R3-A4. Izrada smjernica za prilagodljivu i potpuno aktivnu (imerzivnu) obuku na alatu VR (virtualna stvarnost).

Erasmus+

UVOD

Pomak prema uključivijoj i prilagodljivijoj obuci u sektoru prirodnog kamena poprima nove oblike kreiranjem visoko imerzivnog načina učenja koristeći se alatima virtualne stvarnosti (VR). Ovaj projekt je značajna komponenta integracije naprednih tehnologija u samu obuku, čime se omogućuje personalizirano i učinkovito okruženje za učenje.

Kreiranje obrazovnih smjernica ove vrste zahtijeva detaljan razvojni proces koji počima od identifikacije ključnih scenarija do njihove implementacije u VR okruženju. Započinje fazom istraživanja i analize, gdje se pažljivo odabiru najreprezentativnije situacije i zadaci sektora. Zatim nastavljamo s osmišljavanjem načina poduke koji podjednako uzimaju u obzir funkcionalnost i pristupačnost, osiguravajući da je svaka faza obuke smislena i ostvariva za sve korisnike.

Na ovaj način, VR scenariji su namijenjeni, ne samo da točno simuliraju radne uvjete, već i za prilagodbu specifičnim obrazovnim potrebama korisnika. Rezultirajuće imerzivno iskustvo ima za cilj ne samo poučiti, već i osnažiti korisnike, omogućujući im da razviju praktične vještine u sigurnom i kontroliranom okruženju, spremnom za primjenu u stvarnom svijetu.

Konačno, ovaj obrazovni alat ne samo da postaje most do poslovne kompetencije, već i sredstvo za poticanje većeg razumijevanja i prihvaćanja različitosti na radnom mjestu.

Ovo izvješće i sve informacije o projektu su dostupne na Inclusive Stone web stranici: https://inclusivestone.eu/

Content

1 IZRADA ALATA

Srž ovog projekta temelji se na aktivnim mogućnostima koje nudi VR, virtualna stvarnost. Trenutno, standard za izradu aplikacija u ovom području se temelji na korištenju grafičkih alata koji dolaze iz svijeta videoigara. Iz tog razloga, pristup usvojen za razvoj naših obrazovnih alata slijedi metodologiju sličnu onoj kod stvaranja video igre.

Ova metodologija počiva na tri temeljna stupa koja će biti detaljno analizirana prema dolje navedenom:

- **Dizajn scene**
- **Razvoj 3D okruženja.**
- **Razvoj funkcionalnosti i direktnog iskustva.**

Kako bi se razvoj proveo učinkovito i djelotvorno, treba napomenuti da su korištene različite tehnologije:

- **Blender:** Besplatni softverski alat koji obuhvaća više platformi, koristi širok raspon funkcija 3D grafike kao što su modeliranje, animacija i prikazivanje. Mogućnost slobodnom pristupu i široka zajednica korisnika olakšavaju učenje kroz upute i pregršt dostupnih informacija.
- **Unity:** Ovaj grafički alat, također otvorenog tipa, široko se koristi u razvoju videoigara. Ističe se mogućnošću nadogradnje i prilagođavanja putem skripti, podržanih širokom zajednicom i kompletnom dokumentacijom.
- **Oculus Quest:** Uređaj za virtualnu stvarnost koji je jednostavan za pristup i korištenje, a koji je postao jedan od bitnih oslonaca Metine vizije budućnosti. Nudi prednost bežične veze i besprijekorne integracije s Unityjem, čime se poboljšava pristupačnost i raznolikost u razvoju VR aplikacija.

2 DIZAJN SCENE

Ovaj postupak uključuje detaljno planiranje niza koraka koje će korisnik uraditi od početaka korištenja ovog alata do trenutka kada postigne potpuno aktivno iskustvo. Ukratko, radi se o izradi skripte koja će biti vodić kroz cjeloviti razvoj aplikacije.

Inicijalno je odabrano 6 radnih scenarija koji na odgovarajući način predstavljaju različite uloge identificirane u radnom paketu R1, i to zbog svoje prilagodljivosti i činjenice da ne predstavljaju opasnost za osobe s različitim poteškoćama. Odabrane situacije su sljedeće:

- **Radnik na viličaru - prijevoz tereta:** U kontekstu ovog zadatka, cilj će biti utovar i istovar tri palete mramornih ploča, njihovo premještanje iz pogona za preradu do određenog skladišnog prostora. Ova će aktivnost uključivati manevriranje u područjima s ograničenim prostorom i slaganje materijala značajne težine.
- **Radnik na viličaru – utovar kamiona:** U ovom specifičnom scenariju, cilj je utovariti transportno vozilo s tri palete mramornih ploča, počevši od skladišnog prostora. Operacija će zahtijevati izvođenje manevara u ograničenim prostorima i skladištenje teških tereta.
- **Radnik na mosnoj dizalici - rukovanje pločama:** U ovom zadatku, korisnik će nastojati premjestiti 2 mramorne ploče u osvijetljeni skladišni prostor, izvodeći potrebne manevre pridržavajući se sigurnosnih propisa.
- **Radnik na mosnoj dizalici – rukovanje blokovima:** Kod izvođenja ove aktivnosti, cilj korisnika biti će prevesti blok mramora od kamiona do prostora za utovar, izvodeći potrebne manevre strogo poštujući važeće sigurnosne propise.
- **Radnik na čišćenju:** Svrha ove aktivnosti je identificirati i odabrati odgovarajuću osobnu zaštitnu opremu (PPE) za čišćenje i dezinfekciju radnog pogona, uklanjanje smeća. To se mora provoditi odgovorno i uz najveće poštovanje prema okolišu.
- **Upravljanje otpadom:** Kod izvođenja ovog zadatka, korisnik je odgovoran za rukovanje kemikalijama i otpadom. To uključuje odgovarajući odabir tehnika i alata za recikliranje i pravilno zbrinjavanje otpada, kao i pridržavanje smjernica usmjerenih na osiguranje odgovornog i ekološki održivog gospodarenja otpadom.

U tom kontekstu, odlučeno je dizajnirati model scenarija sastavljen od sekvencijskih faza ili misija, u stilu videoigara; to jest, možete prijeći na sljedeći korak tek nakon što završite prethodni, i to postupno. Stoga se korisniku nameće potreba da se cijeli proces sigurno završi od početka do kraja, a svaka nemogućnost da se to učini kvalificira se kao neuspjeh u scenariju.

Unutar ovih faza, početna faza će se sastojati od detaljnog objašnjenja o određenom stroju ili zadatku. Na taj će način korisniku biti lakše naučiti kontrole koje odgovaraju svakoj situaciji, Na taj će način korisniku biti lakše naučiti alate kontrole koje odgovaraju svakoj situaciji,

omogućujući veću fluidnost u radu bez potrebe da ih se često provjerava te nastaviti s konkretnim zadacima, što će ovisiti o svakoj situaciji.

2.1 Upute/Tutorial

Tijekom faze dizajna VR aplikacije, implementirali smo osnovni element za snalaženje i pripremu igrača: tutorial. Ova komponenta je ključna jer pruža strukturiran i metodičan uvod u okruženje igre, dostupne alate i alate za ostvarivanje cilja.

