Kauno technologijos universitetas

Informatikos fakultetas

Virtualios realybės ir dirbtinio intelekto metodai kognityvinių įgūdžių ir judesių lavinimui

Baigiamasis magistro projektas

Donata Gliaubičiūtė

Projekto autorė

Prof., dr. Tomas Blažauskas

Vadovas

Kaunas, 2023

Kauno technologijos universitetas

Informatikos fakultetas

Virtualios realybės ir dirbtinio intelekto metodai kognityvinių įgūdžių ir judesių lavinimui

Baigiamasis magistro projektas

Dirbtinio intelekto informatika (6211BX007)

Donata Gliaubičiūtė

Projekto autorė

Prof., dr. Tomas Blažauskas

Vadovas

Prof., dr. Alfonsas Misevičius

Recenzentas

Kaunas, 2023

Kauno technologijos universitetas

Informatikos fakultetas

Baigiamojo magistro projekto užduotis

Projekto tema	Virtualios realybės ir dirbtinio intelekto metodai kognityvinių įgūdžių ir judesių lavinimui
Reikalavimai ir sąlygos (tikslinti pavadinimą pagal poreikį)	
Vadovas	Prof., dr. Tomas Blažauskas 2023-05-22
	(vadovo pareigos, vardas, pavardė, parašas) (data)

Donata Gliaubičiūtė. „Virtualios realybės ir dirbtinio intelekto metodai kognityvinių įgūdžių ir judesių lavinimui“. Magistro studijų baigiamasis projektas. Vadovas prof., dr. Tomas Blažauskas; Kauno technologijos universitetas, informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, Informatika (B01).

Reikšminiai žodžiai: reabilitacija, virtualioji realybė, neuroniniai tinklai, žaidimų variklis, *Unreal Engine*, žmogaus judesių sekimas.

Kaunas, 2023. 103 p.

Santrauka

Paskutiniu metu didėja domėjimasis naujais, reabilitacijos proceso gerinimui skirtais metodais, tokiais kaip virtualioji ir papildytoji realybė. Tradicinės reabilitacijos teikimas gali būti brangus ir sunkiai pasiekiamas žmonėms gyvenantiems atokiuose regionuose ar neturintiems galimybės atvykti į reabilitacijos centrus. Virtualiosios realybės reabilitacijos sistemos gali būti prieinamos pacientų namuose ir naudojamos savarankiškai be sveikatos priežiūros specialistų pagalbos. Šios sistemos taip pat gali būti pritaikytos kasdienio gyvenimo veikloms ir gyvenimo kokybės gerinimui.

Rankų judesių sekimo technologijos taikymas virtualioje realybėje gali leisti VR sistemų naudotojams sąveikauti su virtualiais objektais ir situacijomis valdant jų pačių rankas. Duomenys surinkti su judesių sekimo technologijomis gali būti naudojami žmogaus veiklai atpažinti taikant dirbtinio intelekto metodus.

Šiame darbe buvo iškeltas tikslas pasiūlyti ir ištirti dirbtinio intelekto metodų taikymą kognityvinių įgūdžių ir judesių lavinimui naudojant virtualiąją realybę bei judesių sekimo technologijas. Sukurta virtualiosios realybės reabilitacijos sistema naudoja konvoliucinio neuroninio tinklo modelį naudotojo judesių į atitinkamas veiklos klases priskyrimui.

Atliktas tyrimas, ištyrė pritaikytų dirbtinio intelekto metodų tikslumą, judėjimo sekimo technologijų įtaką pritaikytiems dirbtinio intelekto metodams ir sukurtos sistemos panaudojamumą. Tyrimas parodė, kad sukurto modelio tikslumo įvertinimas vidutiniškai 2,28 % buvo geresnis už kitų tyrime naudotų klasifikavimo modelių. Tyrimo metu buvo nustatyta, kad *Meta Quest 2* įrenginio valdiklių taikymas gali lemti 2,68 % aukštesnį modelio tikslumą, nei taikant *Leap Motion Controler* įrenginį. Tyrime buvo nustatytas sukurtos sistemos panaudojamumas nuo „geras“ iki „puikus“ *SUS* skalės ribose.

Donata Gliaubičiūtė. „A study on virtual reality and artificial intelligence application to cognitive skills and movement training“. Master's Thesis Project. Supervisor prof., dr. Tomas Blažauskas; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Computer science, Informatics (B01).

Keywords: rehabilitation, virtual reality, neural networks, game engine, Unreal Engine, human motion tracking.

Kaunas, 2023. 103 pages.

Summary

Recently, there has been a growing interest in new methods for improving the rehabilitation process, such as virtual and augmented reality. The provision of traditional rehabilitation can be expensive and difficult to reach for people living in remote regions or who do not have the opportunity to come to rehabilitation centers. Virtual reality rehabilitation systems can be accessed at patients' homes and used independently without the assistance of healthcare professionals. These systems can also be adapted to activities of daily living and improving quality of life.

The application of hand motion tracking technology in virtual reality can allow users of VR systems to interact with virtual objects and situations by controlling their own hands. Data collected with motion tracking technologies can be used to recognize human activity using artificial intelligence methods.

The goal of this work was to propose and investigate the application of artificial intelligence methods for the training of cognitive skills and movements using virtual reality and motion tracking technologies. The developed virtual reality rehabilitation system uses a convolutional neural network model to assign the user's movements to the appropriate activity classes.

The study investigated the accuracy of the applied artificial intelligence methods, the influence of motion tracking technologies on the applied artificial intelligence methods and the usability of the developed system. The study showed that the accuracy rating of the developed model was on average 2.28% better than other existing classification models. The study found that applying *Meta Quest 2* VR headset controllers can result in 2.68% higher model accuracy than using the *Leap Motion Controller* device. The study determined the usability of the developed system from "good" to "excellent" within the limits of the *SUS* scale.

Turinys

Lentelių sąrašas 8
Paveikslų sąrašas 9
Santrumpų ir terminų žodynas 11
Įvadas 12
1. Judesių atpažinimui taikomų algortimų ir metodų analizė 14
1.1. Viršutinių galūnių judėjimo ir kognityvinių įgūdžių lavinimo metodai 14
1.1.1. Motorinio mokymosi ir kontrolės principų taikymas reabilitacijoje 14
1.1.2. Kognityvinių įgūdžių reabilitacija 15
1.1.3. Reabilitacija taikant virtualią realybę 15
1.2. Dirbtinio intelekto metodai žmogaus veiklai atpažinti 16
1.2.1. Konvoliuciniai neuroniniai tinklai 16
1.2.2. Pasikartojantys neuroniniai tinklai 21
1.3. Duomenų tvarkymo metodai 24
1.3.1. Slankiojantis langas 24
1.3.2. Normalizavimas ir standartizavimas 26
1.4. Rankų sekimo technologijos 28
1.4.1. Optinis sekimas 28
1.4.2. *LiDAR* sekimas 29
1.5. Virtualios realybės formos 30
1.6. Virtualioje realybėje naudojami judesių sekimo metodai 30
1.6.1. “Iš vidaus į išorę“ sekimo metodas 31
1.6.2. “Iš išorės į vidų“ sekimo metodas 31
1.7. Analizės apibendrinimas 31
2. VRR sistemos projektavimas 32
2.1. Programos vystymo etapai 32
2.2. Sistemos kontekstas 32
2.3. Programos kokybės vertinimas 33
2.4. Mašininio mokymosi modelio vertinimo metrikos 33
2.5. Apribojimai 34
2.5.1. Bendri apribojimai 34
2.5.2. Naudojami programinės įrangos paketai 34
2.6. Veiklos sudėtis 35
2.6.2. Sistemos panaudojamumo ir išvaizdos reikalavimai 35
2.7. Funkciniai ir nefunkciniai reikalavimai 35
2.8. Sistemos sudėtis 36
2.8.1. Sistemos ribos ir panaudojimo atvejai 36
2.8.2. Panaudojimo atvejų sąrašas 36
2.9. Reikalavimai duomenims 38
2.10. VRR programos dinaminis vaizdas 39
3. Sistemos kūrimo priemonės ir metodai 40
3.1. Vartotojo sąsajos grafika 40
3.1.1. 3D modeliavimas 40
3.1.2. Optimizacijos metodų taikymas 41
3.1.3. 3D objektų medžiagos 41
3.1.4. Modelių paruošimas žaidimų varikliui 42

