ARTIRILMIŞ GERÇEKLİK İLE MOBİL UYGULAMA GELİŞTİRME: YENİLİKÇİ BİR YAKLAŞIM

1. Esma Gelmez
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,TÜRKİYE
221307099

2. Selin Avcı
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,TÜRKİYE
211307026

3. Helen Hacer Uzunçayır
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,TÜRKİYE
211307061

Özet—Bu makale, artırılmış gerçeklik (AR) teknolojisinin mobil uygulama geliştirmeye entegre edilmesini ele almaktadır. Yazarlar,projenin hedeflerini,kullanılan yöntemleri ve elde edilen sonuçları kapsayan kapsamlı bir açıklama sunmaktadır.

Anahtar Kelimeler—Artırılmış Gerçeklik,Mobil Uygulama,Geliştirme,Teknoloji

I. GİRİŞ

Bu Raporda, IEEE'nin belirlediği standartlara uygun olarak hazırlanmış bir Artırılmış Gerçeklik(AR) projenin detaylı açıklamasını içermektedir. Bu Rapor, artırılmış gerçeklik teknolojisinin mobil uygulama geliştirmeye başarıyla entegre edilmesini ele almaktadır.

II. Arttırılmış Gerçeklik ve ARKit

Bu raporda,öncelikle Arttırılmış Gerçeklik (AR) kavramı ile ARKit'in önemi ve temel özellikleri ele alalım.

A. Arttırılmış Gerçeklik Nedir?

Arttırılmış gerçeklik, cihazın kamerasıyla dijital dünyayı gerçek dünyaya entegre etmek için kullanılır. Bu kullanıcının dijital dünya deneyiminin kalitesini arttırır ve daha ilginç kılar. Arttırılmış gerçeklik, gerçek dünyada erişemediğimiz çoğu unsuru kullanıcının cihazı ve yazılım aracılığıyla gerçek dünyaya getirir

B. ARKit nedir?

Arkit, arttırılmış gerçeklik konusunu swift diline entegre etmek için apple'ın geliştirdiği bir kütüphanedir. Arkit, cihazın konumunu, hareketini ve çevresel bilgileri algılayabilen gelişmiş bir algılama sistemine sahiptir. Bu özellikler, kullanıcıların cihazlarını gerçek dünyada dolaşırken sanal nesneleri görüntülemelerini ve etkileşimde bulunmalarını sağlar.arkit aracılığıyla, arttırılmış gerçeklik uygulamaların kodlanması çok daha basitleşmiş, ulaşılabilirliği artmıştır.

C. ARKit Kütüphanesi ve Temel Özellikleri

Arkit kütüphanesi 2 temel özelliğin birbiriyle uyum içinde çalışmasıyla çalışır. Bunlar, konum algılama ve entegre etmektir.

D. Konum Algılama (Tracking)

Bu özellik, cihazınızın konumunu anlamak için kullanılır. Arkit kütüphanesinin kameranızdan etrafınıza herhangi bir nesne entegre edebilmesi için öncellikle konumunuzu bilmesi gerekir. Konum algılama özelliği etrafınızdaki objeleri de algılar ve buna göre uygulamanızı nesneleri entegre etmesi gereken yerlere yönlendirir. Arkit, cihazın dünyadaki konumunu sürekli olarak izler ve hareket ettikçe bu konumu günceller. Kütüphanede, konumunuzu takip etmek için kullanılan bazı özellikler şöyledir:

- Arworldtrackingconfiguration: cihazın kamerasından görülen dünyayı algılar. Farklı yüzeyler veya gerçek hayatta bulunan objeler bu özelliği kullanarak algılanır.
- Arbodytrackingconfiguration: cihazınızın kamerasından etrafınızdaki ortamı değil, vücudunuzu

algılar. El kaldırma, yürüme veya zıplama gibi hareketler bu özellikle algılanır.

- Arfacetrackingconfiguration: cihazınızın kamerasının görüş alanında olan yüzleri algılamak için kullanılır. Yüzünüzle yaptığınız herhangi bir mimik veya ifadeyi algılama bu özellikle gerçekleşir.

