmek ve daha sonra bunları kayıtlı görüntülerden analiz etmek araştırmacılar arasında halen yaygın bir uygulamadır. Ancak bu yaklaşım, bireysel yorumlama farklılıkları, dikkatsizlik veya yanlılık gibi nedenlerle analizlerde hatalarda neden olabilmektedir [1]. Davranışın farklı boyutlarını değerlendirme olanağı sunan çeşitli deney düzeneklerinin geliştirilmesi ve kullanımının yaygınlaşması, manuel olarak gerçekleştirilen gözlem ve kayıtların yerine otomatize sistemlerin kullanımını daha elverişli hale getirmiştir. Bu amaç için geliştirilen video takip sistemleri, manuel kayıt almaya kıyasla daha güvenli ve tutarlı bir şekilde ve daha uzun zaman dilimlerinde davranışların incelenmesini sağlamaktadır [3].

Davranış analizi ile ilgili olarak halihazırda geliştirilmiş farklı özelliklere sahip akademik ve ticari video takip yazılımları bulunmaktadır. Genel olarak ticari yazılımlar kendi ürettikleri platformlarda takip özelliklerine sahiptir [4]. Bunun yanında videolardan ve kameralardan farklı platformlarda takip işlemi gerçekleştiren ticari yazılımlar da mevcuttur [5] [6]. Ancak bu yazılımlar oldukça yüksek fiyatlı yazılımlardır. Bu alanda geliştirilmiş akademik çalışmalar ele alındığında ise UMA Tracker [7] ve ToxTrac [8] isminde iki çalışmaya rastlanmıştır. Bu çalışmaların ikisi de sadece videolar ile çalışabilmektedir. Canlı görüntü üzerinden analiz yapamamaktadır.

Bu çalışmada akademik ve ticari çalışmalardan farklı olarak gerçek zamanlı analiz yapan ve anında çıktıyı hızlı bir şekilde üretebilen, herkes tarafından kolayca kullanılabilecek tek ekranda birkaç tıklama ile bütün işlemlerin yapılabileceği bir deney takip ve analiz yazılımı ortaya çıkarmak hedeflenmiştir.

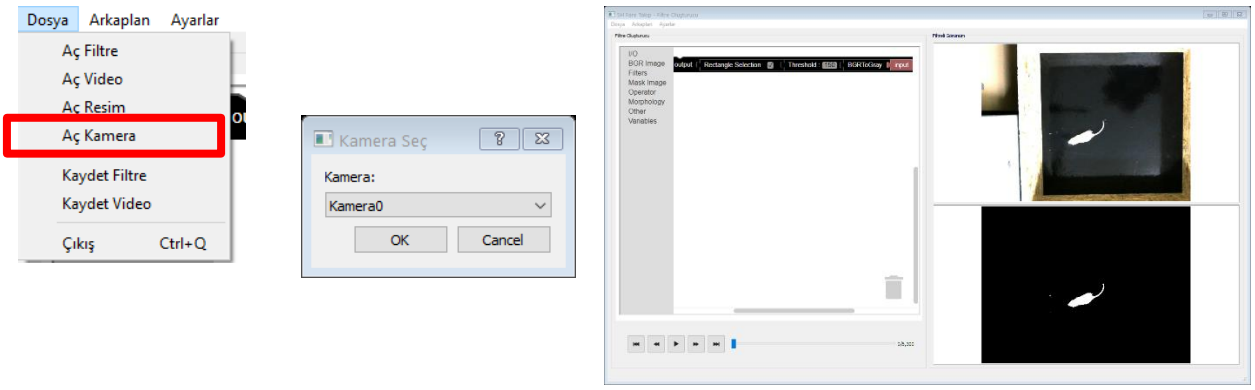
2. Materyal ve Metot

Bu çalışmanın amacı deneysel davranış çalışmalarında kullanılmak üzere bir video takip sistemi geliştirmektir. Yazılım python programlama dili ile açık kaynak kodlu, daha önce kaydedilen deney videolarını kısıtlı özelliklere göre analiz edebilen UMATracker isimli yazılım temel alınarak geliştirilmiştir [7].

UMATracker deney anında kaydedilmiş videolarda bulunan farelerin doğru bir şekilde algılanabilmesi için bir görüntü filtre oluşturma aracı, videodan fareyi takip eden bir takip aracı ve takibi kullanıcının belirleyeceği alanlara göre yapan bir alan takip aracından oluşmaktadır. Bu araç ile doğrudan kameradan canlı olarak analiz yapılamamaktadır. Ayrıca farenin ortalama hızı, en yüksek hızı, toplam kat ettiği mesafe (metrik olarak), alanlara giriş-çıkış frekansları analiz edilememektedir. Bu çalışmada açık kaynak kodlu bu araç bu özellikleri kapsayacak şekilde geliştirilmiştir. Geliştirilen araçlarla bir deneyin takip süreci şu şekilde ilerlemektedir; bir deney ön çekimi, bu çekim görüntülerinden filtre aracı ile filtre oluşturulması ve ardından takip aracında bu filtrenin kullanılarak gerçek zamanlı deney ve takiplerinin yapılması.