3.1.5. Kolizijos geometrijų kūrimas 42
3.2. 3D scenos kūrimas 43
3.3. Naudotojo objekto įvestis 44
3.3.1. Rankų sąveikos mechanizmas 44
3.3.2. Sąveika su aplinkos objektais 45
3.3.3. Judesių registravimas ir projektavimas į dvimatį vaizdą 47
3.4. Duomenų rinkinys 52
3.5. Konvoliucinių neuroninių tinklų architektūrų kūrimas 53
3.6. Modelių parametrų derinimas 56
3.7. Duomenų siuntimas 65
4. Virtualios realybės ir dirbtinio intelekto metodų taikymo tyrimas 67
4.1. Tyrimo priemonės 67
4.1.1. Neuroninių tinklų architektūros 67
4.1.2. Reabilitacijos pratimas 68
4.1.3. Rankų judesių sekimo technologijos 69
4.1.4. Klausimynai 69
4.2. Tyrimo eiga 69
4.2.1. Neuroninių tinklų modelių apmokymas 70
4.2.2. Eksperimentai su VRR sistema 70
4.2.3. Duomenų apdorojimas 71
4.3. Tyrimo rezultatai 71
4.3.1. Neuroninių tinklų modelių įvertinimas 71
4.3.2. Rankų judesių sekimo technologijų taikymo su VRR sistema rezultatai 73
4.3.3. VRR sistemos vertinimas pagal klausimyno rezultatus 75
4.4. Tyrimo apibendrinimas 77
Išvados 78
Literatūros sąrašas 79
Informacijos šaltinių sąrašas 85
Priedai 87
1 priedas. „Influence of Aerial Image Resolution on Vehicle Detection Accuracy“ 87
2 priedas. „A System for Providing Educational Content in Virtual Reality“ 97

Lentelių sąrašas

1.1 lentelė. Aktyvinimo funkcijų grafikai, formulės ir reikšmių aibės [R3] 18
1.2 lentelė. Normalizavimo metodų palyginimas 27
1.3 lentelė. Normalizavimo metodų privalumai ir trūkumai [55] 27
2.1 lentelė. VRR programos vystymo etapai 32
2.2 lentelė. Kokybės vertinimas 33
2.3 lentelė. Veiklos įvykių sąrašas 35
2.4 lentelė. Atliekamo pratimo stebėjimo panaudos atvejo aprašymas 37
2.5 lentelė. Pratimo atlikimo įvertinimo peržiūros panaudos atvejo aprašymas 37
2.6 lentelė. Judėjimo aplinkoje panaudos atvejo aprašymas 37
2.7 lentelė. Sąveikavimo su aplinkos objektais panaudos atvejo aprašymas 38
2.8 lentelė. Pratimo atlikimo panaudos atvejo aprašymas 38
3.1 lentelė. *EMNIST* rinkinį išskaidančių duomenų rinkinių struktūra [R15] 52
3.2 lentelė. Pirmo konvoliucinio neuroninio tinklo modelio architektūros struktūra 54
3.3 lentelė. Antro konvoliucinio neuroninio tinklo modelio architektūros struktūra 55
3.4 lentelė. Trečio konvoliucinio neuroninio tinklo modelio architektūros struktūra 55
3.5 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingus partijos dydžius 57
3.6 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingas aktyvinimo funkcijas 58
3.7 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingus optimizatorius 59
3.8 lentelė. Modelių įvertinimas taikant skirtingus *ImageDataGenerator* klasės objektus 61
3.9 lentelė. Modelių parametrų derinimo rezultatai 62
4.1 lentelė. Modelių treniravimo metu taikyti parametrai 70
4.2 lentelė. *NASA TLX* klausimyno rezultatų interpretavimas 71
4.3 lentelė. Modelių rezultatai papildytam *EMNIST Balanced* duomenų rinkiniui 72

Paveikslų sąrašas

1.1 pav. Akselerometro surinkti laiko eilučių duomenys [R1] 17
1.2 pav. Dvimačio paveikslo duomenų konvoliucijos operacija [R2] 18
1.3 pav. Maksimizavimo ir vidurkinimo apjungimo metodai [R4] 21
1.4 pav. LSTM tinklo ląstelės sandara [R5] 22
1.5 pav. Sistemos planas žmogaus veiklai atpažinti [R6] 24
1.6 pav. „Slankiojančio lango“ duomenų paruošimo metodas [R7] 25
1.7 pav. *Pimax Vision 8K X* virtualiosios realybės akiniai ir jų optinio sekimo įrenginys [R8] . 29
1.8 pav. *Leap Motion Controller* naudojimo pavyzdys ir jį sudarantys elementai [R9] 29
1.9 pav. Virtualios realybės akinių *Varjo XR–3* komponentai [R10] 30
1.10 pav. Sekimo metodų „iš vidaus į išorę“ ir „iš išorės į vidų“ taikymas VR [R11] 31
2.1 pav. Sistemos veikimo kontekstas 32
2.2 pav. Maišos matricos struktūra 33
2.3 pav. Veiklos kontekstas 35
2.4 pav. VRR sistemos panaudojimo atvejų diagrama 36
2.5 pav. Duomenų klasių diagrama 39
2.6 pav. Pratimo įvertinimo formavimo veiklos diagrama 39
3.1 pav. Trimačiai modeliai *Blender* vartotojo sąsajoje 40
3.2 pav. 3D modelio tekstūrų paveikslų atlasas 41
3.3 pav. 3D modelis ir jo UV išklotinė patalpinta ant tekstūrų paveikslų atlaso 42
3.4 pav. VRR programos scena 43
3.5 pav. “M_Water” medžiagos mazgų sistema 43
3.6 pav. „HandsPawn_BP“ klasę sudarantys komponentai 44
3.7 pav. Naudotojo rankų atvaizdavimas 45
3.8 pav. „HandsPawn_BP“ klasės įvykių grafikas 45
3.9 pav. „IsHandClenched“ funkcija 46
3.10 pav. „Set Simulate Physics“ ir „Attach Component To Component“ funkcijų mazgai 46
3.11 pav. „PointTracker“ komponento įvykių grafikas 48
3.12 pav. „FinishRegistration“ įvykio grafikas 48
3.13 pav. „PointRegPawn_BP“ klasės „Start Left“ funkcija 49
3.14 pav. „PointRegPawn_BP“ klasės „Finish Left“ funkcija 49
3.15 pav. „MotionControllerPawn_BP“ klasės įvykių grafikas 50
3.16 pav. Scenos objektas, kuris naudoja „DrawingPlane_BP“ klasę 50
3.17 pav. „Get Projection Transform“ funkcija 51
3.18 pav. „PointProcessor_BP“ klasės įvykių grafikas 52
3.19 pav. *EMNIST Balanced* rinkinio klasių pavadinimai ir duomenų pasiskirstymas [R16] 53
3.20 pav. Papildyto *EMNIST Balanced* rinkinio duomenų pavyzdžiai 53
3.21 pav. Modelių tikslumo ir treniravimui naudoto partijos dydžio priklausomybės diagrama .. 58
3.22 pav. Modelių tikslumo ir skirtingų aktyvinimo funkcijų taikymo priklausomybės diagrama 59
3.23 pav. Modelių tikslumo ir optimizatorių taikymo priklausomybės diagrama 60
3.24 pav. *ImageDataGenerator* klasės objekto su atitinkamais parametrais taikymo pavyzdys .. 61
3.25 pav. Modelių tikslumo ir *ImageDataGenerator* klasės objekto priklausomybės diagrama .. 62
3.26 pav. Modelių treniravimo tikslumo reikšmių pasikeitimo grafikas 63
3.27 pav. Modelių treniravimo praradimo reikšmių pasikeitimo grafikas 63
3.28 pav. Modelių validacijos tikslumo reikšmių pasikeitimo grafikas 63
3.29 pav. Modelių validacijos praradimo reikšmių pasikeitimo grafikas 64

