Design It
BİL496 / YAP496
2. Sprint Sonu Raporu

(Barış Utku Mutan 201104082, Gökçe Başak Demirok 191401005, Murat Şahin 191101010, Mehmet Can Şakiroğlu 191101001)

Genel Bakış

İlk sprintte yaptığımız araştırmalar doğrultusunda elde ettiğimiz bilgileri eyleme dökmeye başladığımız bu sprintte github repomuzun proje üyeleri arasındaki senkronizasyonunu sağlayıp uygulamamıza planlanan fonksiyonları eklemeye başladık.

Bu sprintte yapılan işleri konu başlıkları altında açıklayalım:

Yapılan İşler

- **Ev eşyası talebi konusunda sesli algılanan komutun anlaşılması ve gerekli bilginin çıkarılması**

Bu görev için öncelikle adlandırılmış varlık tespiti (named entitiy recognition- NER) modeli bulup transfer learning ile ev eşyası tespitinin yapılması planlanmıştı. Fakat buna uygun önceden eğitilen bir modele web araştırmaları sonucu ulaşılamadığından iptal edilmişti. Sonraki adımda çoğu ev eşyasının sıfat tamlaması şeklinde olması, bir sıfat tamlaması bulan modeli kullanmanın işe yarayabileceğini düşündürmüştü. 1. Sprintte bu konudan bahsettik ve sonrasında bu modelin isim tamlaması ya da sadece isimden oluşan eşya isminin bulunmasında sıkıntı yarattığını fark ettik.

Halihazırda bir model problemimize çözüm bulamadığı için bu sprintte bert tabanlı bir token classification modeli üzerine çalıştık. Ve karşılaştığımız sorunlara çözümü bulduk. Hazırladığımız modelin girdisi cümle, çıktısı ise bu cümledeki (eğer varsa) ev eyşasını oluşturan kelime ya da kelimeler bütünü. Örnek girdi ve çıktı “Sesli algılanan komutun yazıya dökülmesi (Speech Recognition)” ksımındaki videoda gösterilmiştir.

- **Veri Hazırlama**

Düşündüğümüz modelin girdisi cümleler olduğu için öncelikle yüksek sayıda cümleden oluşan veriye ihtiyacımız vardı. Geçmiş çalışmalarımızda deneyimlediğimiz üzere bert tabanlı transfer learning ile oluşturulacak bir modelde iki bin civarı verinin yeterli olacağını düşündüğümüzden ve bunu elimizle yazmaya vaktimiz olmadığından data augmentation yöntemiyle 2260 cümlelik bir veri oluşturduk. Data augmentation yapmak için öncelikle chatGPT'den içerisinde ev eşyası (furniture) geçen 100 cümle yazmasını istedik ve bir sonraki satırda bu ev eşyası kelimelerinin olduğu yerlere [MASK] yazmasını istedik. Sonrasında https://7esl.com/furniture-vocabulary/ linkinde bulunan kelimelerden ev eşyası sözlüğü oluşturup (linkte yer almayan ama aklımıza furniture olarak istenebilecek olan bilgisayar, tablet gibi kelimeleri de bu sözlüğe ekledik) yazdığımız bir metod aracılığıyla (create_data.ipynb / `func_create_data`) bu [MASK] yazan yerlere sözlükte bulunan kelimelerden koymasını sağlayarak datayı artırdık. Karşılaşılan bir sorun chatgpt'nin birden fazla ev eşyası ile oluşturduğu cümlelerde sadece bir ev eşyasına [MASK] atamasıydı. Bunu da sonrasında manuel olarak cümleleri kontrol edip ekstra olan yerlere de [MASK] ataması yaparak düzelttik. Oluşturulan verinin etiketleme sürecinde ise [MASK] yerine atanan kelimeler (kelime sayısına göre sırayla) B-FUR ve devamında I-FUR olmak üzere, geriye kalan kelimeler içinse O şeklinde atama yaparak gerçekleştirdik. Burada
B-FUR: ev eşyası olarak etiketlenen kelime ya da kelime grubunun başlangıç kelimesi
I-FUR: eğer ev eşyası kelime grubuysa başlangıcından sonraki gelen kelime
O: ev eşyası olmayan kelimelerin etiketleri