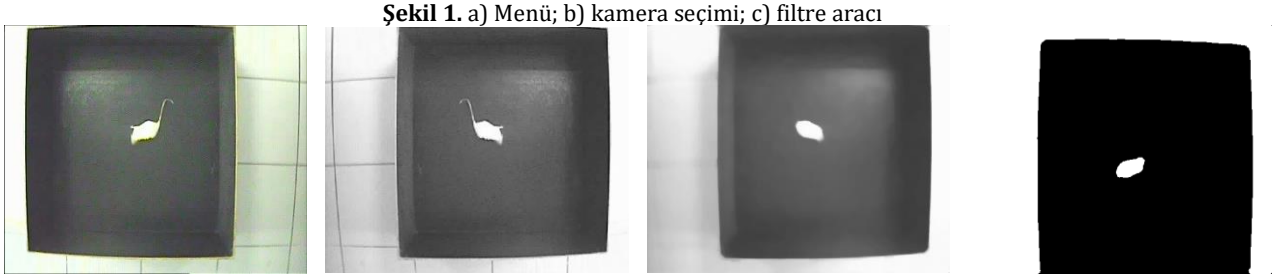
2.1. Filtre aracı

Çalışmada ilk olarak filtre aracı üzerinde geliştirmeler yapılmıştır. Filtre aracı içerisinde çeşitli görüntü işleme seçenekleri bulunmaktadır. Bunlara örnek olarak eşikleme, gri tonlamaya dönüştürme, ortanca filtre, bulanıklaştırma vb. verilebilir. Bunun yanında literatürde anomali tespitinde kullanılan yöntemlerden [18] faydalanarak arka plan görüntüsünü filtreleyen ve bu şekilde fareyi tespit eden bir filtre de mevcuttur. Kullanıcı bu filtreleri kullanarak fareyi arka planda net bir şekilde ayırır ve takip algoritmaları da bu sayede fareyi başarılı bir şekilde takip edebilmektedir. Bu kadar farklı görüntü işleme seçeneğinin olması farklı ortamlarda farklı ışık değerlerinde de farenin takip edilmesini sağlamaktadır. Şekil 1'de filtre oluşturma süreci ve Şekil 2'de filtrenin görüntü üzerine uygulanma aşamaları sunulmuştur.



(a) (b) (c)

Şekil 1. a) Menü; b) kamera seçimi; c) filtre aracı



(a) (b) (c) (d)

Şekil 2. a) Deney görüntüsü; b) BGR2Gray (Gri Tonlama) Filtre; c) Median (Ortanca) Filtre; d)Treshold (Eşik) Filtre

Mevcut filtre aracı resim ve videolar ile çalışmaktadır. Çalışma kapsamında bu araçta filtrenin doğrudan kamera görüntüsünden oluşturulabileceği şekilde geliştirme yapılmıştır. İlk olarak filtre oluşturmaya başlamak için kameradan görüntü alınmasını sağlayan menü eklenmiş (Şekil 1-a), daha sonra bu menüye tıklandığında bilgisayarda o an aktif olarak kullanılan kameraların listelendiği ve kullanıcının bunlardan birini seçebildiği bir pencere eklenmiştir (Şekil 1-b). Son olarak kullanıcının seçtiği kameradan anlık bir kare görüntü alınmakta ve filtre oluşturma ekranına getirilmektedir (Şekil 1-c). Böyle kullanıcı deney düzeneğinin canlı olarak görüntüsünü alarak o görüntüye uygun filtre oluşturabilmektedir.

Kameradan görüntü almak için görüntü işleme çalışmalarında en çok kullanılan kütüphane olan OpenCV kullanılmıştır [9]. Kameradan alınan deney düzeneği görüntüsünde farenin filtre yardımı ile net bir şekilde belirlenebilmesinin ardından takip işlemine geçilmiştir.

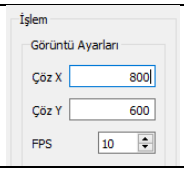
2.2. Takip Aracı

UMATracker halihazırda sadece video görüntüsünden noktasal olarak fare takibi yapabilmektedir. Canlı olarak bir kameradan takip yapabilme, takip işlemini belirli alanlara göre yapabilme, takip işlemini video olarak kaydetme, takip sürelerini hesaplama ya da belirli bir süreye göre takip yapma özelliklerine ise sahip değildir. Bu çalışma kapsamında bu geliştirmeler yapılmıştır.

Takip aracında, filtreleme aracındaki kamera ile ilgili geliştirmelerin ardından bu aracında öncelikle kameradan görüntü alması sağlanmıştır. Görüntü alma işlemi filtre aracı ile aynı şekilde yapılmıştır. Kullanıcıyı "Kamerayı Aç" menüsünde ya da arayüzdeki düğmeye tıkladığında kamera seçme seçeneği çıkmakta ve kullanıcı kamerayı seçtikten sonra arayüzde kameradan alınan görüntü canlı olarak kullanıcıya sunulmaktadır.

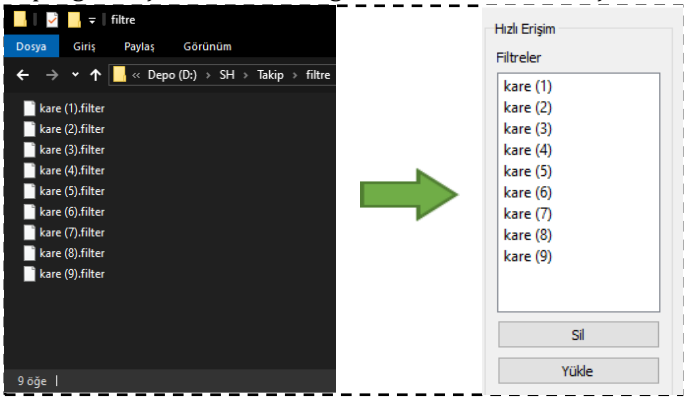
Kamera görüntüsü alınarak yapılan canlı analizlerde en büyük sorunlardan bir tanesi performans sorunlarıdır. Analiz yapılan bilgisayarın işlem gücünün yetersiz olması durumlarında takip işleminin canlı olarak gerçekleştirilmesi mümkün olmaz. Bu gibi durumlara çözüm olarak farklı işlem gücüne sahip bilgisayarlarda yazılımın sorunsuz şekilde takip işlemini gerçekleştirebilmesi için görüntünün çözünürlüğünün ve fps (frame per second – saniye başına görüntü sayısı) değerinin düşürülmesi gerekir. Bu yüzden geliştirilen araca kameradan alınan görüntünün çözünürlüğünü ve fps değerini belirleyebilme seçenekleri de eklenmiştir. Kameradan görüntü alma, çözünürlüğü belirleme ve fps değerini belirleme kodları ve menüleri Tablo 1'de sunulmuştur.