Ploča s uputama pomno je razvijena kako bi se osiguralo sveobuhvatno iskustvo učenja. Započelo je stvaranjem detaljne skripte koja pristupa svakom bitnom koraku koji igrač mora razumjeti prije nego što se upusti u aktivnost.

Prvi dio uputa pokriva rukovanje i funkcije strojeva kojima će igrači upravljati tijekom igre. Nakon toga, upute predstavljaju alat za teleportaciju, ključnu mehaniku koja proširuje strateške mogućnosti i mogućnosti mobilnosti unutar igre. Razvijene su posebne vježbe koje omogućuju igračima da eksperimentiraju s ovim alatom u kontroliranom okruženju, omogućujući njihovo savladavanje prije nego što se suoče sa sličnim situacijama u stvarnosti.

Osim toga, nadzorna ploča uključuje informativne module o korištenju svih dodatnih funkcija koje su važne za zadatak. Svaki element je pažljivo objašnjen i popraćen praktičnim primjerima kako bi se osiguralo cjelovito razumijevanje njegove uporabe i primjene.

Slika 1: Panel tutorial

Realizacija ove ploče s uputama bila je jedan od kamena temeljaca u razvoju naše video igre, osiguravajući da svaki korisnik dobije znanje i iskustvo potrebno za potpuno uživanje u igri i uspjeh u zadatku, a time i u stvarnom radu.

2.2 Misije (Zadaci)

U razvoju našeg projekta implementirali smo sustav misije koji je temeljan za napredovanje korisnika u učenju posla. Ovaj je sustav pomno dizajniran kako bi odgovarao različitim ulogama i poslovima.

Misije spadaju u dvije glavne kategorije koje doprinose raznolikosti i bogatstvu alata:

- **Misije aktivnosti**: Ove su misije srž interaktivnosti unutar igre. One od igrača zahtijevaju da bude aktivno uključen, dovršavajući zadatke koji mogu uključivati bilo što, od upravljanja strojevima do korištenja ruku za podizanje i kontrolu čišćenja. Ove smisije su osmišljene za testiranje motoričkih i strateških vještina korisnika.
- **Q&A misije**: U ovim zadacima igrač se suočava sa situacijama koje zahtijevaju razmišljanje i donošenje odluka na temelju informacija dobivenih na tečajevima koje korisnik mora imati kako bi ova aplikacija bila stvarno korisna. Kroz ploče s pitanjima i odgovorima, ovi zadaci procjenjuju korisnikovo razumijevanje i stečeno znanje.

Slika 2:Zadatak prijevoza blokova

Svaka misija, bez obzira na kategoriju, pažljivo je isprepletena sa sljedećom, osiguravajući da je iskustvo svakog igrača kohezivno i duboko aktivno. Osim toga, ovaj dizajn misije promovira prirodnu krivulju učenja, gdje igrači mogu poboljšati svoje vještine kako napreduju kroz igru.

Ukratko, naš sustav zadataka ne samo da nastoji zabaviti, već i uključiti igrača u proces kontinuiranog učenja i otkrivanja, što je bitno za sveobuhvatno iskustvo koje želimo ponuditi.

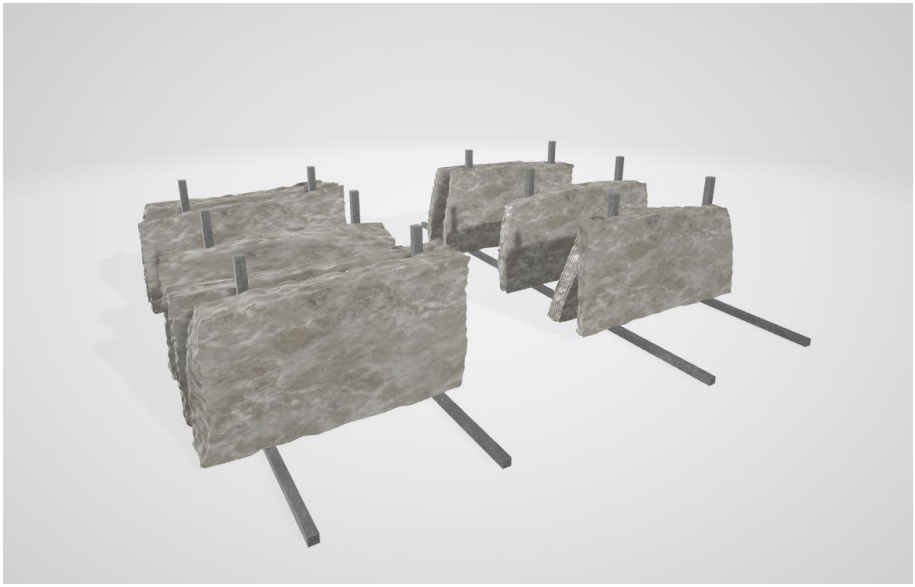
3 RAZVOJ 3D OKRUŽENJA

Stvaranje alata unutar grafičkih mehanizama videoigara često uključuje rukovanje objektima. Stoga je drugi glavni zadatak izrada raznih predmeta ili sredstava, koji su navedeni izravno ili neizravno u prethodno pripremljenim tehničkim skriptama. Ovaj postupak je podijeljen u tri osnovne faze:

- **Modeliranje**: Matematički prikaz nekog objekta se radi u tri dimenzije pomoću specijaliziranog softvera. Za ovaj projekt odlučili smo koristiti Blender.
- **Namještanje**: U ovoj fazi, sastavni dijelovi objekta se rastavljaju kako bi formirali digitalni kostur koji se može kontrolirati, što olakšava izradu glatkih i preciznih animacija. Ovaj korak je ključan za objekte koji će zahtijevati animaciju, a također je proveden koristeći Blendera.
- **Teksturiranje**: Tijekom teksturiranja, primjenjuju se boje i detalji na 3D modele kako bi im se dao realističniji i detaljniji završni izgled, ponovno koristeći softver Blender.

Raspon predmeta napravljenih za ovaj projekt pokriva širok i raznolik spektar, u rasponu od krajnje jednostavnih elemenata do složenih i detaljnih strojeva. Proizvedeni objekti su organizirani u različite kategorije, prema njihovoj složenosti i funkciji unutar igre. Možemo pronaći:

- Kontejneri, zgrade, police i ukrasni elementi.