3.30 pav. Modelio „model_1“ klasifikavimo rezultatų maišos matrica 65
3.31 pav. „TexturePlane_BP“ klasės įvykių grafikas 66
4.1 pav. *ResNet* tinklo liekamųjų blokų struktūros [R17] 68
4.2 pav. Modelių tikslumo treniravimo duomenų rinkiniui reikšmių pasikeitimo grafikas 72
4.3 pav. Modelių tikslumo validacijos duomenų rinkiniui reikšmių pasikeitimo grafikas 72
4.4 pav. Eksperimentų metu CNN modelio teisingai atpažintų klasių pasiskirstymo grafikas ... 73
4.5 pav. Eksperimentų metu CNN modelio neteisingai atpažintų klasių pasiskirstymo grafikas 74
4.6 pav. Simbolių priklausymo tam tikrai klasei tikimybių pasiskirstymo grafikas 75
4.7 pav. *SUS* klausimyno vertinimo skalė [R18] 76
4.8 pav. *IPQ* klausimyno rezultatai 76

Santrumpų ir terminų žodynas

Santrumpos:

VR	virtualioji realybė (angl. *Virtual reality*).
VRR	virtualiosios realybės reabilitacija (angl. *Virtual reality rehabilitation*).
CNN	konvoliucinis neuronins tinklas (angl. *Convolutional neural network*).
RNN	pasikartojantis neuroninis tinklas (angl. *Recurrent neural networks*).
LSTM	ilgalaikės trumpalaikės atminties tinklas (angl. *Long short–term memory network*).
ReLU	rektifikuoto linijinio vieneto funkcija (angl. *The rectified linear unit*).
LiDAR	šviesos aptikimo ir išdėstymo technologija (angl. *Light detection and ranging*).
3D	Trimatis (angl. *Three–dimensional*).
2D	Dvimatis (angl. *Two–dimensional*).

Terminai:

Mašininis mokymasis	dirbtinio intelekto mokslo šaka, kurioje daugiausia dėmesio yra skiriama duomenų ir algoritmų naudojimui, siekiant imituoti žmonių mokymosi būdą ir palaipsniui didinti jo tikslumą (angl. *Machine learning*).
Gilusis mokymasis	mašininio mokymosi metodų šaka, pagrįsta dirbtiniais neuroniniais tinklais (angl. *Deep learning*).
Virtuali realybė	kompiuteriu sugeneruota interaktyvi trimatė aplinka (angl. *Virtual reality*).
Akselerometras	įrenginys, kuris matuoja pagreitį.
Giroskopas	įrenginys, kuris matuoja kampinį greitį ir pasisukimo kampą erdvėje.
Išklotinių žemėlapis	3D objekto geometrijos išskleidimo rezultatas pavaizduotas 2D paveiklse (angl. *UV map*).
Tekstūra	3D objekto medžiagos informacija, kuri yra saugoma 2D paveiksle (angl. *Texture*).
Medžiaga	3D objekto tekstūros, spalvos, metališkumo, šiurkštumo, blizgumo ir kitų parametrų rinkinys (angl. *Material*).
Žaidimų variklis	programinės įrangos kūrimo sistema (angl. *Game engine*).

Įvadas

Virtualioji realybė gali būti apibūdinama kaip įtraukiantis, interaktyvus, daugelio jutimų, orientuotas į žiūrovą, kompiuterių sukurtos trijų dimensijų aplinkos ir technologijų, reikalingų šioms aplinkoms sukurti, derinys [1]. Vienas iš virtualios realybės kūrimo tikslų yra suteikti naudotojui tokią interaktyvią erdvę, su kuria jis galėtų bendrauti ir priimti ją kaip artimą realiajai. Šiais laikais virtualioji realybė daugeliui yra vis labiau prieinama, todėl jos panaudojimo sritis taip pat plečiasi. Medicinoje VR technologijos jau yra naudojamos diagnostikoje, pacientų konsultavimui, ligoninių projektavimui ir net gydymui. Gydytojai ir chirurgai dažniausiai naudoja VR kaip imitacijos priemonę. Medicinos studentai per VR gali įgyti žinių apie žmogaus kūną, praktikuotis atlikti procedūras ir operacijas virtualiems pacientams saugioje, kontroliuojamoje aplinkoje. Mokymosi procesas ir įgūdžių lavinimas tokiu būdu yra atliekamas greičiau ir be rizikos pakenkti realiems pacientams [2].

Virtualiosios realybės technologijos sparčiai populiarėja kognityvinių įgūdžių reabilitacijoje ir lavinant motorinę kontrolę. Kiekvienais metais yra atliekami įvairūs tyrimai ištirti šios technologijos įtaką pacientų nervų sistemai, taip pat keliami klausimai, ar judėjimas virtualioje aplinkoje gali motyvuoti asmenį atlikti reabilitacijos pratimus. Virtualiosios realybės reabilitacijos (angl. *Virtual Reality Rehabilitation – VRR*) programos leidžia pacientui praktikuoti tam tikrą elgesį, sąveikaujant su kompiuterio aplinka, kuri imituoja fizinį buvimą realiame ar įsivaizduojamuose pasauliuose [3].