Tablo 1. Kamera çözünürlük ve fps ayarları menüsü ile kodları.




```
self.cap = cv2.VideoCapture(kamera)
self.cap.set(cv2.CAP_PROP_FRAME_WIDTH, cozX)
self.cap.set(cv2.CAP_PROP_FRAME_HEIGHT, cozY)
self.cap.set(cv2.CAP_PROP_FPS, FPS)
```

Kamera görüntüsü açıldıktan sonra kullanıcının daha önce hazırlayarak kaydettiği filtre dosyasını açması gerekmektedir. Bunun için araştırmacılara kolaylık olması açısından programın ana klasörü altından bulunan "filtre" klasörünün içinde bulunan kayıtlı filtre dosyaları kullanıcıya liste olarak sunulmaktadır. Şekil 3'te klasördeki filtre dosyaları ve program için liste halinde gösterilmeleri sunulmuştur.

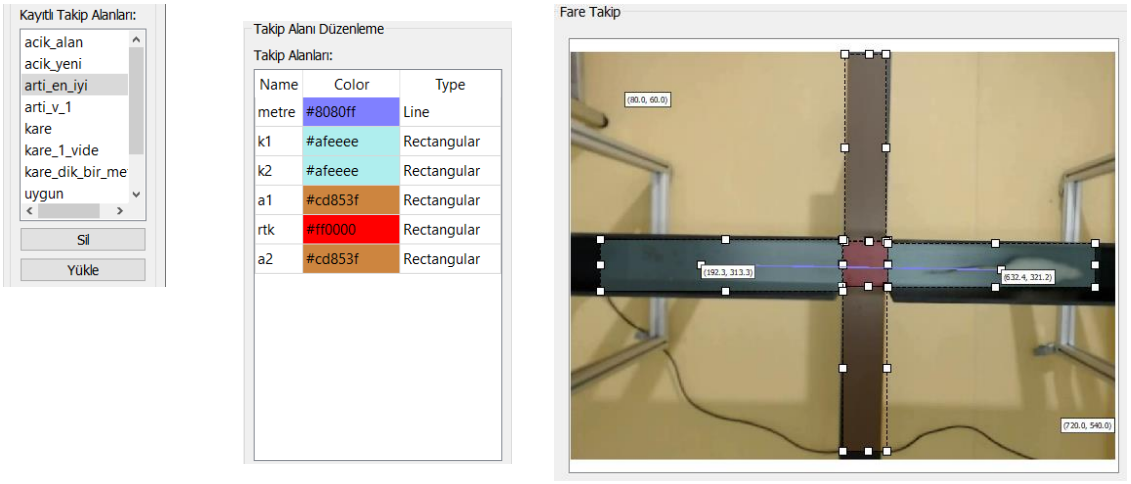


Şekil 3. Filtre kayıtları ve listelenmesi

Kullanıcı kamera görüntüsünü açıp filtreyi seçtikten sonra farenin takibi başlamaktadır. Farenin takip edildiği üzerinde yer alan işaret ile gösterilmektedir. Takip aracı ile 4 farklı takip algoritması kullanılabilmektedir. Bunlar: RMOT [10], Optical Flow DualTVL1 [11], K-means [12]ve GroupTracker [13]. Şekil 4'te GroupTracker algoritması ile bir farenin takip edildiği ekran görüntüsü yer almaktadır. Ayrıca aynı anda birden fazla fare de takip edilebilmekte ve görüntü üzerinde gerçek zamanlı olarak numaralandırılabilmektedir.

Şekil 4. Kameradan alınan görüntüden anlık olarak farenin takibi ve 0 olarak numaralandırılması

Fare takibinin başarıyla gerçekleştirilmesinden sonra farenin davranışlarınız analiz etmek için deney sahası üzerinde takip alanları oluşturulmalıdır. Bunları oluşturabilmek için dikdörtgen ve elips çizme araçları geliştirilmiştir. Yazılımın arayüzü geliştirilirken Qt grafiksel kullanıcı arayüzü geliştirme aracı, takip alanları çizimi içinde Qt aracının QShape bileşeni kullanılmıştır [14]. Kullanıcı arayüzde yer alan "Ekle" düğmesine tıklayarak takip ekranına bir dikdörtgen şekil oluşturmaktadır. Takip alanının üzerinde çıkan ufak beyaz kareler sürüklenip bırakılarak takip alanının boyutu ayarlanabilmektedir. "Takip Alanı Düzenleme" menüsünden takip alanının tipini değiştirerek yuvarlak ya da çizgiye dönüştürebilmektedir. Ayrıca takip alanıza bir isim ve renk verebilmektedir. Buna ek olarak çalışma kapsamında deney düzenekleri için hazırlanmış alanları kaydetme seçeneği eklenmiştir. Böylece araştırmacı her deney yaptığında tekrar tekrar bu alanları oluşturmak zorunda kalmamaktadır. Ayrıca bu alanları kolayca seçebilmesi için daha önce kaydedilmiş alanlar kullanıcı arayüzünde liste halinde kullanıcıya sunulmaktadır. Şekil 5a'da kayıtlı takip alanları, Şekil 5b'de takip alanlarının özellikleri ve Şekil 5c'de ise görüntü üzerinde takip alanlarının nasıl görüntülendiği sunulmuştur.



(a) (b) (c)

Şekil 5. a) Kayıtlı takip alanları; b) takip alanı düzenleme; c) takip alanlarının görünümü

Araştırmacıların analiz etmek istediği değerler arasında farenin toplam kat ettiği mesafe ve ortalama hızı yer almaktadır. Ancak mevcut araç ile bu yapılamamaktadır. Çalışmada bu analizleri yapabilmek için araştırmacıdan kamera görüntüsü üzerinde gerçek bir metreyi ifade eden çizgi tipinde bir takip alanı oluşturması ve adını da metre yapması istenmektedir. Analiz aşamasında bu çizilen çizginin öncelikle piksel cinsinden uzunluğu hesaplanmakta ve böylece Pisagor bağıntısından 1 pikselin kaç cm uzunluğa karşılık geldiği bulunmaktadır. Aşağıda bu işlemin denklemi (1) ve kaynak kodu yer almaktadır.