- **Model Oluşturma**

Bu task'in tamamlanması için token classifiation modeli geliştirdik. Modelin girdisi cümle ve çıktısı cümle içerisindeki kelimelerin etiketleri şeklinde. Örnek olarak:

Input: "I want a table lamp."
Output: "table": B-FUR
"lamp": I-FUR

Veri hazırladıktan sonra modelin kısa sürede eğitilebilmesi için ve komutları çok karışık beklemediğimiz için görece daha az parametreyle eğitilmiş olan distilbert modelini kullandık. Sonrasında ise bu modeli Hugging Face'e atarak kolay kullanılabilir olmasını sağladık.

Model oluşturmamız sonucunda elde ettiğimiz en güzel şey eğitim yaptığımız veride bulunmamasına rağmen eş anlamlı kelimeleri yakalayabilmesi. Tabi bunun sebebini de modele kelimelerin embedding olarak entegre edilmesine bağladık.

Model sonucu fark edilen sıkıntı, sıfat tamlaması şeklinde olan ev eşyalarında sıfatın saptanamaması oldu (Input: "I want a round table." -> Output: "table). Bunun içinse

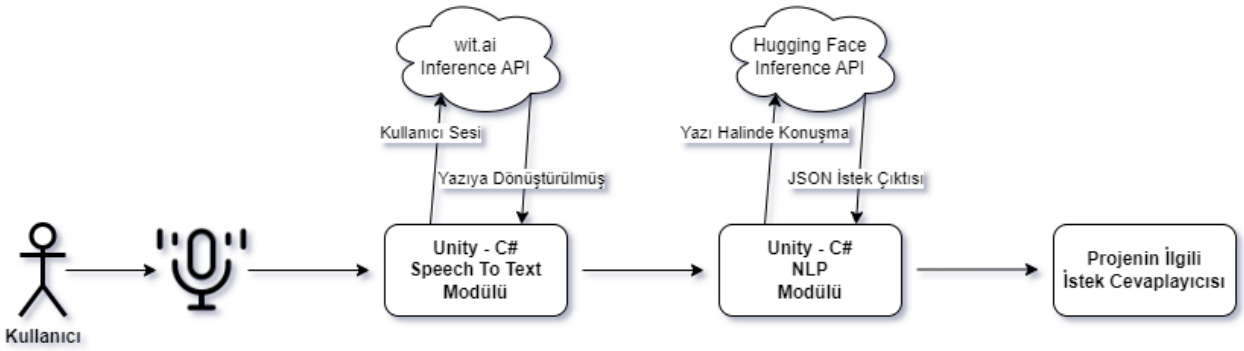
gelecek sprint sonuna kadar modeli sıfat tamlamasını içeren verilerle beslemeyi düşünüyoruz.

- Sesli algılanan komutun yazıya dökülmesi (Speech Recognition)

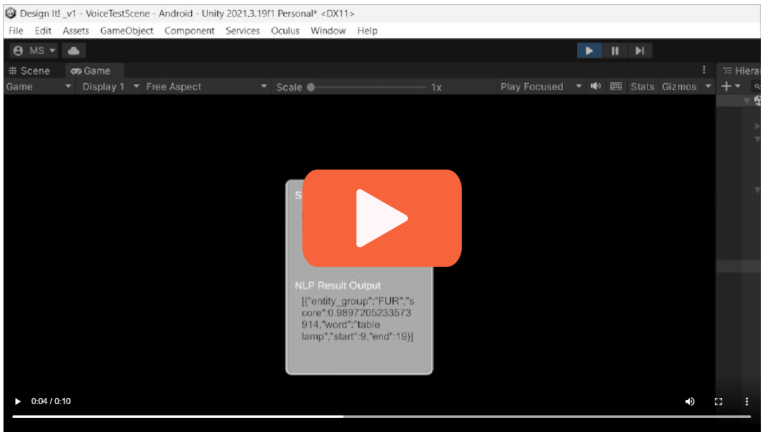
Bu task için başta planımız Kaldi ASR kullanmaktı, fakat daha sonradan Oculus ve Unity işbirliğiyle ortaya çıkan Voice SDK'i fark ettik. Bu SDK kolayca wit.ai hizmetlerinden yararlanabilmemizi sağlıyor. Sunulan hizmetler arasında da bizim de istediğimiz konuşmadan yazıya dönüştürme işlevi mevcut. SDK tarafından sunulan scriptler ve kendi yazdığımız scriptler yardımıyla beklenen çıktıyı alabiliyoruz.