Vis daugiau atsiranda VRR programų, skirtų žmonėms, kurie turi neurologinių pažeidimų, sutrikusį galūnių judėjimo valdymą ar motorinius įgūdžius. Norint sėkmingai atstatyti prarastus įgūdžius ar funkcijas, reabilitacijos proceso metu pacientas turi būti treniruojamas kiek įmanoma arčiau realios aplinkos. Jis turi galėti stebėti savo sveikimo procesą, užduočių atlikimo kokybiškumą, gali kartoti užduotis, kol atliks jas teisingai. VRR programos gali suteikti galimybę pacientams perkelti fizinio pasaulio sudėtingumą į kontroliuojamą aplinką. VR leidžia sukurti sintetinę aplinką, tiksliai kontroliuojant daugybę fizinių kintamųjų, kurie daro įtaką elgesiui ir atlikti fiziologinių ir kinematinių atsakų registravimą [4].

Nuolat tobulėjant mašininio mokymosi technologijai, mašininio mokymosi metodai turi plačias taikymo perspektyvas medicinos srityje. Dirbtinio intelekto algoritmai gali būti pritaikomi judesių atpažinimui, panaudojant virtualiosios realybės technologijas. Naudojant mašininio mokymosi metodus ir dirbtinį intelektą, VRR programos gali atpažinti paciento judesius jam atliekant pratimus, rinkti, analizuoti duomenis, įvertinti paciento pasiektą pažangą [5].

Tyrimo objektas – judesių atpažinimas panaudojant virtualiosios realybės technologijas ir dirbtinį intelektą.

Darbo tikslas – pasiūlyti ir ištirti AI metodų taikymą kognityvinių įgūdžių ir judesių lavinimui naudojant virtualiąją realybę ir judesių sekimo technologijas.

Darbo uždaviniai:

1. išanalizuoti realybėje taikomus judesių ir kognityvinių įgūdžių reabilitacijos metodus;
2. išanalizuoti dirbtinio intelekto algoritmus, kurie gali būti pritaikomi žmogaus judesiams atpažinti;
3. išanalizuoti žmogaus judesių sekimo technologijas, kurios gali būti taikomos virtualios realybės sistemoms;
4. sukurti programinę įrangą, teikiančią judesių lavinimo pratimus;
5. pasirinkti ir pritaikyti dirbtinio intelekto metodus judesiams atpažinti;

6. ištirti pritaikytų dirbtinio intelekto metodų tikslumą;
7. ištirti judėjimo sekimo technologijų įtaką pritaikytiems dirbtinio intelekto metodams;
8. ištirti sistemos panaudojamumą.

Mokslinis naujumas

Atlikus literatūros analizę buvo pastebėta, kad daugelis egzistuojančių virtualiosios realybės reabilitacijos programų reabilitaciją realizuoja taikant judesių atkartojimo pratimus. Šiame projekte yra aprašomas inovatyvus priėjimas prie šios problemos, kuris reabilitacijos realizavimui naudoja paveikslų piešimą virtualioje realybėje ir įvertinimo bei grįžtamojo ryšio suteikimą taikant naudotojo sukurtų vaizdų atpažinimą. Projekte buvo ištirtas ir pasiūlytas konvoliucinio neuroninio tinklo modelis, kuris įvertina virtualioje realybėje rankomis nupiešto paveikslo atitikimą šablonui.

Darbo struktūra

Šį dokumentą sudaro analizės, projektavimo, kūrimo, tyrimo ir išvadų skyriai. Analizės skyriuje yra aprašomi su projekto tema susiję viršutinių galūnių judėjimo ir kognityvinių įgūdžių lavinimo metodai, dirbtinio intelekto metodai žmogaus veiklai atpažinti, duomenų tvarkymo metodai, rankų sekimo technologijos ir virtualioje realybėje naudojami judesių sekimo metodai. Dokumento projektavimo skyriuje yra pateikiamas sistemos funkcionalumas, apribojimai, reikalavimai ir dinaminis sistemos vaizdas. Kūrimo skyriuje yra aprašomas sistemos kūrimo procesas ir pritaikyti metodai. Tyrimo skyriuje yra aprašomos atlikto tyrimo priemonės, eiga ir gauti rezultatai. Bendri darbo rezultatai apibendrinami išvadose.

1. Judesių atpažinimui taikomų algortimų ir metodų analizė

Šiame skyriuje yra aprašomi viršutinių galūnių judėjimo ir kognityvinių įgūdžių lavinimo metodai, dirbtinio intelekto metodai žmogaus veiklai atpažinti, viršutinių galūnių judėjimo sekimo technologijos ir virtualiosios realybės įrenginių naudojami judėjimo sekimo metodai.

1.1. Viršutinių galūnių judėjimo ir kognityvinių įgūdžių lavinimo metodai

Šiame poskyryje yra analizuojami metodai, kurie gali būti taikomi kognityvinių gebėjimų, pavyzdžiui, dėmesio, atminties ir problemų sprendimo įgūdžių, lavinime. Poskyryje analizuojami metodai taip pat gali pagerinti motorinę kontrolę, bendrą gyvenimo kokybę ir savarankiškumą kasdienėje veikloje.

1.1.1. Motorinio mokymosi ir kontrolės principų taikymas reabilitacijoje

2014 metais *Stony Brook, Thomas Jefferson* ir *Cardiff* universitetų mokslininkų bendras straipsnis apžvelgia pagrindinius motorinio mokymosi ir motorinės kontrolės principus, kurie gali būti naudojami viršutinių galūnių reabilitacijoje skatinant naujų įgūdžių įgijimą ar jų atgavimą po praradimo [6]:

- kartojimas;
- grįžtamasis ryšys;
- užduočių specifiškumas;
- intensyvumas ir sudėtingumas (angl. *Intensity and complexity*);
- kontekstiniai įsikišimai (angl. *Contextual interference*).

Reabilitacijos pratimai taikant kartojimo principą nurodo tam tikro judesio su pažeista viršutine galūne pasikartojantį atlikimą. Šios rūšies reabilitacijos pratimų rezultatai parodo, kad intensyvi struktūrinė praktika pagerina judesių kokybę, atlikimo laiką ir sukuria smegenų neurologinių substratų (angl. *Neuro substrates*) pokyčius, kurie lemia geresnes judėjimo galimybes [7]. Judesių kartojimas taip pat padeda pagerinti raumenų jėgą, koordinaciją ir kontrolę.

Reabilitacijos metu pastebėjimų ir komentarų teikimas gali padėti pacientui suprasti, kaip jis atlieka pratimą ir nustatyti sritis, kurias reikia tobulinti. Grįžtamąjį ryšį gali teikti terapeutas arba įvairios technologijos, pavyzdžiui, jutikliai ir biologinio grįžtamojo ryšio prietaisai (angl. *Biofeedback devices*), tokie kaip judėjimo, raumenų veiklos, jėgos, spaudimo matavimo prietaisai ir virtualiosios realybės sistemos [8].