E. Entegre etmek (Rendering)

Arkit kütüphanesi kodunuzda bulunan 3 boyutlu dijital nesneleri gerçek hayata entegre eder. Bu kategorinin altına, 3 boyutlu obje oluşturmak ve ona bir hareket katmak girebilir. Aynı zamanda entegre etme başlığının en önemli özelliklerinden biri ise ortama 3 boyutlu objenin gözükebilmesi için dijital dünyadan ışık katmaktır. Bu noktadaki her şey bilgisayarınızın arkit arayüzünde gerçekleşir, yani tamamen dijital dünyada gerçekleşir. Gerçek dünyaya bir obje entegre ederken uygulamanız, konum algılamanın verdiği sonuçlardan yararlanır.

III. ARDK(ARTIRILMIŞ GERÇEKLIK GELIŞTIRME KITI) KULLANIMI

Ardk, artırılmış gerçeklik projelerini geliştirmek için sunduğu araç seti ile, kullanıcıların projelerine entegre etmelerini kolaylaştırır, ayrıca kurulum adımları ve platform seçimi gibi önemli aşamalarda rehberlik sağlar.

A. ARDK Nedir?

ARDK, artırılmış gerçeklik projelerini geliştirmek için kullanılan bir araç setidir.

B. ARDK Kurulum Adımları

ARDK'yi projeye eklemek için şu adımları takip ettik:

1) ARDK Eklentisi Kurulumu:

a) ARDK eklentisini GitHub üzerinden veya tarball olarak indirdik.Unity projesinde Package Manager'ı açtık ve "Add package from git URL..." seçeneğiyle https://github.com/niantic-lightship/ardk-upm.git adresini ekledik.

2) Lightship Etkinleştirme :

a) Proje içinde Lightship'i etkinleştirmek için: Unity üst menüsünden "Lightship > XR Plug-in Management" seçeneğine gidip ilgili platformu seçtik. Android için "Niantic Lightship SDK + Google ARCore" veya iOS için "Niantic Lightship SDK + Apple ARKit" seçeneklerini işaretledik

C. API Anahtarı Ekleme

a) Lightship API'yi kullanmak için: lightship.dev adresinden giriş yaptık ve projeler bölümüne gittik.Var olan veya yeni bir proje seçtik.Projemizin Genel Bakış bölümünden API anahtarını kopyaladık.Unity projesinde "Lightship > Settings" menüsünden API anahtarını yapıştırdık.

D. Mobil Platform Seçimi

a) Projenin mobil platformunu belirlemek için: Unity "File > Build Settings" penceresini açtık.iOS veya Android'i seçtik ve platformu değiştirdik.İlgili platforma özgü ayarları gerçekleştirdik.

E. Gradle (Sadece Unity 2021 için Android)

a) Eğer Unity 2021 kullanıyorsak: Gradle 6.7.1'i indirip yükledik.

IV. PROJE GELIŞTIRME VE UYGULAMA AŞAMALARI

A. Proje Genel Bakışı

Proje, artırılmış gerçeklik ve XR teknolojilerini kullanarak interaktif bir oyun ortamı oluşturmayı amaçlamaktadır.

Projenin temel amacı, geleneksel kamera kullanımının ötesine geçerek video aracılığıyla artırılmış gerçeklik deneyimi sunmaktadır. Bu bağlamda, ana kamera yerine XR Origin kullanılarak video üzerinden oyun içeriği görüntülenmektedir.

Meshing nesnesi, XR Origin içine entegre edilerek projenin temel yapı taşlarından birini oluşturur. Meshing'in detaylı ayarları, Lightship'in önceden yapılandırılmış ayarları sayesinde kolayca gerçekleştirilebilmektedir.

Projenin önemli bir özelliği, telefona kurulum yapmadan önce video aracılığıyla artırılmış gerçekliği test edebilme imkanı sunmasıdır. Bu, çevredeki cisimlerin şekillerini tarayarak sınırları belirleyen bir sistem geliştirmeyi mümkün kılar.