mesafeHesapla Bağıntısı:

$$mesafe = karekök\ (|x2 - x1|2 + |y2 - y1|2) \quad (1)$$

x1, y1: çizginin başlangıç koordinatları
x2, y2: çizginin bitiş koordinatları

```
metrePikselMesafe = self.anaPencere.mesafeHesapla(
        float(self.anaPencere.metre[0][0]),
        float(self.anaPencere.metre[0][1]),
        float(self.anaPencere.metre[1][0]),
        float(self.anaPencere.metre[1][1]),
    )
```

Bir metrenin piksel cinsinden karşılığı bulunduktan sonra farenin toplam kat ettiği mesafe hesaplanmaktadır. Bunun için de metre için olduğu gibi farenin bir önceki koordinatı ile bir sonraki koordinatı arasındaki mesafe Pisagor bağıntısı ile hesaplanmaktadır. Kayıt süresince de bu piksel cinsinden değerler kümülatif olarak toplanmaktadır. Son olarak kayıt sona erdiğinde toplam kat edilen mesafe metrenin piksel cinsinden değerine bölünüp 100 ile çarparak farenin toplam olarak kaç cm yol kat ettiği hesaplanmaktadır. Farenin ortalama hızı ise toplam kat edilen mesafenin saniye cinsinden belirlenen kayıt süresine bölünmesi ile bulunmaktadır. Aşağıda bu işlemim kaynak kodu (2) sunulmuştur.

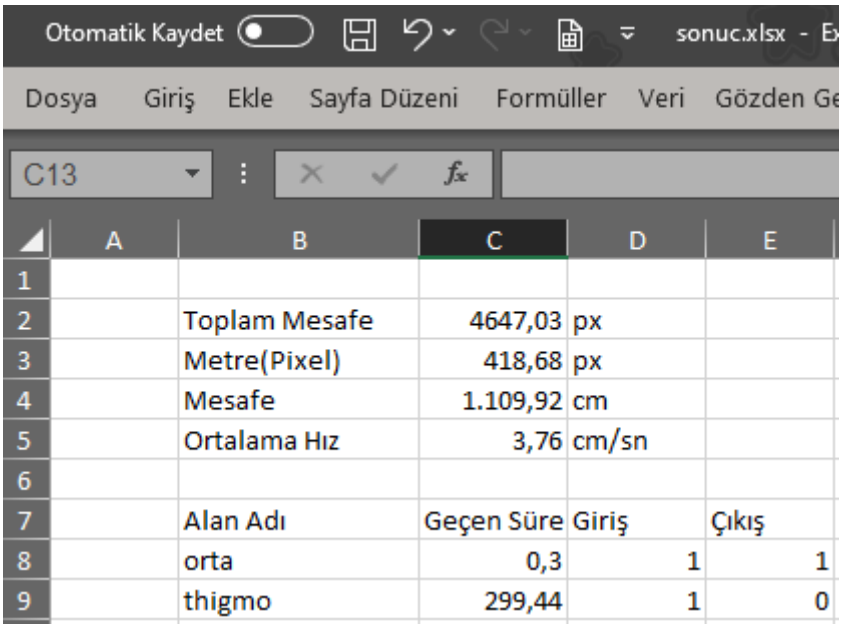
```
hız = toplamMesafe / kayitSuresi
```

$$toplamKatEdilenMesafe = (toplamPikselMesafe\ /\ metrePikselMesafe) * 100 \quad (2)$$

Araştırmacı fare, alanlar ve metrik ölçüm için 1 metrelik çizgiyi belirledikten sonra deney için hazır duruma gelmektedir. Bundan sonra deneye başlamak için arayüzde bir kayıt numarası ve bu kayıt ile ilgili girmek istiyorsa bir açıklama girebilmektedir. Daha sonra deneyin süreceği süreyi saniye cinsinden belirledikten sonra "Kaydı Başlat" tuşuna tıkladığında gerçek zamanlı olarak kayıt ve analiz işlemleri başlamaktadır. Kayıt işlemine başlanıldığında ilk olarak ilk kameradan alınan görüntü çerçevesi resim olarak kaydedilmektedir. Kayıt süresince farenin koordinatları sürekli olarak kayıt altına alınmakta ve anlık olarak her video çerçevesinde kat ettiği mesafe hesaplanmaktadır. Bununla birlikte farenin hangi takip alanı içinde yer aldığı bilgisi de kaydedilmektedir.

Kayıt işlemi tamamlandığında anlık olarak yapılan analizlerin yanında, toplam kat edilen mesafe, ortalama hız değerleri hesaplanmaktadır. Ayrıca farenin takip alanlarında bulunma durumları incelenerek, belirlenen takip

alanlarında ne kadar süre bulunduğu, kaç defa o takip alanlarına girip çıktığı hesaplanmaktadır. Çıktılar ise bir Excel dosyası olarak Şekil 6'da sunulduğu şekilde araştırmacı için kaydedilmektedir.



	A	B	C	D	E
1					
2		Toplam Mesafe	4647,03	px	
3		Metre(Pixel)	418,68	px	
4		Mesafe	1.109,92	cm	
5		Ortalama Hız	3,76	cm/sn	
6					
7		Alan Adı	Geçen Süre	Giriş	Çıkış
8		orta	0,3	1	1
9		thigmo	299,44	1	0

Şekil 6. Excel çıktısı

Tüm bu analizler başta belirlenen kayıt numarası adında açılan klasör içerisinde kayit.csv isimli bir dosyaya kaydedilmektedir. CSV dosyaları bir ayraç işareti ile ayrılmış verilerin satır satır saklamak için kullanılan bir metin dosya biçimidir [15]. Tablo 2'de CSV dosyasının içeriği ve alanların açıklamaları sunulmuştur.