Užduoties specifiškumo principas yra paremtas idėja, kad motorinio mokymosi ir kontrolės įtakai turi specifiniai užduočių reikalavimai. Praktikuojant judesius, būdingus veiksmams, kuriuos žmogus nori atlikti savo kasdieniame gyvenime, smegenys gali geriau prisitaikyti ir išmokti tam tikrų motorinių modelių, reikalingų veiksmams atlikti. Reabilitaciniai pratimai, kurie yra pritaikyti prie asmens tikslų ir poreikių, ne tik pagerina judesių mokymąsi ir kontrolę, bet ir labiau motyvuoja bei įtraukia pacientą. Sutelkus dėmesį į pacientui svarbias užduotis, reabilitacijos pratimų programa gali būti prasmingesnė ir labiau atitikti paciento tikslus bei poreikius [9].

Pagal intensyvumo principą didelio intensyvumo treniruočių tikslas yra per trumpą laiką atlikti daug pratimų arba vieno pratimo pakartojimų. Taikant sudėtingumo principą yra svarbu į reabilitacijos procesą įtraukti įvairius skirtingus pratimus, nes taip galima paskatinti pacientą pritaikyti naujus įgūdžius kituose pratimuose ar situacijose [10].

Kontekstinių įsikišimų principo taikymas suteikia nenuspėjamumo jausmą reabilitacijos užsiėmimams. Šis principas apima įvairių pratimų atlikimą atsitiktine arba blokuota (angl. *Blocked*) tvarka. Pavyzdžiui, terapeutas gali paprašyti paciento atsitiktine tvarka siekti paimti daiktus esančius įvairiose kryptyse. Šis metodas gali būti sudėtingas pacientui, nes nuolatos reikia keisti dėmesio objektą ir prisitaikyti prie skirtingų situacijų [11].

1.1.2. Kognityvinių įgūdžių reabilitacija

Kognityvinius įgūdžius žmogus naudoja mąstyti, skaityti, mokytis, prisiminti, samprotauti ir atkreipti dėmesį. Švietimo specialistė Susan du Plessis išskaido kognityvinius įgūdžius į suvokimo (regimo ir girdimo), dėmesio (sutelkto, išlaikyto ir paskirstyto), atminties (trumpalaikės, darbinės, ilgalaikės, regimosios, girdimosios) ir loginio mąstymo (dedukcinio ir indukcinio) įgūdžius [12].

Kognityvinė reabilitacija apibrėžiama kaip sistemingai taikomų medicininių ir terapinių paslaugų visuma, kuria siekiama pagerinti kognityvines funkcijas [13]. 2010 metais JAV mokslininkė Alison Cernich ir kiti mokslinio straipsnio [14] autoriai teigė, jog specifinių kognityvinių įgūdžių reabilitacijai po smegenų traumos yra reikalingas dėmesio, atminties, problemų sprendimo, kalbos ir vykdomųjų funkcijų lavinimas.

Ergoterapiautės Courtney Maher nuomone [15] atminties lavinimo pratimų atlikimo metu pacientas turi būti skatinamas įsiminti informaciją iš paveikslėlio, pasakojimo ar sąrašo, telefono numerius, adresus ir kitokią informaciją, įvardinti aplinkoje matomus objektus. Dėmesio lavinimo pratimai apima susitelkimą į konkrečią užduotį tam tikrą laiką ir pašalinių trukdžių ignoravimą. Dėmesio lavinimui galima taikyti skaičių sudėjimo, atėmimo, daugybos ir dalybos užduotis. Problemų sprendimo pratimų pavyzdžiai – galvosūkių sprendimas, strateginiai žaidimai ir sprendimų priėmimo praktikavimas kasdienėse situacijose. Kalbos lavinimo pratimų metu pacientas atlieka skaitymo, rašymo, garsų tarimo ir kalbėjimo su kitais užduotis. Vykdomųjų funkcijų lavinimas sutelkiamas į įgūdžius, leidžiančius žmonėms planuoti, organizuoti, nustatyti prioritetus, inicijuoti, palaikyti, stebėti ir koreguoti savo elgesį siekiant tikslų. Šie įgūdžiai yra būtini kasdieniam funkcionavimui, mokymuisi ir sprendimų priėmimui. Tikslų sudarymas, darbų planavimas ir užduočių prioritizavimas gali būti naudojami kaip vykdomųjų funkcijų lavinimo užsiėmimai.

1.1.3. Reabilitacija taikant virtualią realybę

Paskutiniu metu didėja domėjimasis naujais, reabilitacijos proceso gerinimui skirtais metodais, tokiais kaip virtualioji ir papildytoji realybė [16]. Tradicinės reabilitacijos teikimas gali būti brangus ir sunkiai pasiekiamas žmonėms gyvenantiems atokiuose regionuose ar neturintiems galimybės atvykti į reabilitacijos centrus. Rhutuja Khokale ir kiti mokslinio straipsnio [17] autoriai kalba apie virtualiosios realybės taikymo galimybę, kaip šios ir panašių problemų sprendimą.

Virtualiosios realybės reabilitacijos sistemos gali būti prieinamos pacientų namuose ir naudojamos savarankiškai be sveikatos priežiūros specialistų pagalbos. Šios sistemos taip pat gali būti pritaikytos kasdienio gyvenimo veikloms ir gyvenimo kokybės gerinimui. Lyginant su tradiciniais reabilitacijos metodais, VR naudojimas reabilitacijos procese gali suteikti aukštesnio pasikartojimo ir intensyvumo užduotis, objektyvų grįžtamąjį ryšį, padidinti paciento įsitraukimą ir motyvaciją [18]. VR technologija paremta reabilitacijos sistema *GTMR* teikia kognityvinio ir motorinio lavinimo užduotį, sugeba pritaikyti užduoties sudėtingumą prie paciento įgūdžių, bei nuolat jį didinti [19].

VR technologija taip pat gali pasiūlyti pacientams patrauklią ir saugią aplinką gebėjimams lavinti. Naudojantis VR sistemomis pacientai gali susitelkti į dėmesio, vykdomųjų funkcijų ir atmintiems gebėjimų lavinimą. Jiahao Meng, Zeya Yan ir Feng Gu atliktas tyrimas [20] nustatė virtualiosios ir papildytosios realybės technologijų naudą pacientų, patyrusių insultą, motorinės kontrolės lavinime. "Virtualiosios ir papildytosios realybės taikymas viršutinių galūnių judėjimo ir kasdienių veiklų atkūrimui" tyrimas [21] parodė, jog lyginant su tradicinės reabilitacijos metodais VR taikymas reabilitacijoje labiau pagerino pacientų motorines, protines ir socialines funkcijas.