GameboardNavigation nesnesi, projenin kullanıcı arayüzünü oluşturmak üzere tasarlanmıştır. Bu nesne, içine eklenen component'ler aracılığıyla mavi kare bloklarını yönetir ve Gameboard Manager'a entegre edilerek ana kameranın kontrolünü sağlar.
GameboardCharacter nesnesi, kullanıcının projedeki etkileşimini temsil eder. Bu karakter için oluşturulan C# dosyaları (Mesh_Character ve Click to Move), karakterin hareketlerini ve etkileşimlerini yönetir. Cube nesnesi üzerinden kullanıcı, gri karelerin üzerine çıkarak interaktif bir deneyim yaşar.

Projede yer alan zıplama hareketleri, kullanıcının karakteri ile etkileşime geçerek bulunduğu alandaki gri karelerin yerine geçmesini sağlar. Bu sayede, kullanıcıya dinamik ve eğlenceli bir oyun deneyimi sunulur.

Sonuç olarak, proje, video aracılığıyla artırılmış gerçeklik deneyimini başarıyla gerçekleştiren bir oyun ortamı sunmaktadır. Kullanıcının karakter üzerinden etkileşimde bulunarak oluşturduğu yolu takip etmesi, projenin temel dinamiklerini oluşturur.

B. Main Camera ve XR Origin Kurulumu

,Projenin başlangıcında, ana kamerayı kaldırıp XR Origin'i ekledik. Bu adım, oyunu videoda çalıştırmamıza olanak sağladı. İlgili ayarları şu şekilde gerçekleştirdik:

Ana kamerayı kaldırıp XR Origin'i ekledik.

[Görsel 1]: XR Origin kurulumunu gösteren bir ekran görüntüsü.

C. *Meshing Nesnesi ve Ayarları*

XR Origin'e eklediğimiz Meshing nesnesini ve ayarlarını burada tanımlıyoruz:

Meshing nesnesini XR Origin'in içine yerleştirdik.

[Görsel 2]: Meshing ayarlarının ekran görüntüsü.

D. *Lightship Ayarları ve Test*

Projenin mobil platformunu belirlemek adına şu adımları takip ettik:

a) Unity'de "File > Build Settings" penceresini açtık.
b) Mobil platformu seçerek (iOS veya Android), platformu değiştirdik.
c) İlgili platforma özgü ayarları gerçekleştirdik.
d) Lightship ayarlarını yaparak projeyi test ettik.

[Görsel 3]: Lightship ayarlarının ekran görüntüsü.

E. *GameboardNavigation Nesnesi Oluşturma ve Konfigürasyon*

Unity 2021 kullanıyorsak, şu adımları takip ederek GameboardNavigation nesnesini oluşturup konfigüre ettik:

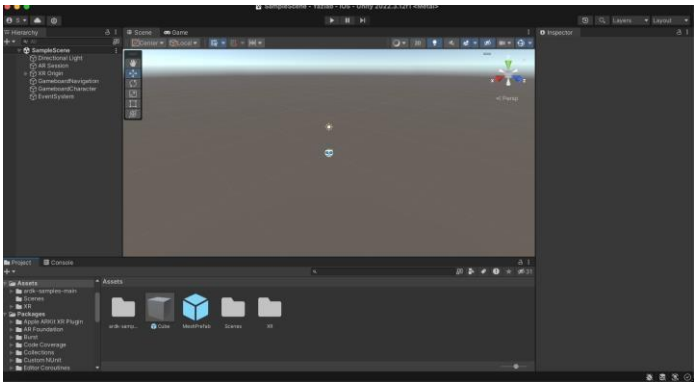
a) Gradle 6.7.1'i indirip yükledik.
b) GameboardNavigation nesnesini oluşturduk.
c) İçine gerekli component'leri ekleyip konfigüre ettik

[Görsel 4]: GameboardNavigation ayarlarının ekran görüntüsü

F. *GameboardCharacter Nesnesi ve C# Dosyalarının Yüklenmesi*

ARDK kullanarak projemize GameboardCharacter nesnesini ekledik ve bu adımları izledik:

a) GameboardCharacter nesnesini oluşturduk.
b) İlgili C# dosyalarını GameboardCharacter'e ekledik