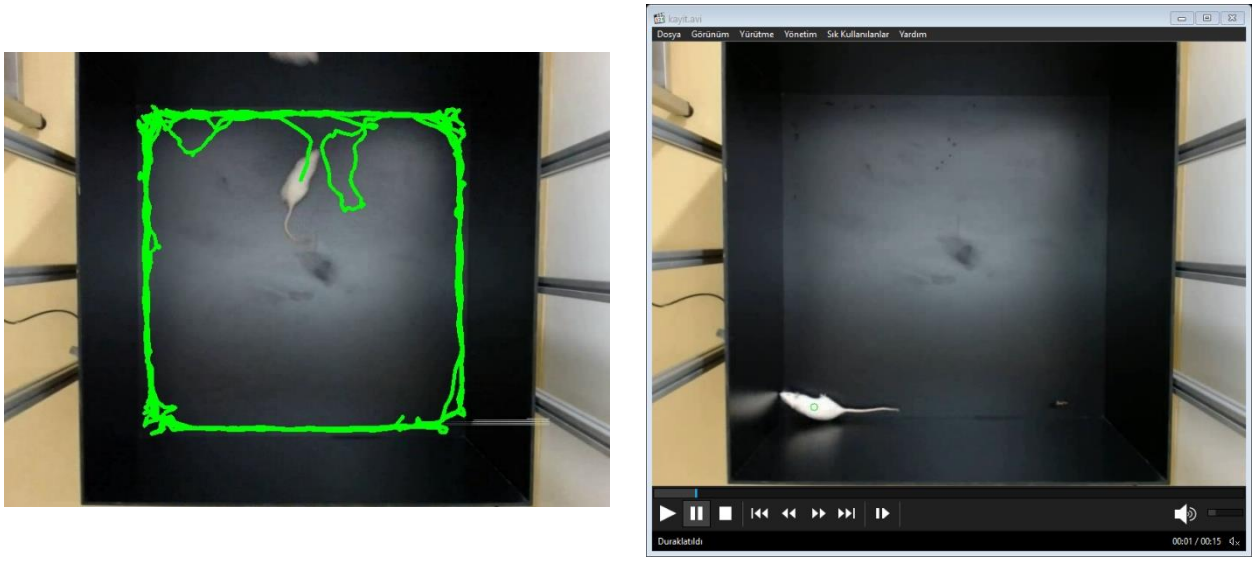
Tablo 2. CSV dosyası ve içeriği.

```
x;y;ic;dis;sure;mesafe
422.15833333333336;77.67916666666667;1;0;1.325611
421.9298245614035;79.6774628879892;1;0;1.376595;2.0113189819123045
415.12200137080197;84.33036326250857;1;0;1.427593;8.24596498232285
391.0067491563555;85.8138357705287;1;0;1.478581;24.160837734822366
```

x, y, ic, dis, sure ve mesafe sütun başlıklarını ifade etmektedir. Alttaki noktalı virügl ile ayrılmış değerler ise her kaydın ilgili sütuna gelen değerini ifade etmektedir. Sütunların karşılıkları ise aşağıdaki gibidir.

x: farenin yatay eksende pozisyonu
y: farenin dikey eksende pozsiyonu
ic: ic isimli takip alanında farenin bulunma durumu
dis: dis isimli takip alanında farenin bulunma durumu
sure: bu veri kaydedildiği anda deneyin başlangıcından itibaren geçen süre
mesafe: farenin bir önceki durumu ile bu durumu arasında kat ettiği mesafe

Son olarak kaydın başlangıcında kaydedilen resim üzerinde farenin geçtiği tüm noktalar OpenCV kütüphanesinin çizgi çizme aracı kullanılarak çizilen çizgiler ile birleştirilerek farenin yol haritası görsel olarak analiz edilmektedir. Görsel analizin sonuçları Şekil 7a'da sunulmuştur. Bunun yanında tüm analiz işlemi video dosyası olarak ta kayıt altına alınmaktadır. Video kaydı da yine OpenCV kullanılarak kullanıcının belirlemiş olduğu çözünürlük ve fps değerlerine göre Xvid biçiminde sıkıştırılmış olarak kameradan alınan anlık görüntülerin üzerine farenin bulunduğu noktaya yeşil bir çember çizilerek yapılmaktadır. Xvid sıkıştırma biçimi kullanılmasının sebebi, açık kaynak kodlu özgür bir biçim olması ve tüm platformlarda kullanılabilmesidir [20]. Ayrıca sıkıştırma oranı ve kalite bakımından çalışma sürecinde yeterli sonuçlar üretmiştir. Video kaydının örnek ekran görüntüsü Şekil 7b'de sunulmuştur.



(a) (b)

Şekil 7. Farenin geçtiği yolların çizimi

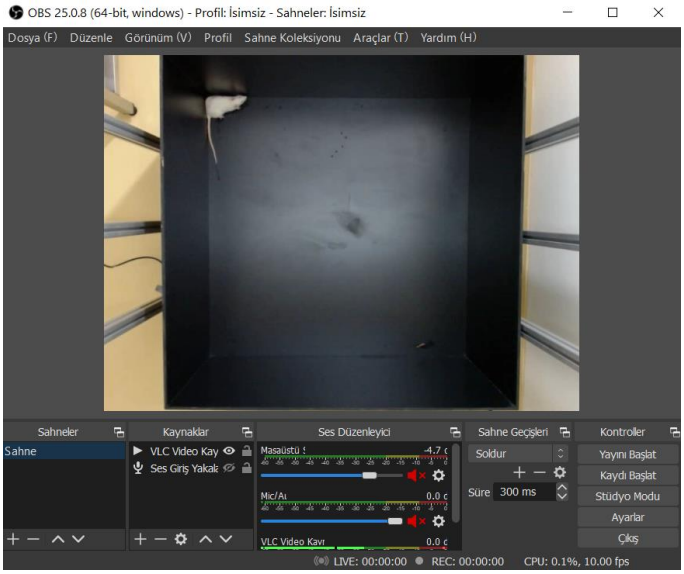
3. Bulgular

Geliştirilen aracın yazılım geliştirme süreci ve sonrasında istenilen işlemleri doğru bir şekilde yaptığının denetlenmesi için hem daha önce analizi yapılmış çalışmalarla karşılaştırması yapılmış hem de gerçek zamanlı uygulamalar yapılmıştır.