2023 metais Rosaria De Luca, Simona Leonardi ir Giuseppa Maresca vykdyto tyrimo [22] rezultatai patvirtino, kad virtualioji realybė gali būti laikoma naudingu papildomu reabilitacijos metodu, padedančiu sustiprinti pacientų, sergančių afazija (angl. *Afasia*) po insulto, kalbėjimo gebėjimų funkcinį atsistatymą. Tyrime dalyvavo 30 pacientų (kurių amžiaus vidurkis buvo 51 metai) su afazija (kalbos sutrikimu) po smegenų insulto. Atsitiktinės tvarkos būdu buvo sudarytos dvi tiriamųjų žmonių grupės (po 15 žmonių): kontrolinė ir eksperimentinė. Kontrolinės grupės tiriamieji atliko kalbos ir kognityvinių įgūdžių lavinimą taikant tradicinės reabilitacijos metodus. Eksperimentinei grupei buvo taikomas įgūdžių lavinimas naudojant VRR sistema *BTS–Nirvana*. Ši sistema turi virtualius ekranus, kuriuose naudotojas gali sąveikauti su terapeuto nustatytomis užduotimis atlikdamas konkrečius pratimus. Abi grupės atliko kartojimo, skaitymo, įvardinimo, rašymo ir skaičiavimo užduotis, kurios buvo vertinamos pagal *Token* testą. Tyrimo pabaigoje buvo nustatyta, kad eksperimentinės grupės pacientai pagal *Token* testo standartus labiau pagerino skaitymo, įvardijimo ir skaičiavimo įgūdžius nei kontrolinės grupės pacientai.

Corina Schuster–Amft, Kynan Eng ir Zorica Suica 2018 metais vykdė klinikinį tyrimą [23], kurio tikslas buvo palyginti VRR ir tradicinės reabilitacijos metodų taikymą viršutinių galūnių motorinės funkcijos gydyme. Tyrime dalyvavo 54 pacientai nuo 20 iki 81 metų, kurie patyrė insultą ne ankščiau, kaip prieš 6 mėnesius. Per keturių savaičių laikotarpį kontrolinės ir eksperimentinės grupės pacientai atliko 16 reabilitacijos užsiėmimų, kurie truko po 45 minutes. Tyrimas parodė jog abi grupės žmonių patyrė panašų užsiėmimų poveikį ir naudą. Tiriamiesiems įgūdžių pagerėjimas labiausiai jautėsi per pirmąsias dvi savaites ir išliko iki dviejų mėnesių stebėjimo laikotarpio pabaigos.

Atsižvelgus į minėtų tyrimų išvadas galima teigti, kad VR technologija tapo perspektyvia motorinio mokymosi ir kognityvinių įgūdžių lavinimo priemone, kuri realų pasaulį imituojančioje virtualioje aplinkoje teikia užduotis orientuotam, pasikartojančiam ir intensyviam mokymui [24].

1.2. Dirbtinio intelekto metodai žmogaus veiklai atpažinti

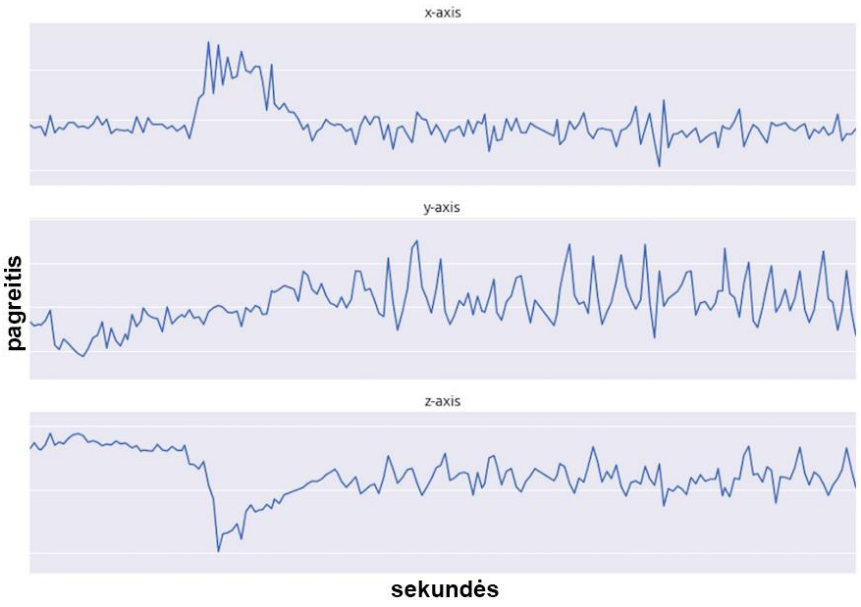
Pastaruoju metu, giliojo mokymosi modeliai, tokie kaip konvoliuciniai neuroniniai tinklai ir pasikartojantys neuroniniai tinklai, parodė galintys pasiekti gerų rezultatų mokydamiesi savybes iš neapdorotų ir jutiklių surinktų duomenų. Duomenų savybių išgavimo ir modelio kūrimo procedūros dažnai atliekamos vienu metu giliojo mokymosi modeliuose. Šiame poskyryje yra analizuojami dirbtinio intelekto metodai naudojami žmogaus veiklai atpažinti.

1.2.1. Konvoliuciniai neuroniniai tinklai

Pirmieji konvoliucinių neuroninių tinklų (CNN) modeliai buvo sukurti vaizdų klasifikavimo problemoms spręsti, kai modelis 2D paveikslo duomenims priskiria pavadinimą (angl. *Label*) ar klasę. Klasifikavimo užduotis taip pat gali būti taikoma žmogaus veiklai atpažinti iš vienmačių duomenų sekų, pavyzdžiui, pagreičio ir giroskopų surinktų duomenų. Konvoliuciniai neuroniniai tinklai gali išmokti išgauti ypatybes iš stebėjimų sekų ir suskirstyti jas į skirtingus veiklos tipus. Vienas iš CNN

naudojimo sekų klasifikavimui pranašumų yra tai, kad šie tinklai gali mokytis iš neapdorotų duomenų realiu laiku [25].

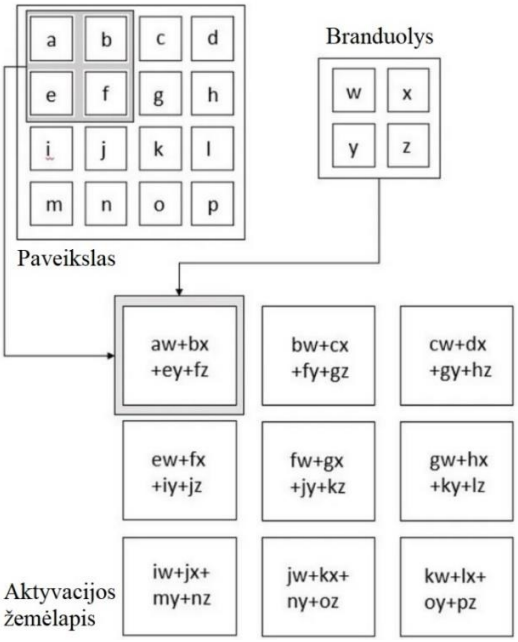
Vienmačio konvoliucinio neuroninio tinklo modeliai gali būti taikomi nuspėti žmogaus veiklą pagal judesio pėdsaką (angl. *Pattern*) naudodami jutiklius, tokius kaip akcelerometras [26]. 2012 metais Genuvos universiteto mokslininkai sukūrė duomenų rinkinį, kurį sudaro išmaniųjų telefonų surinkta žmogaus veiklos informacija [27]. Duomenų rinkiniui sukurti buvo panaudota 30 tiriamųjų judėjimo informacija, kai tiriamieji atliko 6 veiksmus, tokius kaip vaikščiojimas, lipimas laiptais, sėdėjimas, stovėjimas ir kiti, dėvint išmanųjį telefoną pritvirtintą prie liemens. Buvo registruojami judėjimo x, y ir z ašimis (žr. 1.1 pav.) tiesinio pagreičio ir giroskopiniai duomenys, matuojant 50 duomenų taškų per sekundę. Tyrimui buvo sukurtas mašininio mokymosi modelis galintis atpažinti ir nuspėti žmogaus atliekamą veiksmą pagal gaunamus duomenis iš išmaniojo telefono.