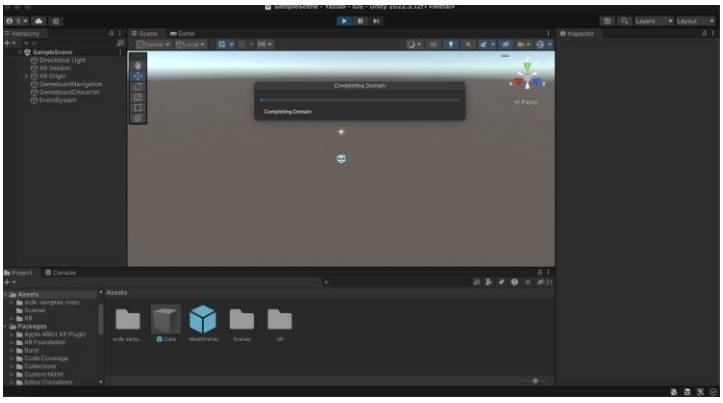


[Görsel 5]: GameboardCharacter ayarlarının ekran görüntüsü.

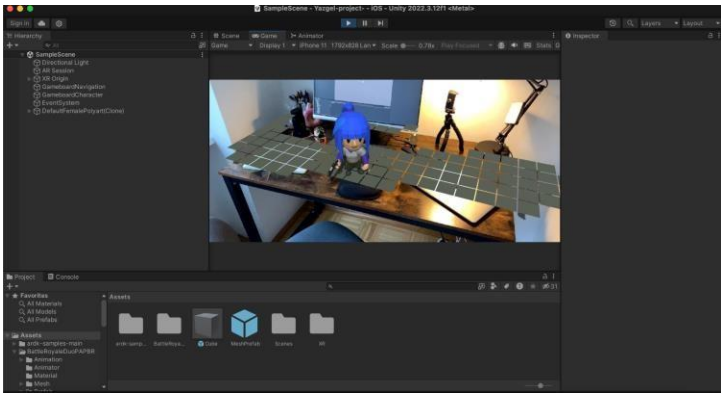
G. *Karakter Oluşturma ve Hareket*

ARDK kullanarak oluşturduğumuz Karakter nesnesi hakkında detaylar:

a) Karakter nesnesini oluşturduk.
b) Görünüm ve hareket özellikleri hakkında ayrıntılar.



[Görsel 6]: Karakter yüklenirken ekran görüntüsü.



[Görsel 7]: Karakterin ekran görüntüsü.

H. *Zıplama Hareketleri ve Gri Kare Blokları*

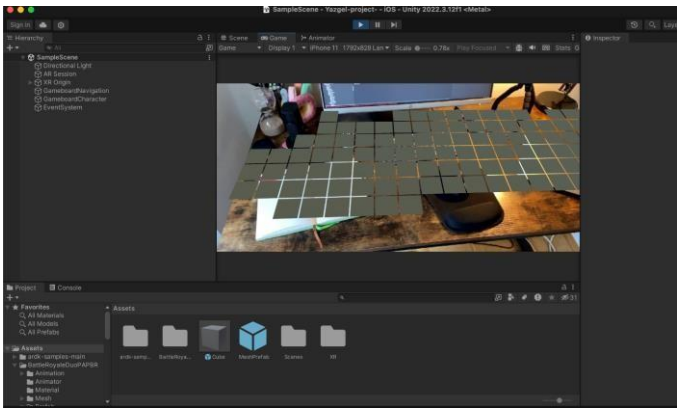
ARDK kullanılarak zıplama hareketlerinin implementasyonu ve gri kare bloklarının oluşturulması adımları şu şekildedir:

a) Zıplama Hareketleri Implementasyonu:

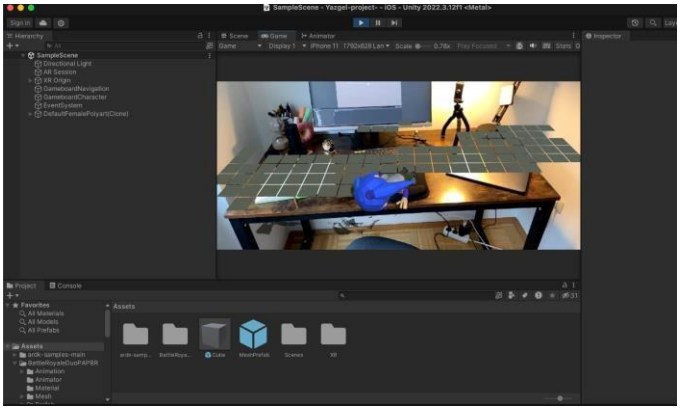
1. Zıplama hareketlerini kontrol etmek üzere gerekli kodlamayı gerçekleştirdik.
2. Oyuncunun dokunmatik ekran üzerindeki hareketlerini algılayarak zıplama işlevselliğini sağladık.

b) İlgili C# dosyalarını GameboardCharacter'e ekledik Gri Kare Blokları Oluşturma:

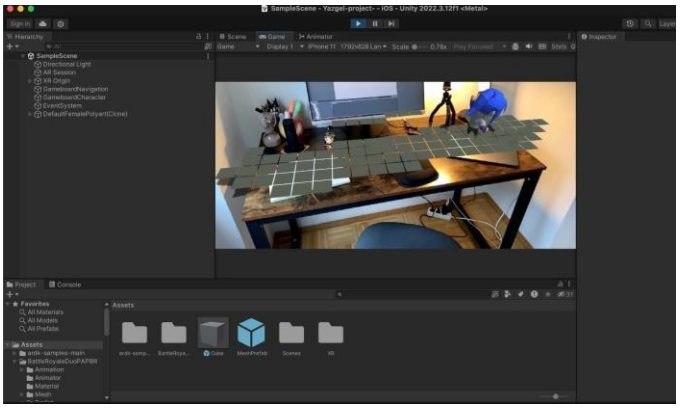
1. ARDK'nın sağladığı özellikleri kullanarak gri kare bloklarını oluşturduk.
2. Blokların konumları ve yerleşimi için gerekli ayarlamaları yaptık.



[Görsel 8]: Gri kare bloklarının ekran görüntüsü.



[Görsel 9]: Zıplama ve yatma hareketlerinin ekran görüntüsü.



[Görsel 10]: Zıplama ve yatma hareketlerinin ekran görüntüsü.

V. CLICK TO MOVE KODLAMA

A. *StartAfterPlacement Metodu*

PlacementOnMesh_Character.characterPlaced etkinliği tetiklendiğinde çağrılır.Agent değişkeni, GameboardAgent türündeki bir nesneye atanır.IsPlaced bayrağı true olarak ayarlanır.MoveAlongWayPoints metodu bir Coroutine olarak başlatılır.

B. *Update Metodu*

IsPlaced bayrağı kontrol edilir; eğer yerleştirme yapılmamışsa geri dönülür.UNITY_EDITOR ortamında sol fare tıklaması algılandığında veya UNITY_IOS veya UNITY_ANDROID ortamlarında bir dokunma algılandığında, TouchToRay metodu çağrılır.Dokunma (veya fare tıklaması) bir UI öğesi üzerinde gerçekleşmişse, "UI Hit was recognized" logu yazdırılır ve fonksiyondan çıkılır.

C. *MoveAlongWayPoints Coroutine*

Sonsuz bir döngü içinde çalışır.Agent'in durumu ve yolun durumu (Path Status) kontrol edilir.Eğer karakterin yolu tamamlanmışsa ve hedeflenen yol noktası listesinde bir sonraki nokta varsa, karakterin hedefi ayarlanır.Eğer karakter boşta duruyorsa (Idle) ve hedeflenen yol noktası listesinde bir sonraki nokta varsa, karakterin hedefi ayarlanır.Belirli bir süre sonra tekrarlanır.