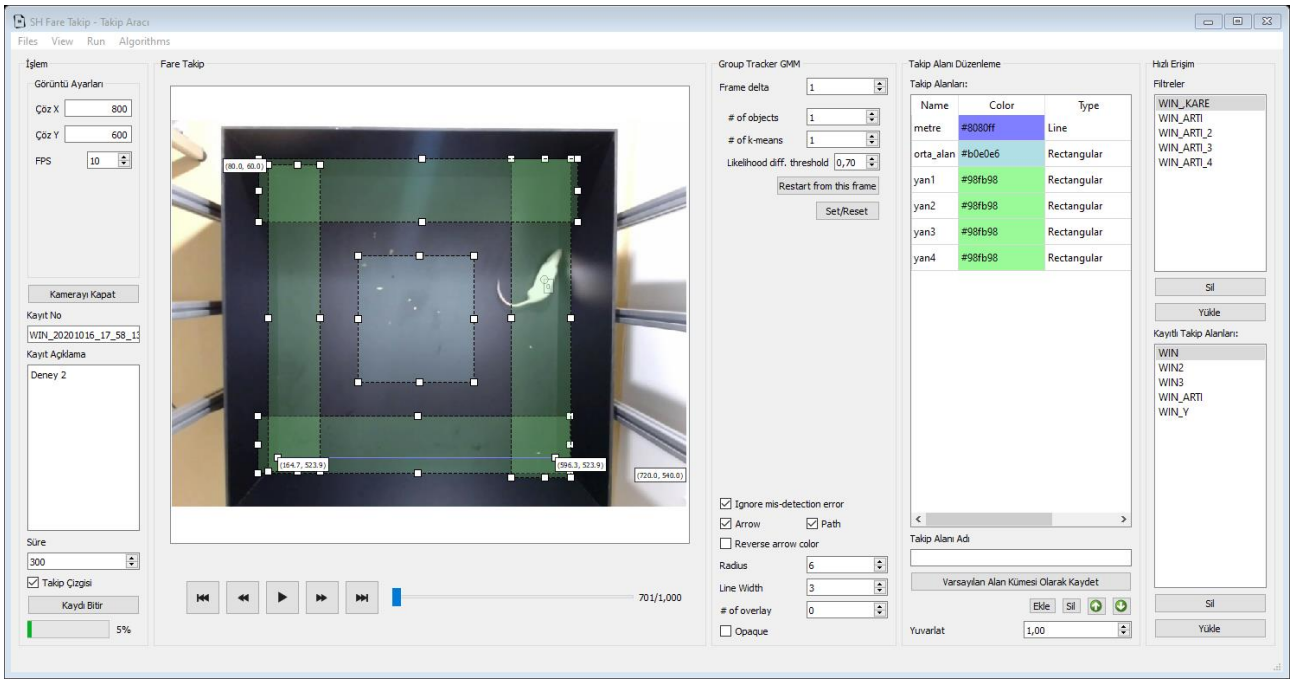
Yazılımın geliştirilmesinin ardından ilk aşamada gerçek zamanlı olarak çalışmak çeşitli sorunlar ortaya çıkarmıştır. Bunların ilki salgın nedeniyle farklı araştırmacıların aynı ortamda uzun süre çalışmasının zorluğudur. İkinci zorluk gerçek bir canlı ile çalışmalar yapılırken sürekli olarak hataların ve eksiklerin tespit edilerek giderilirken ayrıca deney hayanı ile ilgilenilmesidir. Üçüncü ve en büyük zorluk ise yazılım ile analiz edilen tüm görüntülerin doğruluğunun tespit edilebilmesi için ayrıca gerçek olarak insan gücü ile analiz edilmesidir. Bu vb. sorunlardan dolayı çalışmada ilk olarak daha önce analizi yapılmış görüntüler ile çalışılmasına karar verilmiştir.

Geliştirilen yazılım kayıt edilmiş videolar ile değil de sadece kameradan alınan görüntüler ile çalışacak şekilde geliştirilmiştir. Ancak sanal kamera yöntemi ile bu kısıt aşılabilmektedir. Bu yöntemde bir video bir sanal kamera yazılımı ile sanki bilgisayara bağlanan bir kameradan görüntü alınıyormuşçasına gösterilebilmektedir.

Sanal kamera oluşturmak için OBS Studio yazılımı ve bu yazılımın VirtualCam eklentisi kullanılmıştır [16] [17]. Araç yardımı ile fare deney videoları kamera görüntüsü olarak sunulmuştur. Şekil 8'de bir videonun OBS yazılımı ile sanal kamera olarak yayınlanmasını ve Şekil 9'da geliştirilen takip yazılımında gösterilmesi sunulmuştur.

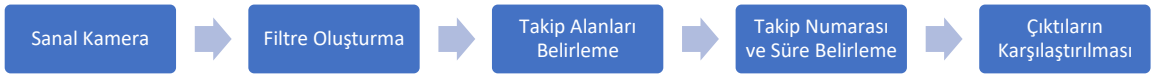


Şekil 8. OBS yazılımı ile sanal kamera yayını



Şekil 9. Takip yazılımı

Geliştirilen takip aracı daha önce kare ve artı şeklindeki platformlarda kaydedilmiş fare hareket mesafesi, ortalama hızı ve belirlenen bölgelerde geçirdiği süre analizleri yapılmış 18 tane 5 dakikalık deney videosu ile karşılaştırma yapılarak test edilmiştir. Bu videolar bu alanda kabul görmüş Noldus yazılımı ile analiz edilmişlerdir [21] [22]. Test yapabilmek için öncelikle videolar için filtre aracı ile filtreler oluşturulmuştur. Daha sonra takip aracı ile kare ve artı platformlar için kayıtlara uygun takip alanları çizilmiştir. Son olarak videoların tamamı yazılım ile analiz edilmiştir. Tüm İşlemin akış şeması Şekil 10'da, sonuçların daha önce yapılan analizlerle karşılaştırılması Tablo 3'te sunulmuştur.