1.1 pav. Akselerometro surinkti laiko eilučių duomenys [R1]

CNN modeliai dažniausiai sudaryti iš dviejų pagrindinių tipų elementų: konvoliucinių sluoksnių ir apjungimo (angl. *Pooling*) sluoksnių. Konvoliucinio sluoksnio branduolys (angl. *Kernel*), naudoja imlų lauką (angl. *Receptive field*) savybių išgavimui iš sluoksnio įvesties duomenų, pavyzdžiui dvimačio vaizdo arba vienmačio signalo, ir žingsnį (angl. *Stride*), kuriuo slenka per įvesties duomenis [28]. Konvoliucinio sluoknsio branduolys taip pat yra žinomas kaip svorių (mokomųjų parametrų) rinkinys ar filtras, o branduolių rinkinys yra laikomas filtrų žemėlapiu (angl. *Filter map*).

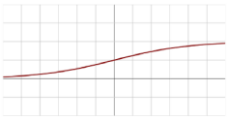
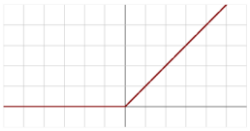
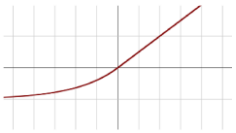
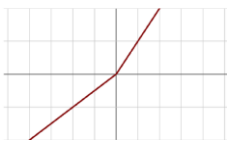
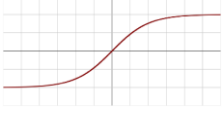
Konvoliuciniai sluoksniai atlieka dviejų matricų, branduolio ir imlaus lauko dalies (angl. *Receptive field*), taškines sandaugas vadinamas konvoliucijomis. Konvoliuciniuose sluoksniuose imlaus lauko dydis yra parenkamas pagal branduolio dydį. Konvoliucijos operacijos rezultatas yra aktyvacijos žemėlapis (angl. *Activation map*) – dvimatis reikšmių masyvas, kurį sudaro vieno konvoliucinio sluoksnio neurono išvestis (žr. 1.2 pav.) [29]. Kiekvienas sluoksnio neuronas sukuria savo aktyvacijos žemėlapį, o kiekvienas žemėlapis atspindi vieno filtro (branduolio), taikomo atitinkamiems duomenims, išvestį [30]. Savybių žemėlapis (angl. *Feature map*) – tai visų konvoliucinio sluoksnio sukurtų aktyvacijos žemėlapių rinkinys [31].



1.2 pav. Dvimačio paveikslo duomenų konvoliucijos operacija [R2]

Konvoliuciniame sluoksnyje atliekama konvoliucija yra tiesinė (angl. *Linear*) operacija, suteikianti tiesinę transformaciją įvesties duomenims. Konvoliucijos sluoksniuose yra naudojamos aktyvinimo funkcijos (angl. *Activation functions*) suteikti netiesiškumą konvoliucijos operacijos rezultatui. Aktyvinimo funkcijos gali palengvinti CNN modelio prisitaikymą prie įvairių duomenų, išvesties diferencijavimą ir normalizavimą [32]. Tokios aktyvinimo funkcijos kaip *ReLU*, *PreLU*, *ELU*, *Tanh*, *Leaky ReLU* ir kitos apriboja neuronų išvesties reikšmes tam tikrame intervale ir padeda neuroniniui tinklui pagerinti gebėjimą generalizuoti (susieti pagal požymius) naujus duomenis [33]. Šiame skyrelyje analizuojamos aktyvinimo funkcijos yra pavaizduotos 1.1 lentelėje.

1.1 lentelė. Aktyvinimo funkcijų grafikai, formulės ir reikšmių aibės [R3]

Funkcijos pavadinimas	**Funkcijos grafikas**	**Įgijamų reikšmių aibė**
Sigmoid		(0,1)
ReLU		[0,∞)
ELU		$(-\alpha,\infty)$
Leaky ReLU		(–∞,∞)
Tanh		[–1,1]

Sigmoid yra logistinė (angl. *Logistic*) aktyvinimo funkcija, kuri išveda reikšmes tarp 0 ir 1, todėl gali būti naudojama dvejetainio klasifikavimo ir logistinės regresijos problemoms spręsti [34]. Ši funkcija yra apskaičiuojama pagal (1.1) formulę:

$$f(x) = \frac{1}{1+e^{-x}}; \quad (1.1)$$

čia *x* – funkcijos įvesties reikšmė; *e* – konstanta, kurios apytikslė reikšmė yra 2,71828.

Sigmoid funckija gali sulėtinti giliųjų neuroninių tinklų darbą, nes yra paremta eksponentų skaičiavimu. Funkcija nėra centruota į 0, todėl taikant šią funkciją gali būti sunku optimizuoti tinklo sluoksnių parametrus. *Sigmoid* funkcijos grafike didelė dalis kreivės yra dalinai tiesi, todėl yra apskaičiuojami maži funkcijos gradientai, kas gali sukelti nykstančio gradiento (angl. *Vanishing gradient*), dar vadinamą neuronų prisotinimo (angl. *Saturation*), problemą, kuri pasireiškia kai išvestis mažai kinta arba visai nekinta [34].

ReLU yra netiesinė aktyvinimo funkcija naudojanti (1.2) formulę, pagal kurią *f(x)* lygi 0, kai *x* reikšmė yra mažesnė už 0 ir *f(x)* lygi *x*, kai *x* yra daugiau arba lygi 0:

$$f(x) = \begin{cases} x, kai\ x \geq 0 \\ 0, kai\ x < 0 \end{cases}; \quad (1.2)$$

čia x – funkcijos įvesties reikšmė.

ReLU funkcijos išvedamos reikšmės gali būti tik teigiamos, todėl funkcijos išvestinė (gradientas) visada bus lygi 1. Kai funkcijos išvestinė išlieka pastovi esant teigiamai įvesties reikšmei, modelio mokymosi ir klaidų minimizavimo laikas gali sutrumpėti. Lyginant su *Sigmoid* funkcija, *ReLU* konvergavimas yra apytiksliai 6 kartus greitesnis ir *ReLU* funkcijos naudojimas gali padėti išvengti tinklo neuronų prisotinimo problemos [35]. Tačiau vienas iš šios funkcijos trūkumų gali būti „mirštančių“ neuronų problema, kai tinklo neuronas nuolat grąžina 0 reikšmę, jei įvesties reikšmė buvo mažesnė už nulį [36].