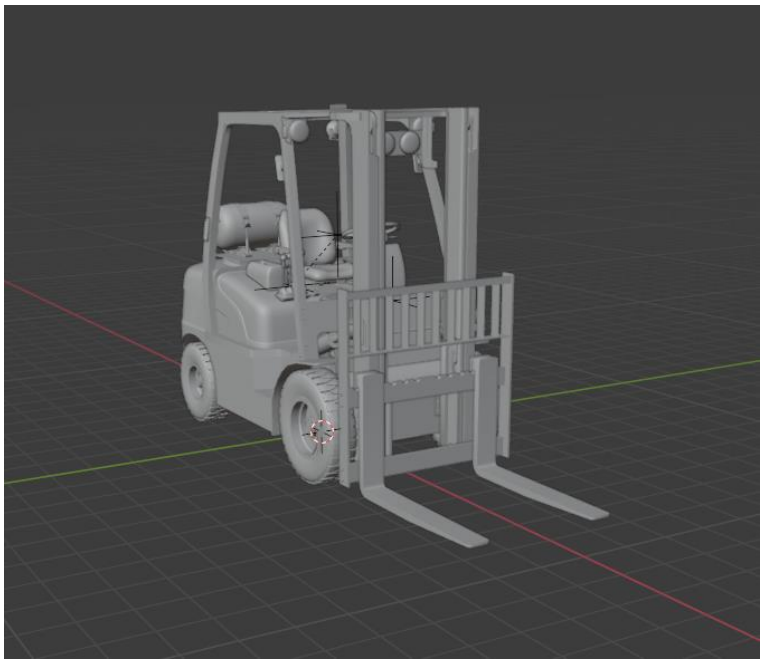


Slika 3: Kamene ploče.

- Osobna oprema i korisni predmeti, ograničene uporabe.

Slika 4: Drugi elementi.

- Posebni strojevi za različite situacije.



Slika 5: Model viličara

Nakon što su objekti koji će se koristiti u igri dizajnirani i kreirani, sljedeći korak je priprema scenarija. Ovaj je proces ključan za samu funkcionalnost igre kako bi točno odražavao okruženje u kojem će se odvijati konačna aktivnost. Važno je napomenuti da se ovaj proces ponavlja i podložan je prilagodbama jer naknadno testiranje može otkriti potrebu za izmjenama. Za izradu scenarija korišten je Unity graphics alat koji će biti isti onaj koji će se koristiti za izradu konačne aplikacije.

Budući da su radna mjesta koja su odabrana u prethodnim zadacima radna mjesta koja se obavljaju u tvornici, a ne u kamenolomu, razvijen je samo jedan scenarij. Scenarij dizajniran za ovaj projekt je sljedeći:

- **Tvornica prirodnog kamena**: Pokazuje se virtualni prikaz stvarne tvornice, s jasno izdvojenim područjima uključujući strojarnicu, skladište kemikalija i odjel za ekstrakciju otpada. Ovaj detaljni dizajn pomaže korisnicima da se upoznaju s određenim radnim okruženjem i raznim zadacima koji se obavljaju u industriji prirodnog kamena.

Slika 6: Prikaz tvornice

4 FUNKCIONALNOST RAZVOJ I AKTIVNO (IMERZIVNO) ISKUSTVO

Razvoj funkcionalnosti je korak u kojem prethodno prikupljeni koncepti i informacije dolaze u život. Svrha ove faze je pretvoriti teoretski alat u praktičnu i operativnu primjenu. Za izvršenje ovog zadatka odabrali smo grafički alat Unity zbog njegove jednostavnosti ugradnje elemenata virtualne stvarnosti (VR) , kao i zbog široke podrške koju nudi zajednica korisnika.

U **Unity-**ju projekti su strukturirani u Scene koje se mogu povezati kroz niz događaja. Svaka Scena se sastoji od nekoliko objekata, a njima su dodijeljene specifične komponente koje pokazuju njihove atribute i ponašanja unutar igre. Proces razvoja funkcionalnosti je opsežan i može se podijeliti u nekoliko ključnih faza, uključujući **dizajn i arhitekturu alata**, **modeliranje podataka** za strukturiranje informacija, **integraciju SDK-a** za proširenje mogućnosti alata i **upravljanje događajima** za interaktivni i neprekinuti tijek igre.

4.1 Dizajn & Arhitektura

Dizajn i arhitektura uključuju detaljnu razradu prethodno razvijene skripte, s namjerom da precizno definiraju put koji korisnik mora slijediti i odluke koje će se morati donijeti u svakoj sceni ili zaslonu igre. Ovaj proces je ključan jer uspostavlja osnovnu strukturu na kojoj će se graditi korisničko iskustvo.

Tijekom ove faze uzima se u obzir logika navigacije, raspored interaktivnih elemenata i slijed događaja koji će voditi korisnika kroz igru. Interakcija sa svakim objektom i likom unutar virtualnog okruženja pomno je planirana, kao i prijelazi između scena, kako bi se osiguralo intuitivno i imerzivno (aktivno) iskustvo koje drži korisnika uključenim u priču i ciljeve igre.

Unutar projekta korisnici će se susresti s dvije vrste glavnih Scena dizajniranih u Unity app-u: scena izbornika i specifična scena za svaku radnu stanicu.

4.1.1 Name principal (Odabir imena)

Zaslon izbornika predstavljen je kao intuitivan i pristupačan uvod u virtualnom okruženju, gdje korisnici poduzimaju prve korake u alatu. Na početku su suočeni s osnovnim izborima: jezične postavke, koje osiguravaju da im iskustvo bude personalizirano i razumljivo. Nakon što se odaberu ove bazične postavke, zaslon polako vodi korisnika prema razumijevanju glavne svrhe scene, a to je izbor specifične radne situacije koja će se istraživati. Iako je interaktivnost u ovoj fazi namjerno ograničena kako bi se fokus zadržao na cilju učenja, iskustvo je osmišljeno na način da bude ugodno i usmjeravajuće, na taj način se priprema korisnika za prijelaz na sljedeću fazu, gdje će odabir u izborniku odrediti slijedeću Radnu scenu.

4.1.2 Status radne Scene

Ova faza predstavlja bit alata, gdje svaka predložena radna situacija oživljava u vlastitoj pojedinačnoj sceni. Slijedeći prethodno uspostavljenu shemu misije, korisnici su uronjeni u niz zadataka osmišljenih da oponašaju odgovornosti i izazove određenog radnog mjesta. Avantura

počinje detaljnim tutorialom (uputama), pomno prilagođenim neposrednim zadacima, pružajući tako čvrstu osnovu za učenje i izvršavanje sljedećih misija.

Kako korisnik napreduje i prolazi svaku misiju, sljedeća se otključava, što omogućuje linearni napredak i osjećaj stalnog postignuća. Nakon uspješnog suočavanja i dovršetka posljednje misije, nudi vam se opcija da ponovno pokrenete tu Scenu i usavršite svoje vještine ili se vratite na glavni izbornik da istražite i naučite o drugom poslu, čime se olakšava kontinuirani proces učenja i šire istraživanje raznih aspekata industrije prirodnog kamena.

4.2 Modeliranje podataka

Cilj ove faze je stvoriti sveobuhvatni okvir za pohranu i dohvaćanje podataka. Odlučili smo koristiti PlayerPrefs, Unity modul koji olakšava čuvanje informacija putem sustava ključeva povezanih s nazivom projekta. U Oculus aplikacijama ove varijable se spremaju u obliku XML datoteke koja se nalazi u /data/data/pkg-name/shared_prefs/pkg-name.v2.playerprefs.xml direktoriju, gdje "pkg-name" odgovara imenu dodijeljenom aplikaciji. Unity čini pristup ovim podacima iznimno lakim kroz funkcije klase PlayerPrefs.