D. *TouchToRay Metodu*

Ekran üzerinde bir dokunuş veya fare tıklaması noktasını alır ve bu noktaya bir ışın çıkarır.Işın, bir nesne ile çarpışırsa, çarpışma noktasını wayPoints listesine ekler.Bu script, karakterin belirli bir noktaya tıklanarak hareket etmesini sağlar. wayPoints listesine eklenen noktalar, karakterin hareket etmesi gereken yolun belirlenmesini sağlar. Bu yol, karakter agent tarafından takip edilir ve belirli bir sırayla hareket eder. Script, dokunmatik ekranlarda veya fare kullanılarak karakterin hareketini kontrol etmek üzere tasarlanmıştır.

VI. GUNCHARACTERANIMATIONCONTROLLER KODLAMA

A. *ControlState Coroutine*

Sonsuz bir döngü içinde çalışır.Geçen zamanı (elapsedTime) ve son iki kare arasındaki mesafeyi hesaplar.Hızı hesaplar ve bu hızı kullanarak animasyon durumunu kontrol eder.Eğer hız belirtilen eşik değerinden büyükse ve karakter "isRunning" durumunda değilse, animasyon durumu "isRunning" olarak değiştirilir.Eğer hız belirtilen eşik değerinden küçükse ve karakter "isIdle" durumunda değilse, animasyon durumu "isIdle" olarak değiştirilir.Belirli bir süre sonra tekrarlanır.

B. *ChangeToState Metodu*

Animator parametrelerini döngüyle kontrol eder.Belirtilen durumu (setToState) true olarak ayarlar ve diğerlerini false yapar.Bu durumu Animator bileşenine uygular.Bu kod, karakterin hareket hızına bağlı olarak "isIdle" ve "isRunning" animasyon durumları arasında geçiş yapmasını sağlar.

gunCharacterAnimation değişkeni üzerinden karakterin animasyonlarına erişir ve belirtilen hız eşiğine göre animasyon durumlarını kontrol eder. Animasyon durumu değiştiğinde, ChangeToState metodunu kullanarak ilgili animasyon parametrelerini günceller. Bu sayede karakterin animasyonları, hızına bağlı olarak dinamik bir şekilde değişir.

VII. PLACEMENTONMESH_CHARACTER KODLAMA

A. Update Metodu

Eğer nesne yerleştirilmişse, fonksiyonu sonlandırır (return).UNITY_EDITOR ortamında sol fare tıklaması algılandığında veya UNITY_IOS veya UNITY_ANDROID ortamlarında bir dokunma algılandığında TouchToRay metodu çağrılır.Eğer dokunma (veya fare tıklaması) bir UI öğesi üzerinde gerçekleşmişse, "UI Hit was recognized" logu yazdırılır ve fonksiyondan çıkılır.

B. TouchToRay Metodu

Ekran üzerinde bir dokunuş veya fare tıklaması noktasını alır ve bu noktaya bir ışın çıkarır (Raycast).Işın, bir yüzeyle çarpışırsa, çarpışma noktasında placementObject'in bir kopyasını instantiate eder (oluşturur).Oluşturulan nesne placedObjects listesine eklenir.IsPlaced bayrağı true olarak ayarlanır.CharacterPlaced olayı tetiklenir ve diğer kodlar bu olaya abone olmuşsa, karakterin yerleştirildiği bildirilir.

SONUÇ

Proje, ARDK 3 ve ARKit kullanarak Unity üzerinde basit bir artırılmış gerçeklik oyununu başarıyla geliştirdik. Oyuncular, gerçek dünyada bir karakteri hareket ettirme deneyimini yaşayabilirler. Bu projenin temel amacı, AR teknolojilerini kullanarak oyun geliştirmenin başarıyla nasıl gerçekleştirilebileceğini öğretmektedir.

KAYNAKLAR

[1] ARKit: Temel Bilgiler. Apple'ın arttırılmış gerçeklik… | by Rana Taki | TurkishKit | Medium J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] lightship.dev/docs/ardk/

[3] siliconangle.com/2023/10/30/niantic-announces-major-3-0-release-lightship-ar-developer-platform/#:~:text=Niantic%20Inc.%2C%20the%20developer%20of,features%20and%20easier%20multiplayer%20support.

[4] lightship.dev/docs/ardk/setup/