Ayrıca bu task'a ek olarak bu issue kapsamında bu modül başka bir taskta geliştirilmiş olan NLP modeline bağlanarak sesten aksiyona giden pipeline kurulmuş olundu. Pipeline tasarlanırken kurulan birbirinden bağımsız yapılar sayesinde NLP modeli bağımsız bir şekilde gelişmeye devam ederken projemizde otomatik olarak güncellenebilecek diyebiliriz.

Kurulmuş olan yapı:



Demo videosunu aşağıdan izleyebilirsiniz:



(Videoyu açmak için görselin üstüne tıklayıp linke basabilirsiniz.)

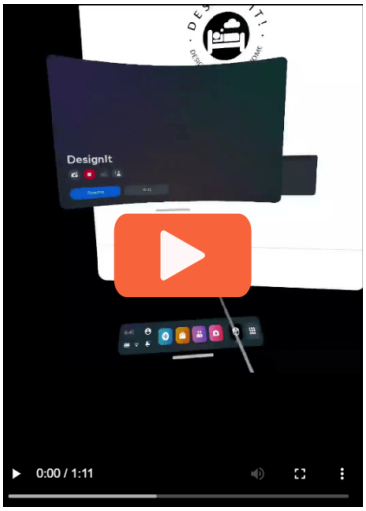
- Kimlik doğrulama işlevinin yapılması ve entegre edilmesi

Auth0 Lab Unity SDK kullanılarak task tamamlandı. Authentication sağlandı ve projeyle bağlantısı kuruldu, kullanıcılar uygulamada ilerlemek için giriş yapmak zorundalar. Ayrıca estetik açıdan güzel görünmesi için uygulamanın genel menüsü ile authentication menüsü aynı şablona oturtuldu.

Çalışma sistemi aşağıdaki şekilde:

1. Kullanıcı uygulamayı açar, giriş yap tuşuna basar.
2. Bir adet URL ve 8 haneli kod verilir.
3. Kullanıcı verilen URL'ye bir mobil cihazdan girer ve kodu girer.
4. Kullanıcı mobil cihazından kolayca giriş yapar ve senkronize olarak headsette de giriş yapmış olur.

Demo videosunu aşağıdan izleyebilirsiniz:

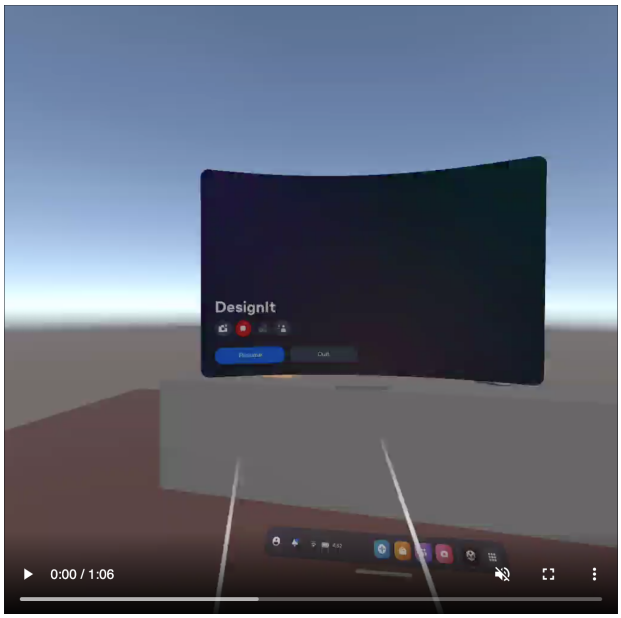


(Videoyu açmak için görselin üstüne tıklayıp linke basabilirsiniz.