Zbog načina na koji je ova aplikacija strukturirana, morat ćemo pohraniti zajedničku varijablu: onu koja određuje jezik. Koristit ćemo sljedeće značajke PlayerPrefsa za upravljanjem iste:

- **SetString (ime*, vrijednost*):** Ovom funkcijom dodijelit ćemo tekstualnu vrijednost, koja će biti ISO kod odabranog jezika, varijabli identificiranoj s "name".
- **GetString (*ime*):** Ova nam funkcija omogućuje dohvaćanje vrijednosti dodijeljene varijabli "name". Koristit ćemo ga u svakoj sceni za identifikaciju odabranog jezika.

4.3 SDK Integracija

Stvaranje aplikacija VR virtualne stvarnosti temelji se na tri temeljna stupa: kompatibilnom uređaju koji se može spojiti na računalo, grafičkom alatu za videoigre i kompletu za razvoj softvera (SDK). Na početku ovog poglavlja detaljno su opisana prva dva elementa, pri čemu je Oculus uređaj, a Unity odabrani grafički alat. Što se tiče SDK-a, koji je skup alata koji pomaže programerima softvera u procesu stvaranja, postoji nekoliko dostupnih opcija koje odgovaraju Unityju. Iako postoji posebna opcija za Oculus, odabran je XR Interaction Toolkit, paket koji je razvio Unity koji olakšava prilagodbu projekta različitim vrstama uređaja. To znači da, iako je alat inicijalno razvijen za Oculus Quest, može se instalirati na bilo koji uređaj koji je kompatibilan s Unity, čime se proširuje njegova pristupačnost i doseg.

Kako biste integrirali SDK i omogućili interakciju u Unity, slijedite korake u nastavku:

1 Kada pokrenete Unity, stvarate novi projekt pomoću predloška Universal Render Pipeline, koji je ključan za optimiziranu grafiku, koji je ključni aspekt za jednostavno i zadovoljavajuće VR iskustvo.
2 SDK, u ovom slučaju XR Interaction Toolkit, instalira se preko prozora Package Manager koji se nalazi u opciji izbornika Window.
3 Potrebno je navesti vrstu uređaja na kojem će se projekt razvijati. Da biste to učinili, potvrdite okvir VR podržan unutar Project Settings/Player.

S ovim koracima vaš je Unity projekt spreman za razvoj virtualne stvarnosti. Međutim, da biste započeli interakciju vašeg uređaja s grafičkim okruženjem, bitno je slijediti upute u sljedećem odjeljku.

1. Kada pokrenete Unity, stvarate novi projekt pomoću predloška Universal Render Pipeline, koji je ključan za optimiziranu grafiku, ključni aspekt za lagano i zadovoljavajuće VR iskustvo.

2. SDK, u ovom slučaju XR Interaction Toolkit, instalira se preko prozora Package Manager koji se nalazi u opciji izbornika Window.

3. Potrebno je navesti vrstu uređaja na kojem će se projekt razvijati. Da biste to učinili, potvrdite okvir VR podržan unutar Project Settings/Player.

S ovim koracima vaš je Unity projekt spreman za razvoj virtualne stvarnosti. Međutim, da biste započeli interakciju vašeg uređaja s grafičkim okruženjem, bitno je slijediti upute u sljedećem odjeljku.

4.4 Upravljanje događanjima

Upravljanje događanjima nedvojbeno je najširi dio u stvaranju funkcionalnosti naše aplikacije. Temelji se na korištenju komponenti koje nudi Unity, uz izradu sažetih skripti koje omogućuju željenu interakciju između korisnika i sustava. Budući da je svaka scena prepuna jedinstvenih detalja koji utječu na vašu interakciju s njom, ovaj bi se odjeljak mogao znatno proširiti. Međutim, u ovom ćemo se dokumentu usredotočiti samo na isticanje onih zajedničkih i ključnih aspekata koji su važni za postizanje naših ciljeva interaktivnosti.

4.4.1 Postavljanje scene za VR interakciju

Nakon što pripremite svoj Unity projekt i grafički dizajnirane scenarije, sljedeći korak je priprema scene za interakcijski uređaj za integraciju s kamerom, što olakšava testiranje tijekom razvoja. Ovaj postupak, opisan u sljedećim fazama, mora se izvršiti za svaku od scena u projektu:

1 Uklanja zadanu kameru koja se prema zadanim postavkama pojavljuje u sceni.
2 Napravite prazan objekt i dodajte mu komponentu XR Rig, koja će funkcionirati kao srž interakcije između korisnika i stroja. Nazvat ćemo ovaj objekt VR-Rig.
3 Unutar VR-Rig-a stvara prazan objekt koji će služiti kao početna referentna točka za interakciju korisnika. Nazvat ćemo ga Camera Offset.
4 Dodajte kameru kao dodatni dio Camera Offset i ovoj kameri dodajte komponentu Tracked Pose Driver. Ovaj element će djelovati kao naše virtualne oči, a komponenta će omogućiti rotaciju pogleda.

Konfigurirajte varijable komponente XR Rig (roditelj, VR-Rig) s referentnim položajem i kamerom koju ste upravo izradili. Postavite vrijednost 'pod' u atributu Tracking Origin Mode tako da se kamera automatski prilagođava visini korisnika.

5 TKako biste omogućili pokretljivost i vidljivost upravljačkog programa, kreirajte dva dodatna objekta kao dodatni dio Camera Offset i dodijelite im atribut XR Controller.

6 Dodijelite model koji želite koristiti kao kontrolere u atributu Prefab modela svakog XR kontrolera.
7 Konačno, ovim dvama objektima dodaje se komponenta XR Direct Interactor, koja će pružiti mogućnost interakcije s drugim objektima u sceni.

4.4.2 Kontrolni unosi

Ovaj postupak se odnosi na dobivanje vrijednosti odaslanih interakcijom pomoću gumba na kontrolama. Te su vrijednosti presudne za izvođenje raznih radnji unutar aplikacije, poput hvatanja objekata ili odabira različitih opcija. Prije nego što nastavite s bilo kojom operacijom s ovim podacima, bitno je razumjeti vrstu informacija koje pruža svaki unos iz kontrola. Ove informacije mogu se vidjeti u prozoru XR Interaction Debugger, kojem se može pristupiti putem Window/Analysis.

Nakon što to shvatite, sljedeći korak je izrada skripte koja vam omogućuje prikupljanje spomenutih vrijednosti. Klasa stvorena u tu svrhu naziva se HandController (Ručna kontrola) i dodana je kao komponenta svakom od 3D modela navedenih u atributu Model Prefab kontrola. Istaknute karakteristike koda uključuju:

```
public class HandController : MonoBehaviour
{
    //Input
    private InputDevice targetDevice;

    //Controller model options
    public bool showController = false;
    public List<GameObject> controllers;
    private GameObject spawnedController;

    //Device characteristics
    public InputDeviceCharacteristics controllerChar;

    //Hand model
    public GameObject handModel;
    private GameObject spawnedHandModel;

    private Animator handAnimator;

    //Get if is Right or left
    public string isRight;

    //Input variables
    public float trigVal;
    public bool primaryButton;
    public float gripValue;
    public Vector2 axisVal;
```

Slika 7: Clase Ručne kontrole

1. Na početku skripte možete vidjeti navedene varijable. One koje su javne mogu se mijenjati i djelovat će kao komponente.