Şekil 10. Test süreci adımları

Tablo 3. Referans değerler ile takip çıktılarının karşılaştırılması

Kare Platform						Artı Platform							
Ort. Hız(cm/sn)		Orta Alan(sn)		Thigmo Zone(sn)		Ort. Hız(cm/sn)		Açık Kol.(sn)		Kap. Kol.(sn)		Ortak Alan(sn)	
Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY
5,48	5,47	0,3	0,3	299,44	299,5	2,24	2,22	1,84	1,83	297,52	296,7	1,68	1,67
8,97	8,88	4,88	4,87	281,44	281,3	2,68	2,58	0	0	301,04	300,81	0	0
10,3	10,43	9,84	9,76	270,4	270,42	6,15	6,2	3,36	3,42	275,12	274,85	22,56	22,77
7,786	7,73	2,08	2,15	283,52	283,53	3,68	3,57	0,24	0,23	291,52	292,02	9,28	9,45
8,94	8,74	1,12	1,13	291,6	291,63	3,78	3,85	0	0	284,16	283,96	16,88	17,06
10,1	9,85	0,28	0,29	297,12	297,13	5,37	5,35	0	0	286,16	288,01	14,88	14,85
7,01	7,44	1,6	1,61	293,6	293,52	4,03	4,03	0	0	301,04	300,89	0	0
9,18	8,89	5,76	5,74	285,36	284,32	3,62	3,65	0	0	296,8	296,52	4,24	4,22
4,46	4,66	7,76	7,68	286,32	286,3	3,71	3,59	32,96	33,1	227,92	228,04	40,16	40,25
0,184		**0,031**		**0,156**		**0,057**		**0,024**		**0,49**		**0,078**	

Tablo 3'te Ref. ile ifade edilen alanlar daha önce Noldus yazılımı ile analiz edilmiş değerleri GY (Geliştirilen Yazılım) ise aynı görüntü kayıtlarının geliştirilen takip yazılımı ile yapılmış analiz sonuçlarını göstermektedir. Tablo 3'te sonuçlar incelendiğinde referans analiz değerleri ile analiz sonuçları arasında kare platformda ortalama hızlarda 0,184 cm/sn, orta alan geçirme sürelerinde ortalama 0,031sn ve thigmo zone(kare platformun duvarlarına yakın olan alanlar) sürelerinde ise ortalama 0,156 sn fark ortaya çıkmıştır. Artı platformda ortala hız değerleri arasında ortalama 0,057 cm/sn, açık kol geçirme sürelerinde ortalama 0,024sn, kapalı kol geçirme sürelerinde ortalama 0,49sn ve ortak alan geçirme sürelerinde ise ortalama 0,078sn fark ortaya çıkmıştır.

Yazılımın, analizi daha önce yapılmış olan videolar ile test edilmesinin ardından gerçek ortamda test işlemi gerçekleştirilmiştir. Test işleminde hem kare hem artı platformda ikişer kere olmak üzere 5 dakikalık analizler yapılmıştır. Şekil 11'de test işleminden görüntüler sunulmuştur.

Şekil 11. Test platformu ve test işlemleri

Bu süreçte alınan görüntüler ayrıca video olarak kaydedilmiş ve Noldus programı ile de analiz edilmiştir. Geliştirilen yazılım ve Noldus yazılımının ürettiği sonuçlar karşılaştırılmış ve Tablo 4'te sunulmuştur.

Tablo 4. Referans değerler ile takip çıktılarının karşılaştırılması.

Kare Platform						**Artı Platform**							
Ort. Hız(cm/sn)		**Orta Alan(sn)**		**Thigmo Zone(sn)**		**Ort. Hız(cm/sn)**		**Açık Kol.(sn)**		**Kap. Kol.(sn)**		**Ortak Alan(sn)**	
Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY	Ref.	GY
6,54	6,48	1,2	1,23	295,12	295,5	1,98	1,96	0	0	298,32	298,72	1,68	1,67
3,97	3,88	3,38	3,33	291,41	292,2	2,47	2,44	3,35	3,39	291,12	291,81	5,21	5,28
0,075		**0,04**		**0,58**		**0,025**		**0,02**		**0,54**		**0,04**	

Tablo 4'te sonuçlar incelendiğinde referans analiz sonuçları ile yazılımın analiz sonuçları arasında kare platformda ortalama hızlarda 0,075 cm/sn, orta alan geçirme sürelerinde ortalama 0,04sn ve thigmo zone(kare platformun duvarlarına yakın olan alanlar) sürelerinde ise ortalama 0,58 sn fark ortaya çıkmıştır. Artı platformda ortala hız değerleri arasında ortalama 0,025 cm/sn, açık kol geçirme sürelerinde ortalama 0,002sn, kapalı kol geçirme sürelerinde ortalama 0,54sn ve ortak alan geçirme sürelerinde ise ortalama 0,04sn fark ortaya çıkmıştır.

Yazılım hem daha önceden kaydedilmiş ve analiz edilmiş videolar hem de 2 artı platform ve 2 kare platform ile gerçek zamanlı olarak test edilmiştir. Bu testler sonucunda elde edilen değerler yazılımın başarılı bir şekilde deney hayvanını takip edebildiğini ve sonuçları analiz edebildiğini göstermiştir. Sonuçlar incelendiğinde ortalama değerlerde ortaya ufak ta olsa farklılıklar çıkmıştır. Bu farklılıklar farenin takip edilirken orta noktasına göre takip edilmesinden kaynaklanmaktadır. Filtre oluşturulurken yapılacak ufak değer değişiklikleri farenin orta noktasının hesaplanmasında farklılıklara sebep olabilmektedir. Bunun sonucunda da değerlerde ufak farklar meydana gelebilmektedir. Ancak bu ufak farklılıklar analizlerde anlamlı farklılıklar oluşturmamıştır.