- ## Unity’de Modellerle Seç-Sürükle-Bırak-Rotate Gibi Etkileşime Geçilebilmesi

Bu issue’da, Unity platformunda projemiz kapsamındaki odada olan eşya modellemeleriyle etkileşime geçilebilmesi amaçlanmaktaydı. Modellemelere, Rigidbody ve OVR Grabbable component'larının eklenmesi ve eşyaların bir Collider'a sahip olmasıyla beraber bu issue çözümlenmiştir. Issue'ya dair geliştirmeler [#9-Inventory-List-of-Items-and-#13-Grab-Interact] branch'indeki [Commit on Issues #9 and #13] commit'i ile proje eklenmiştir.

İlgili geliştirmeler sonrasındaki demo videosu aşağıya eklenmiştir:



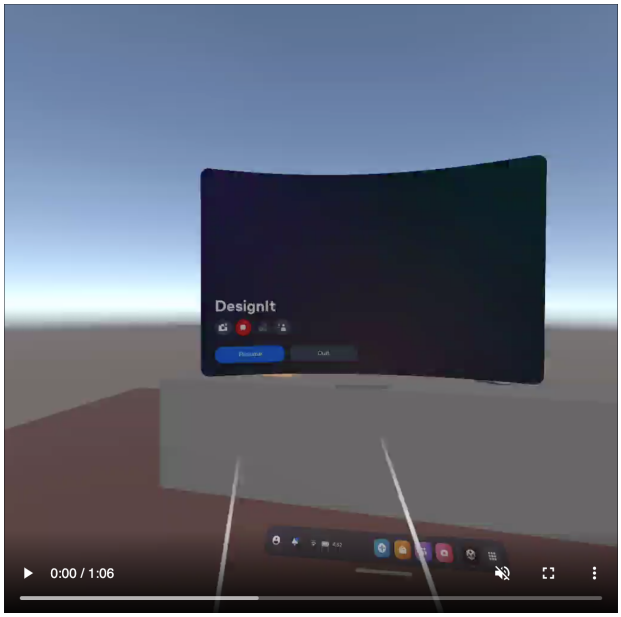
(Videoyu açmak için görselin üstüne tıklayıp linke basabilirsiniz.)

- ## VR’da Eşya Modellerinin Listelenerek Gösterilebilmesi

Bu issue’da, eşyaların modellemelerinin listelenmiş olması ve kullanıcının bunlarla sürükle-odaya-bırak şeklinde etkileşime geçebiliyor olması amaçlanmaktaydı. Bu issue kapsamında öncelikle Inventory adlı modellemeleri tutabilen bir liste oluşturuldu, bu liste sol controller'daki B tuşuna basılarak açılıp kapanabiliyor. Kullanıcı buradaki inventory'sinden modellemeleri çekip odaya istediği bir yere koyabiliyor ve isterse odadan bir modellemeyi çekip inventory'sine koyadabiliyor. Bunun da birden fazla eşyayı aynı anda bir yerden başka bir yere taşımak için faydalı olacağını düşündük. Böylelikle bu issue tamamlandı. Kodlama kısmında ise ilgili modellemelere yazılmış olan Item script'inin de eklenmesiyle bu sağlanabiliyor. Inventory kısmı ile ilgili olarak da InventoryVR ve Slot isimli iki tane script mevcut. YOLO'dan çıkan tespit edilmiş şeylere ait isimlere ve modellemelere erişebileceğimiz bir pipeline henüz

tamamlanmadığı için sadece issue'nun bu kısmındaki deneme kutu ve kürelerle yapıldı, çalışıyor. İleride YOLO ile alakalı kısımlar da tamamlanınca onlarla da denenecek ancak temel prensibinde çok fark olmadığı için problem olmayacağını düşünmekteyiz. Issue'ya dair geliştirmeler [#9-Inventory-List-of-Items-and-#13-Grab-Interact] branch'indeki [Commit on Issues #9 and #13] commit'i ile proje eklenmiştir.

İlgili geliştirmeler sonrasındaki demo videosu aşağıya eklenmiştir:



(Videoyu açmak için görselin üstüne tıklayıp linke basabilirsiniz.)