2 Kada kreirate novu skriptu, Unity automatski generira dvije funkcije: Start, koja se pokreće samo jednom na početku kada se inicira komponenta , i Ažuriranje, koja se ponavlja uzastopno tijekom životnog ciklusa komponente.

3 Prva funkcija koja se pokreće unutar ove komponente je TryInitialize, koja identificira VR uređaj prema njegovim specifičnim karakteristikama, kao npr. je li kontroler desno ili lijevo, koristeći funkciju GetDevicesWithCharacteristics modula InputDevices.

```
//Try get input device and attach the selected model to it
void TryInitialize()
{
    //Get VR devices by characteristics
    List<InputDevice> devices = new List<InputDevice>();
    InputDevices.GetDevicesWithCharacteristics(controllerChar, devices);


    if (devices.Count > 0)
    {
        //Get target device and get a controller model
        targetDevice = devices[0];
        GameObject prefab = controllers.Find(con => con.name == targetDevice.name);

        if (prefab)
        {
            spawnedController = Instantiate(prefab, transform);
        }
        else
        {
            Debug.LogError("Did not find corresponding controller model");
            spawnedController = Instantiate(controllers[0], transform);
        }

        //Instantiate Hand Model and get Animator component
        spawnedHandModel = Instantiate(handModel, transform);
        handAnimator = spawnedHandModel.GetComponent<Animator>();
    }
}
```

Slika 8: TryInitialize funkcija

1 Zatim se izvršava funkcija Update, dozivajući funkcije UpdateEvents i UpdateAnimations.

2 UpdateEvents prikuplja vrijednosti koje šalju kontrole pomoću funkcije TryGetFeatureValue i pohranjuje ih u unaprijed označenim javnim varijablama.

3 UpdateAnimations je funkcija dizajnirana za pokretanje animacija koje odgovaraju svakom gumbu, u slučaju da imate namješteni model ruke. Iako ova funkcija nije ključna u zadaći dobivanja vrijednosti, ona pridonosi interaktivnosti i realnom prikazu radnji.

```
//Update hand events
void  UpdateEvents()
{
    //Trigger event
    if(targetDevice.TryGetFeatureValue(CommonUsages.trigger, out float triggerVal))
    {
        trigVal = triggerVal;
    }

    //PrimaryButton event
    if (targetDevice.TryGetFeatureValue(CommonUsages.primaryButton, out bool primary))
    {
        primaryButton = primary;
    }

    //Grip event
    if (targetDevice.TryGetFeatureValue(CommonUsages.grip, out float gripVal))
    {
        gripValue = gripVal;
    }

    //Joystick event
    if(targetDevice.TryGetFeatureValue(CommonUsages.primary2DAxis, out Vector2 axis))
    {
        axisVal = axis;
    }
}
```

Slika 9: Función UpdateEvents

4.4.3 Upravljanje viličarom

Za postizanje uvjerljivog i funkcionalnog iskustva virtualne stvarnosti pri upravljanju viličarom neophodno je imati niz detaljnih i specijaliziranih skripti. Ove skripte su odgovorne za pružanje precizne interaktivnosti i kontrole koji su potrebni za učinkovitu simulaciju rada ove vrste strojeva. Unutar VR alata bit će potrebna primarna skripta posvećena vožnji, koja će korisniku omogućiti manevriranje viličarom unutar virtualnog okruženja. Osim toga, bit će potrebno nekoliko dodatnih skripti usmjerenih na kontrolu poluga i drugih mehanizama kamiona, osiguravajući da se sve moguće radnje u stvarnoj operaciji mogu replicirati virtualno.

Zatim ćemo detaljno objasniti svaku od ovih skripti, istražujući kako rade, interakciju s elementima viličara i kako pridonose produbljenom i realističnom korisničkom iskustvu u rukovanju ovim strojevima u kontekstu virtualne stvarnosti .

- SkriptaCarController pruža interaktivno iskustvo vožnje u virtualnoj stvarnosti, gdje korisnik može izravno utjecati na ubrzanje vozila putem okidača kontrola, što se odražava u varijabli accelVal. Osim toga, skripta prilagođava smjer vozila mijenjanjem kuta kotača na temelju korisničkog unosa, koji se obrađuje pomoću varijable currentSteerAngle.

Kočnice se aktiviraju gumbom, koji utječe na varijablu isBreaking, a skripta upravlja fizikom sudara kako bi simulirala odskok kada je to potrebno. Ukratko, ovaj kod prevodi interakcije korisnika u mehanička ponašanja vozila unutar virtualnog okruženja.

```
// Update is called once per frame
void FixedUpdate()
{
    GetValues();

    if (!isBouncing)
    {
        foreach (WheelCollider wheel in wheels)
        {
            wheel.motorTorque = strength * Time.deltaTime * accelVal;

            wheel.wheelDampingRate = dampening;

            wheel.brakeTorque = isBreaking ? brakeStrength : 0;
        }

        foreach (var wheel in wheelsSteer)
        {
            WheelCollider collider = wheel.GetComponent<WheelCollider>();
            Transform trans = wheel.transform;

            collider.steerAngle = currentSteerAngle;
            collider.wheelDampingRate = dampening;
            //UpdateWheelVisual(collider, trans);
        }
    }

}
```

Slika 10: FixedUpdate funkcija

- Skripta RotateElement omogućuje korisniku upravljanje rotacijom dizala u okruženju virtualne stvarnosti pomoću ručnih kontrola. Kroz interakciju s okomitom ulaznom osi kontrolera, varijabla upForce prilagođava i određuje intenzitet i smjer rotacije toRotate objekta. S ugrađenim ograničenjima koja zaustavljaju rotaciju kada se dosegnu unaprijed definirana ograničenja, ova skripta osigurava gladak i kontroliran rad rotacijskog mehanizma dizala.

```
void FixedUpdate()
{
    UpdateForce();

    if (upForce > 0 && rotateUp)
    {
        float rotationAmount = 20f * Time.deltaTime * upForce;
        toRotate.localRotation *= Quaternion.AngleAxis(rotationAmount, Vector3.up);
        rotateDown = true;
    }
    else if (upForce < 0 && rotateDown)
    {
        float rotationAmount = 20f * Time.deltaTime * upForce;
        toRotate.localRotation *= Quaternion.AngleAxis(rotationAmount, Vector3.up);
        rotateUp = true;
    }
    }

void OnTriggerEnter(Collider col)
{
    if(col.gameObject.tag == "RotationLimit" && upForce > 0)
    {
        rotateUp = false;
    }
    else if(col.gameObject.tag == "RotationLimit" && upForce < 0)
    {
        rotateDown = false;
    }
}

void UpdateForce()
{
    if (inputs.Length == 0)
    {
        inputs = GameObject.FindGameObjectsWithTag("GameController");
    }
    else
    {
        foreach (var element in inputs)
        {
            HandController hand = element.GetComponent<HandController>();

            if (hand.isRight == "right" && grab.isGrabbing)
            {
                upForce = hand.axisVal.y * 0.5f;
            }
            else if(!grab.isGrabbing)
            {
                upForce = 0;
            }

        }
    }
}
```