4. Tartışma ve Sonuç

Bu çalışma sonucunda fareler ya da benzeri deney hayvanları ile çeşitli platformlarda davranış analizleri yapılabilecek bir araç geliştirilmiştir. Ortaya çıkarılan araç ile gerçek zamanlı olarak farenin pozisyonu, toplam kat ettiği mesafe, ortalama hızı ve araştırmacı tarafından deney platformu üzerinde belirlenen alanlarda ne kadar süre kaldığı ve bu alanlara giriş-çıkış frekansları analiz edilebilmektedir. Ayrıca farenin hangi rotayı izlediği görsel olarak kaydedilmektedir.

Geliştirilen aracın gerçek zamanlı analiz özelliği sayesinde araştırmacılar önce deney yaparak videoyu kaydetme ve daha sonra videoları tekrar analiz etmek yerine bir kerede tüm analiz işlemini gerçekleştirebilmektedir. Ayrıca filtrelerin ve takip alanlarının kaydedilerek kolay bir şekilde kayıttan yüklenebilmesi sayesinde zaman ve emek kazancı sağlanmaktadır. Son olarak aracın referans bir çizgi ile mesafe hesaplayabilme özelliği herhangi bir yükseklikten çekilen tüm görüntülerde metrik olarak doğru bir hesabı mümkün kılmıştır.

Geliştirilen yazılım üzerine yapılacak gelecek çalışmalarda ip kameralardan görüntü alarak uzaktan davranış analizi, Richardson vd. [18] çalışmalarında kullanılan yönteme benzer filtre aracına alternatif olarak derin öğrenme yöntemleri ile birden fazla fareyi tanıma, numaralandırma [19] ve takip yöntemi, yazılımı tek bir kişinin bilgisayarda ayar yapmadan kullanabilmesi için sesli komutlarla kontrol özellikleri eklenmesi düşünülmektedir.

Teşekkür

Bu çalışma, Balıkesir Üniversitesi tarafından desteklenmiştir, BAP Proje No: 2019/097.

Kaynakça

[1] Catarinucci, L., Colella, R., Mainetti, L., Patrono, L., Pieretti, S., Secco, A. ve Sergi, I. 2014. An animal tracking system for behavior analysis using radio frequency identification. Lab Animal, 43(2014), 321-327, doi:https://doi.org/10.1038/laban.547

[2] Woehr, D., Arciniega, L. ve Poling , T. 2013. Exploring the Effects of Value Diversity on Team Effectiveness, Journal of Business and Psychology, 28(2013), 107-121.

[3] Spink, A., Tegelenbosch, R., Buma, M. ve Noldus, L. 2001. The EthoVision video tracking system—A tool for behavioral phenotyping of transgenic mice, Physiology & Behavior, 73(2001), 731-744, doi:https://doi.org/10.1016/S0031-9384(01)00530-3.

[4] Aguiar, P., Mendonça, L. ve Galhardo, V. 2007. OpenControl: A free opensource software for video tracking and automated control of behavioral mazes, Journal of Neuroscience Methods, 166(2007), 66-72.

[5] EthoVision XT. https://www.noldus.com/ethovision-xt (Erişim Tarihi: 14.09.2020).

[6] VideoMot2. https://www.tse-systems.com/product-details/videomot/ (Erişim Tarihi: 14.09.2020).

[7] Yamanaka, O. ve Takeuchi, R. 2018. UMATracker: an intuitive image-based tracking platform, Journal of Experimental Biology, 221(2018).

[8] Rodriguez, A., Zhang, H., Klaminder, J., Brodin, T., Andersson, P., & Andersson, M. 2018. ToxTrac: a fast and robust software for tracking organisms, Methods in Ecology and Evolution, 9(2018), 460-464.

[9] Open Computer Vision: Open CV. https://opencv.org/ (Erişim Tarihi: 08.09.2020).

[10] Yoon, J., Yang, M.-H., Lim, J., & Yoon, K.-J. 2015. Bayesian Multi-object Tracking Using Motion Context from Multiple Objects, IEEE Winter Conference on Applications of Computer Vision, 6-9 Ocak 2015, Waikoloa, 33-40.

[11] Zach, C., Pock, T., & Bischof, H. 2007. A Duality Based Approach for Realtime TV-L1 Optical Flow, Joint Pattern Recognition Symposium, 12-14 Kasım 2007, Berlin, 214-223.

[12] Llyod, S. 1982. Least squares quantization in PCM, IEEE Transactions on Information Theory, 28(1982), 129-137.

[13] Fukunaga, T., Kubota, S., Oda, S., & Iwasaki, W. 2015. GroupTracker: Video tracking system for multiple animals under severe occlusion, Computational Biology and Chemistry, 57(2015), 39-45.

[14] Qt | Cross-platform software development for embedded & desktop. https://www.qt.io/ (Erişim Tarihi: 17.09.2020).

[15] Y. Shafranovich. 2020. Common Format and MIME Type for Comma-Separated Values (CSV) Files. https://tools.ietf.org/html/rfc4180 (Erişim Tarihi: 09.09.2020).

[16] Open Broadcaster Software. https://obsproject.com/tr (Erişim Tarihi: 09.09.2020).

[17] OBS Virtual Cam. https://github.com/CatxFish/obs-virtual-cam (Erişim Tarihi: 22.09.2020).

[18] Peixoto H.M., T. R.-A. 2019. Mice Tracking Using The YOLO Algorithm, PeerJ Preprints, 7(2019).

[19] Kılıç, E., Öztürk, S. 2020. İnsansız Hava Aracı Görüntülerinde Evrişimli Sinir Ağı Kullanarak Araç Sayımı için Yeni Bir Haritalama Yöntemi, Erciyes Üniversitesi Fen Bilimleri Enstitüsü Fen Bilimleri Dergisi, 36(2020), 119-127.

[20] Xvid. https://www.xvid.com/ (Erişim Tarihi: 18.09.2021).

[21] Noldus, L.P.J.J., Spink, A.J. & Tegelenbosch, R.A.J. EthoVision: A versatile video tracking system for automation of behavioral experiments. Behavior Research Methods, Instruments, & Computers 33, (2001), 398–414. https://doi.org/10.3758/BF03195394

[22] Noldus EthoVision. https://www.noldus.com/ethovision-xt/ (Erişim Tarihi: 18.09.2021).