- YOLO Training

Bu issue'da, YOLO'nun, bir furniture dataseti ile eğitiminin yapılması amaçlanmaktaydı. Ancak seçilen YOLO version'ı olan YOLOv5'ın hazır ve açık kaynak modellerinden olan yolov5s.pt, yolov5m.pt gibi modeller zaten train edilmiş modeller ve hali hazırda birçok detect etmesini isteyebileceğimiz objeyi tanıyabilir durumdalar. Ayrıca train etmek için kullanmak isteyebileceğimiz furniture konusunda güzel içeriğe sahip, en kapsamlı datasetler ile de bu model zaten train edilmiş durumda.

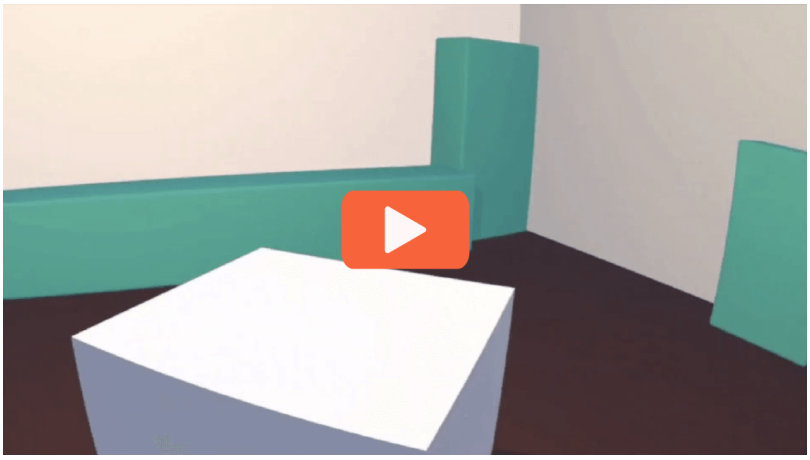
Bununla ilgili olarak train edilmiş olan dataset ve hali hazırda tanınabilen bir takım objelerin listesi aşağıdaki gibi:
- CHAIR, COUCH, POTTED PLANT, BED, DINING TABLE, TV, TOILET, REFRIGERATOR, MICROWAVE, OVEN, TOASTER, SINK, CLOCK, VASE, LAPTOP, BOOK, TEDDY BEAR

Dolayısıyla bu issue, YOLO'yu train etmeye gerek kalmadığı için kapatıldı.

- ## Oynama esnasında yeni objelerin dinamik olarak yaratılması

Bu issue kapsamında Unity platformunda oyun oynanırken kullanıcının yeni eşyaları odada yaratabilmesinin temel geliştirme sürecini gerçekleştirmek amaçlanmıştır. Bu hedefin gerçekleştirilebilmesi için açılan issue #28 kapsamında #28-render-instantiate branchindeki dd20651 hashli ‘Spawning objects on raycast trigger’ commit’i ile proje eklenmiştir. Hedefin gerçekleştirilmesi amacıyla RightControllerAnchor isimli sağ joystick’i temsil eden GameObject’e rayspawn.cs scripti atanmıştır. Bu script sağ joystick’teki tetik tuşu çekildiği zaman joystick’in render edilmiş odadaki Transform’unu baz alarak joystick’in o anki oryantasyonu doğrultusunda bir ışın çıkarıp. Işın bir objeye vurunca oluşan RayCastHit objesine dair bilgileri kullanarak Unity’nin built-in Instantiate() metodunu çağırarak yeni objeyi yaratmaktadır. Işının bir objeyle etkileşime geçebilmesi için oda ilk render edilirken dinamik olarak çıkarılan tüm fiziksel objelere collider eklenecek şekilde FurnitureSpawn.cs’i de güncellenmiştir.

İlgili geliştirmeler sonrasındaki demo videosuna aşağıda erişebilirsiniz:



(Videoyu açmak için görselin üstüne tıklayıp linke basabilirsiniz.)