Slika 11: Clase RotateElement

-

- Skripte MoveUp i MoveRetractil zajedno upravljaju okomitim kretanjem objekta u okruženju virtualne stvarnosti. Dok se MoveUp brine o izravnom okomitom kretanju objekta,

MoveRetractil kontrolira okomito kretanje povezane komponente, kao što je uvlačivi jarbol. Oba koriste upForce vertikalni unos sile, na koji utječe interakcija korisnika s kontrolerom, za određivanje smjera i intenziteta kretanja. Koordinacija između ove dvije skripte osigurava glatko i ograničeno vertikalno kretanje, pri čemu MoveRetractil također uzima u obzir sudare i prostorna ograničenja kako bi spriječio neželjena kretanja, kao što je dostizanje gornjeg ili donjeg graničnika dopuštenog kretanja.

```
// Update is called once per frame
void FixedUpdate()
{
    UpdateForce();

    if (goUp && upForce > 0)
    {
        transform.Translate(new Vector3(0, 0, 1f) * upForce * constForce);
        goDown = true;
    }
    else if (goDown && upForce < 0)
    {
        transform.Translate(new Vector3(0, 0, 1f) * upForce * constForce);
        goUp = true;
    }

}

void UpdateForce()
{
    if (inputs.Length == 0)
    {
        inputs = GameObject.FindGameObjectsWithTag("GameController");
    }
    else
    {
        foreach (var element in inputs)
        {
            HandController hand = element.GetComponent<HandController>();

            if (hand.isRight == "right" && grab.isGrabbing)
            {
                upForce = hand.axisVal.y * 0.5f;
            }
            else if (!grab.isGrabbing)
            {
                upForce = 0;
            }

        }
    }
}
```

Slika 12: Clase MoveUp

4.4.4 Upravljanje dizalicom

Kontrola dizalice u okruženju virtualne stvarnosti sastavni je dio interaktivnog iskustva, omogućujući korisnicima precizno i realistično upravljanje objektima unutar trodimenzionalnog prostora. Za kretanje mosta i kuke koriste se dvije bitne skripte:

CraneController i HookController. Prvi upravlja operacijama mosne dizalice, kao što je prenošenje u različitim smjerovima, dok se drugi bavi specifičnim pokretima kuke, olakšavajući spuštanje, penjanje i precizno pozicioniranje kuke. Dodatno, za hvatanje i rukovanje daskama i drugim predmetima koristi se skripta AttachTarget koja omogućuje učinkovito spajanje i odvajanje, osiguravajući da predmeti ostanu fiksirani tijekom transporta i glatko se otpuštaju na željeno odredište. Zajedno, ove skripte stvaraju kohezivan i učinkovit sustav upravljanja za rad mosne dizalice unutar simulacije..

- CraneController skripta je srž sustava za upravljanje mosnom dizalicom, koji omogućava bočno, okomito i frontalno kretanje motora, kabine i užadi dizalice. Koristi varijablu snage za određivanje intenziteta i smjera pokreta, dok stanje označava vrstu pokreta koji se izvodi. Booleove varijable kao što su isBack, isFront, isLeft, isRight, isDown i isUp djeluju kao granični prekidači kako bi spriječili pomicanje dizalice izvan svojih fizičkih ograničenja. S druge strane, HookController radi u tandemu s CraneControllerom, posebno nadzirući sudare s određenim oznakama za upravljanje i ograničavanje okomitog kretanja kuke, sprječavajući njeno pomicanje preko gornje i donje granice. Obje skripte osiguravaju nesmetan i siguran rad mostne dizalice, omogućujući korisniku precizno rukovanje teretom unutar virtualnog okruženja.
- Skripta AttachTarget brine o mehanici hvatanja i držanja objekata, poput dasaka, u okruženju virtualne stvarnosti. Koristi HingeJoint za simulaciju fleksibilne spojne točke između objekta i točke vezivanja, omogućujući kontrolirano kretanje klatna. Detektiranje blizine objekta meti i korisnički unos presudni su za aktiviranje stiska. Kada korisnik pritisne gumb za držanje i objekt je u ispravnom položaju, skripta aktivira teksturu vezice i konfigurira svojstva HingeJointa, postavljajući ograničenja i brzinu za simulaciju realnog kretanja. Ako korisnik otpusti gumb za hvatanje ili se objekt pomakne predaleko, skripta onemogućuje vezu, dopuštajući da se objekt otpusti na kontrolirani način. Osim toga, skripta komunicira s pločom s uputama kako bi vodila korisnika kroz proces rukovanja objektima

```
// Update is called once per frame
void FixedUpdate()
{
    //Check stop state and power value
    if (power != 0 && state != "Block")
    {
        //Lateral movement
        if (state == "left_right")
        {
            if (power > 0 && !isLeft)
            {
                motor.transform.Translate(new Vector3(1f, 0, 0) * power * constForce);
                isRight = false;
            }
            else if (power < 0 && !isRight)
            {
                motor.transform.Translate(new Vector3(1f, 0, 0) * power * constForce);
                isLeft = false;
            }
        }

        //Vertical movement
        if (state == "up_down")
        {
            if(power > 0 && !isUp)
            {
                ropes.transform.localScale = ropes.transform.localScale + new Vector3(0, 0, 1f) * (-power * constForce * scaleValue);

                foreach (var ele in gancho)
                {
                    ele.transform.Translate(new Vector3(0, 0, 1f) * power * constForce);
                }

                isDown = false;
            }
            else if (power < 0 && !isDown)
            {
                ropes.transform.localScale = ropes.transform.localScale + new Vector3(0, 0, 1f) * (-power * constForce * scaleValue);

                foreach (var ele in gancho)
                {
                    ele.transform.Translate(new Vector3(0, 0, 1f) * power * constForce);
                }

                isUp = false;
            }
        }

        //Frontal movement
        if (state == "back_front")
        {
            if (power > 0 && !isBack)
            {
                overhead.transform.Translate(new Vector3(0, 1f, 0) * power * constForce);
                isFront = false;
            }
            else if (power < 0 && !isFront)
            {
                overhead.transform.Translate(new Vector3(0, 1f, 0) * power * constForce);
                isBack = false;
            }
        }
    }
}
```

Slika 13: FixedUpdate funkcija CraneController class

4.4.5 Upitnik manager

Kako bi se optimizirao i pojednostavio proces kreiranja upitnika, osmišljena je klasa pod nazivom QuizManager koja standardizira izradu upitnika. Rad ove klase podijeljen je na sljedeće korake:

1 Varijable su definirane za pohranjivanje svih pitanja, trenutnog pitanja, točnog odgovora i elemenata korisničkog sučelja (UI).
2 Implementirana je funkcija InitQuiz čija je svrha započeti sve potrebne varijable i zatim pozvati SetNextQuestion za pokretanje kviza.

3 Funkcije SetNextQuestion i SetAnswerOptions odgovorne su za ažuriranje varijabli i komponenti ploče, kao što su tekstovi i gumbi, dodjeljivanje novog pitanja i odbacivanje prethodnog ako je potrebno.
4 Funkcija SelectButton(Button but) aktivira se kada se pritisne tipka za odgovor. Za pitanja s jednim odgovorom, ova funkcija će dovesti do izvršenja CheckResponse.
5 Funkcija CheckResponse sadrži logiku potrebnu za provjeru je li odabrani odgovor točan. Ako da, korisnik će prijeći na sljedeću ploču; u suprotnom, imati ćete priliku ponovno pokušati odgovoriti.

4.4.6 Prevoditelj

Ovaj proces djeluje na svaku scenu u pozadini kako bi svakom elementu korisničkog sučelja alata pružio tekstove prevedene na odabrani jezik.

Prvi korak je bio prijevod tekstova na jezike partnera koji su sudjelovali. Ti su prijevodi strukturirani u formatu u kojem je svaka riječ ili fraza povezana s jedinstvenim ključem, što olakšava dohvaćanje teksta na željenom jeziku. Za upravljanje ovim sustavom razvijene su dvije specifične klase:

1 MainTranslate: Ova klasa ima funkciju povezivanja prevoditeljskih skripti s operativnim okruženjem, tj. kodom. Kad god treba lokalizirati određeni ključ generiran u spomenutoj skripti, MainTranslate će pružiti odgovarajuću riječ ili izraz na ispravnom jeziku.
2 SceneTranslate: Zadaća ove klase je komunikacija MainTranslate-a, na početku svake scene, koji je jezik odabran u izborniku opcija.

Dok su ovi procesi aktivni, jedini zahtjev za korisnika je da svaki redak teksta koji želi umetnuti zamijeni sljedećim redkom koda:

```
text = MainTranslate.Fields[textKey];
```

Slika 14: Redak za prijevod.

4.4.7 Interakcija sa svjetlom

Ova funkcija omogućuje korisniku interakciju s objektima ili pločama koje se nalaze na određenoj udaljenosti. To se postiže stvaranjem virtualne zrake koja dolazi iz korisnikove ruke, omogućujući mu da izvodi određene radnje na određenim objektima. U kontekstu ovog projekta ključno je da korisnik može odgovoriti na pitanja postavljena dok se koristi ovom tehnikom. Za implementaciju ove vrste interakcije potrebno je slijediti sljedeće korake:

1 Generirajte Ray Interactor objekt za svaku ruku, kojem možete pristupiti putem izbornika GameObject/XR.

2 Odaberite objekte s kojima ćete komunicirati podešavanjem parametra Raycast Mask komponente XR Ray Interactive. Ova vam postavka omogućuje odabir sloja na koji će zraka djelovati, stoga objekti namijenjeni interakciji sa zrakom moraju biti dodijeljeni odgovarajućem sloju.
3 U ovom scenariju smo odabrali sloj korisničkog sučelja za panele koje nudi Unity..
4 Kako biste spriječili da svjetlost bude stalno vidljiva, potrebno je modificirati atribut Invalid Color Gradient komponente XR Interactor Line Visual na transparentnu vrijednost. Na ovaj način, zraka će postati vidljiva samo kada ruka pokazuje na ploču.
5 Oba objekta bit će ugrađena kao povezani djelovi Camera Offset, koji je prethodno kreiran u fazi pripreme scene, čime se osigurava da su dio objekta povezanog s VR uređajem.

Implementacijom ovih koraka postiže se funkcionalnost potrebna za generiranje i interakciju s različitim gore opisanim nadzornim pločama.

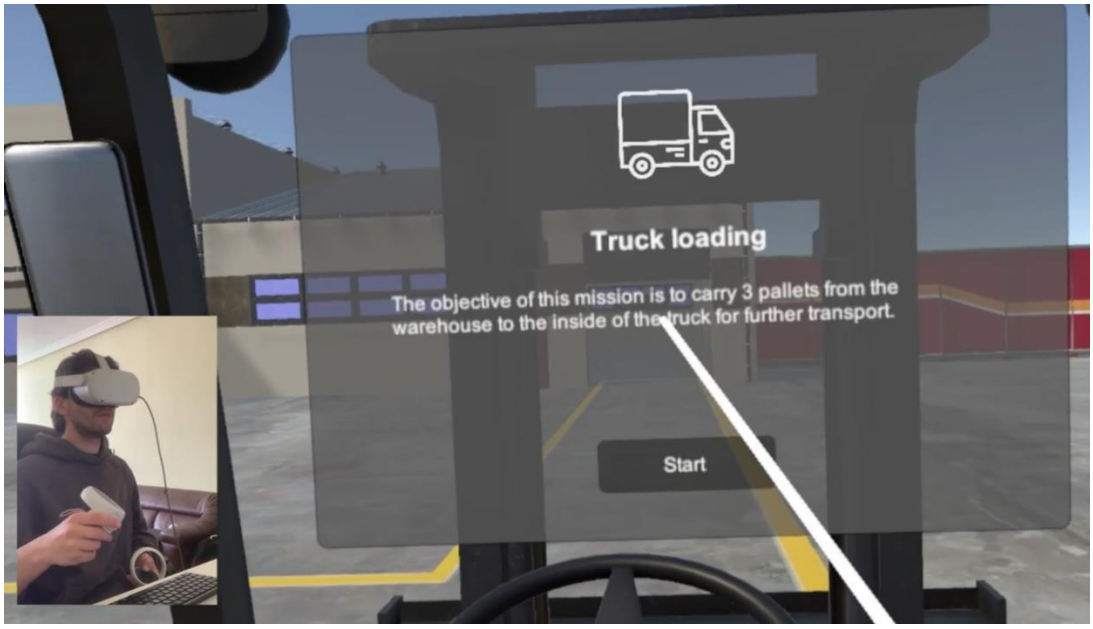


Figure 15: Interakcija sa svijetlom

5 YOUTUBE VIDEO ZAPISI

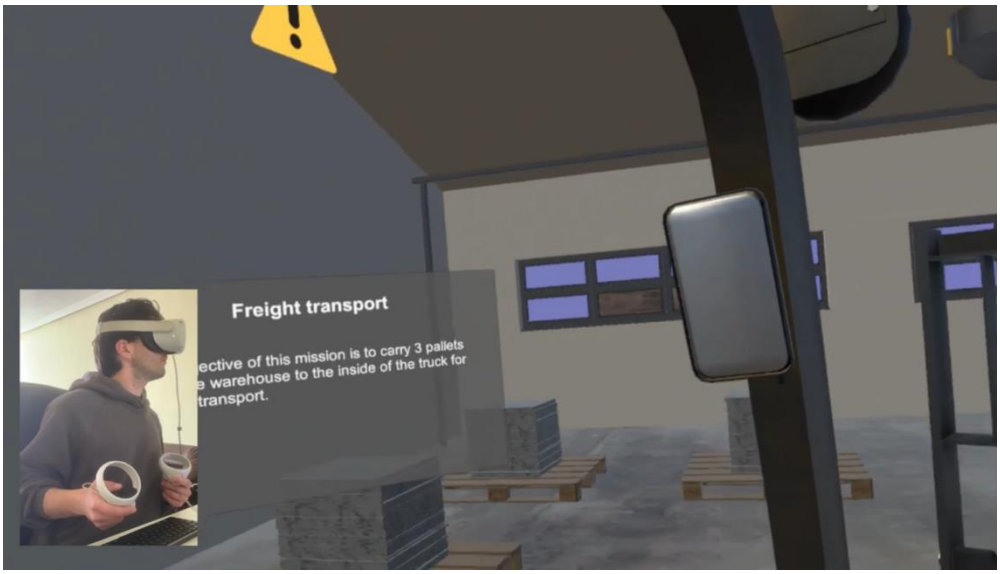
Kako bismo nadopunili papir i ponudili dinamičniju perspektivu našeg VR alata, odabrali smo niz YouTube videozapisa koji pokazuju njegovu funkcionalnost. U nastavku predstavljamo ove audiovizualne materijale koji pružaju jasan i izravan uzorak izvedbe i mogućnosti koje nudi naše VR rješenje.