Sonraki Sprint Planlaması

- ## Oyun esnasında 3D model yaratılması, silinmesi

İkinci sprintte kullanıcının odaya küp eklemesini sağlamıştık. Artık küp yerine gerçek mobilya (masa, sandalye vb.) eşyaların 3D modellerinin düzgün bir şekilde kullanıcı tarafından odaya eklenebilmesi ve eklenen eşyaların da silinebilmesi hedeflenmektedir. Bununla ilgili issue https://github.com/cansakiroglu/DesignIt/issues/36 olarak tanımlanmış, Barış Utku Mutan ve Murat Şahin’e atanmıştır.

- Ray-cast'ın 3D render edilmesi

Kullanıcının yeni objeleri yaratacağı koordinatı tam olarak belirleyebilmesi için joystickten yapılan ray-casting işlemindeki ışının kullanıcının görebileceği şekilde her frame render edilmesi hedeflenmektedir. Bununla ilgili issue https://github.com/cansakiroglu/DesignIt/issues/37 olarak tanımlanmış ve Barış Utku Mutan'a atanmıştır.

- Headset - Bilgisayar Arası Obje Tespiti İçin API Oluşturulması

Geliştirilen obje tanıma modülünün çalışması için tamamen headsete gömülü bir sistem oluşturamıyoruz, çünkü önceki raporumuzda da bahsettiğimiz gibi Meta güvenlik nedeniyle uygulamaların Passthrough API yardımıyla gelen görüntüleri işlemesini mümkün kılmamış. Dolayısı ile görüntülerin headsetten bilgisayara, daha sonra da sonuçların bilgisayardan headsete aktarılması gerekli. Sidequest kullanarak kablo ile görüntü alıp, yine kablo ile sonuç aktarımını yapabilirdik. Fakat bunu daha kullanışlı şekilde yapmak için ADB (Android Debug Bridge) aracını kablosuz olarak kullanacağız. Bilgisayarla headseti aynı ağa bağlayıp ADB yardımıyla komut bağlantısı kuracağız. İki cihaz arasında sürekli mesajlaşmalarla sonuç olarak headsetin ilettiği bir fotoğrafa karşılık headsete bir etiket dönülmesini planlıyoruz.

İlgili issue https://github.com/cansakiroglu/DesignIt/issues/38 olarak tanımlanmış ve Murat Şahin'e atanmıştır.

- YOLO'nun VR'dan Gelen Görüntülerle Çalışma Performansını İyileştirme

Kullanacağımız YOLO modeli normal görseller üzerinde oldukça yüksek bir başarımla furniture detection yapabilmekteyken, VR'dan gelen görüntülerle çalıştırıldığında daha düşük bir başarımla çalışıyormuş gibi gözükmekte. Bunun kaynak sebebi olarak; görüntülerin çözünürlüklerinin düşük olması, fisheye formatında olması, noisy olması, edgelerin keskin gözükmemesi gibi potansiyel sebepler aday gösterilebilir. Bu gibi durumlara karşın denenebilecek çeşitli metotlar mevcut. Bunların görüntülerde denenmesi gibi yöntemlerle modelin başarımını arttırmak amaçlanmakta.
Bu kapsamdaki issue GitHub'da https://github.com/cansakiroglu/DesignIt/issues/35 olarak tanımlanmış ve Mehmet Can Şakiroğlu'na atanmıştır.

- NLP Model Geliştirmesi

Sıfat içeren ev eşyalarının sıfatını anlamakta problem yaşayan modelimizi bu sprintte sıfatlı cümle içeren data ile besleyerek problemin çözülmesi. Bu kapsamdaki issue GitHub'da

https://github.com/cansakiroglu/DesignIt/issues/39 olarak tanımlanmış ve Gökçe Başak Demirok'a atanmıştır.

- Kural tabanlı komut algılama sisteminin oluşturulması

Aşağıda bulunan komutların algılanması için kural tabanlı bir model sisteminin tasarlanıp kurulması.

*Video chat/ görüntüyü aç kapa, sesi aç kapa, çağrıyı başlat çağrıyı bitir
*Envanteri aç kapa / oyun için menüyü aç kapa
*Menüye dön

Bu kapsamdaki issue GitHub'da https://github.com/cansakiroglu/DesignIt/issues/30 olarak tanımlanmış ve Gökçe Başak Demirok'a atanmıştır.