- Promotion: https://www.youtube.com/watch?v=Ogs2WzzCRe0&ab_channel=AEIPiedraNatural

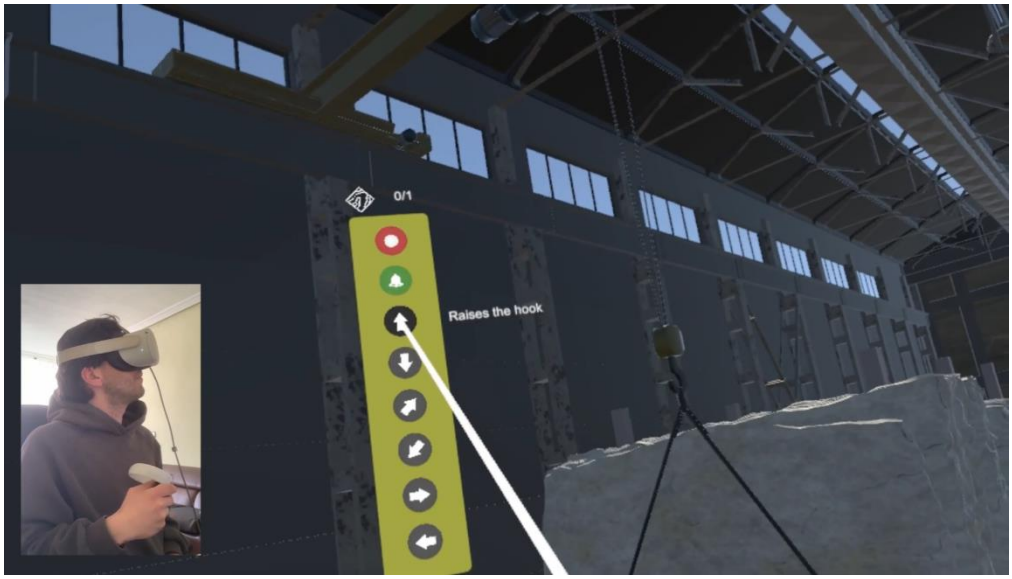
- Main Menu: https://www.youtube.com/watch?v=I7j9rTMeWmo&ab_channel=AEIPiedraNatural

- Forklift – Storage: https://www.youtube.com/watch?v=hCkCl9ihiLU&t=14s&ab_channel=AEIPiedraNatural

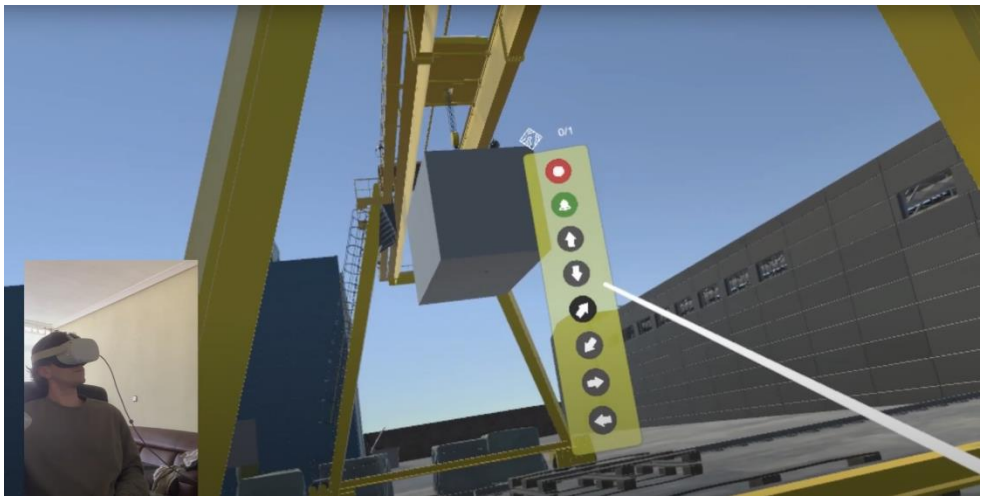
- Forklift – Truck loading: https://www.youtube.com/watch?v=NnpA44V6DMY&t=13s&ab_channel=AEIPiedraNatural



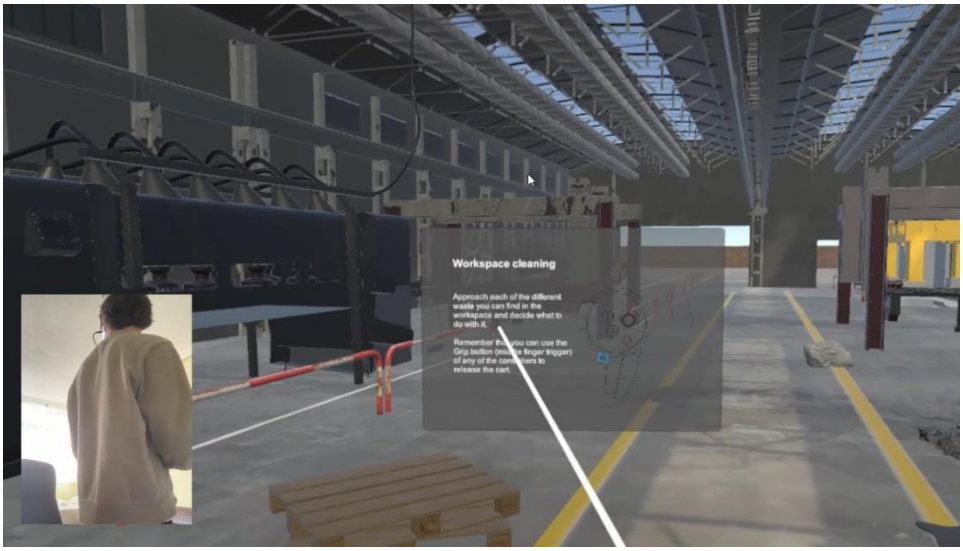
- Overhead Crane – Slabs: https://www.youtube.com/watch?v=LK9pSCPLMrw&ab_channel=AEIPiedraNatural



- Overhead Crane – Blocks: https://www.youtube.com/watch?v=tVyFFRJyQ4U&ab_channel=AEIPiedraNatural



- cleaning: hattops://www.youtube.com/watch?v=twiffarjyak4u&ab_channel=apedarentural



- Waste management: https://www.youtube.com/watch?v=mojMZ2G6Huc&ab_channel=AEIPiedraNatural