Kitaip nei *ReLU*, aktyvinimo funkcija *ELU* gali apdoroti neigiamos reikšmės įvestį ir grąžinti išvestį, kuri nėra visada lygi 0. *ELU* funkcijos įvesties reikšmei artėjant prie neigiamos begalybės reikšmės, funkcijos išvestis yra lygi 0. Tačiau, kai įvesties reikšmė yra teigiama, *ELU* funkcija veikia taip pat kaip *ReLU*, jos išvestis tampa lygi įvesčiai [37]. Šios funkcijos (1.3) formulė:

$$f(\alpha, x) = \begin{cases} x, kai\ x > 0 \\ \alpha(e^{x} - 1), kai\ x \leq 0 \end{cases}; \quad (1.3)$$

čia x – funkcijos įvesties reikšmė; α – hiperparametras, kurio reikšmė yra intervale [0,1; 0,3]; *e* – konstanta, kurios apytikslė reikšmė yra 2,71828.

ELU funkcijos hiperparametras α yra naudojamas kontroliuoti išvesties reikšmių aibės ribą, nepriklausomai nuo to, kokia maža buvo įvesties reikšmė. Ši aibės riba dar yra vadinama prisotinimo tašku, kuriame ir žemiau jo funkcijos išvestis išlieka panaši į α reikšmę [38]. Funkcijos grafiko kreivė artėjant į neigiamą begalybę *x* koordinačių ašies srityje netampa visiškai tiesi, dėl (1.3) formulėje esančios *e* konstantos. Dėl šios priežasties CNN modelis nėra paveikiamas triukšmo (angl. *Noise*), jam nėra sukeliamas disbalansas ir modelio mokymosi gebėjimai gali būti pagerinami, tačiau skaičiavimo sąnaudos gali būti didelės. Naudojant *ELU* aktyvinimo funkciją, gali sumažėti modelio perkrovimo

(angl. *Overfitting*) rizika, kai modelis treniravimo duomenims pateikia aukšto tikslumo spėjimus, bet žemo tikslumo spėjimai yra grąžinami, jei modeliui buvo pateikti nematyti duomenys [38].

Leaky ReLU aktyvinimo funkcija yra *ReLU* funkcijos alternatyva, kuri bando išvengti „mirštančių“ neuronų problemos. Pagal (1.4) formulę *Leaky ReLU* funkcija gavus neigiamos vertės įvestį, grąžina nuo α hiperparametro priklausančią ir proporcingą įvesčiai išvesties reikšmę:

$$f(\alpha, x) = \begin{cases} x, kai\ x \geq 0 \\ \alpha x, kai\ x < 0 \end{cases}; \quad (1.4)$$

čia x – funkcijos įvesties reikšmė; α – hiperparametras, kurio reikšmė yra 0,01.

Kai funkcijos įvesties reikšmės yra neigiamos, funkcijos gradientas bus lygus α, todėl tinklo neuronas nesusidurs su prisotinimo problema [39].

Tanh aktyvinimo funkcija grąžina išvesties reikšmes intervale nuo –1 iki 1. *Tanh* funkcijos grafikas yra centruotas į 0, todėl šios funkcijos naudojimas CNN sluoksniuose gali palengvinti parametrų optimizavimą. Tačiau, *Tanh* funkcija neišsprendžia nykstančio gradiento problemos, kuri taip pat egzistuoja, kai yra naudojama *Sigmoid* aktyvinimo funkcija. *Tanh* funkcijos (1.5) formulėje dalyvauja *e* konstanta, todėl ši funkcija reikalauja panašių skaičiavimo sąnaudų kaip *Sigmoid* funkcija [40].

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}; \quad (1.5)$$

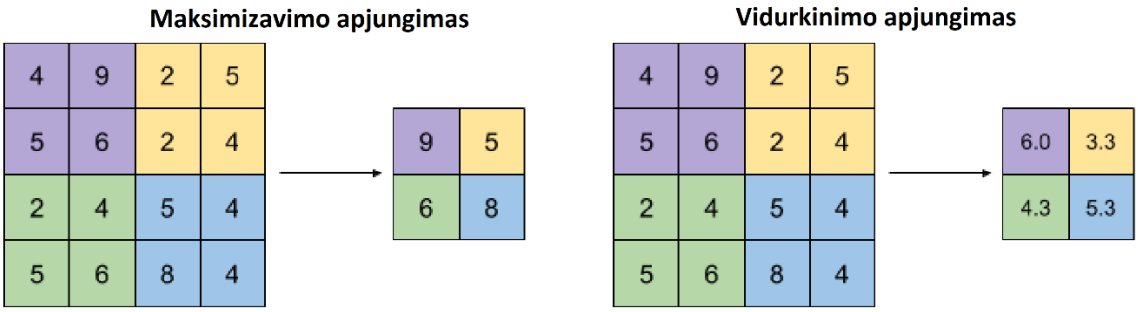
čia *x* – funkcijos įvesties reikšmė; *e* – konstanta, kurios apytikslė reikšmė yra 2,71828.

Softmax aktyvinimo funkcija dažnai yra naudojama paskutiniuose neuroninio tinklo sluoksniuose palengvinti reikšmių paskirstymą į daugiau nei į vieną klasę. Pagal (1.6) formulę ši funkcija ne tik paskirsto išvesties reikšmes intervale nuo 0 iki 1, bet ir normalizuoja jas taip, kad bendra reikšmių suma būti lygi 1. Dėl šios priežasties *Softmax* funkcijos išvestis nurodo tikimybinių reikšmių pasiskirstymą (angl. *Probability distribution*) į kiekvieną iš *K* klasių [41].

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}; \quad (1.6)$$

čia $\vec{z}$ – neuroninio tinklo neapdorotų išvesties reikšmių vektorius; *e* – konstanta, kurios apytikslė reikšmė yra 2,71828; *K* – klasių kiekis; *i* – vektoriaus elemento eilės numeris; *j* – maksimali reikšmių sumos vertė.

Apjungimo sluoksniai konvoliucinių neuroninių tinklų architektūroje atlieka savybių žemėlapio projekcijų paėmimo ir „išgryninimo“ (angl. *Distill*) operacijas skirtas įvesties duomenų sumažinimui. „Išgryninimo“ metu savybių žemėlapio projekcijos yra perskaičiuojamos naudojant signalo vidurkinimo arba signalo maksimizavimo procesus (žr. 1.3 pav). Egzistuoja įvairių tipų apjungimo metodai. Maksimizavimo (angl. *Max pooling*) ir vidurkinimo apjungimas (angl. *Avarage pooling*) yra dažniausiai naudojami konvoliucinio neuroninio tinklo apjungimo metodai. Maksimizavimo apjungimas apskaičiuoja didžiausią kiekvienos savybių žemėlapio dalies reikšmę, o vidurkinimo apjungimas apskaičiuoja kiekvienos savybių žemėlapio dalies vidutinę reikšmę [42].



1.3 pav. Maksimizavimo ir vidurkinimo apjungimo metodai [R4]

Apjungimo sluoksniai įprastai tinklų architektūrose naudojami po vieno ar daugiau konvoliucinių sluoksnių ir yra skirti įtvirtinti ankstesnių sluoksnių savybių žemėlapiuose saugomas atpažintas duomenų savybes. Apjungimas gali būti laikomas metodu, leidžiančiu suspausti arba apibendrinti įvesties duomenų savybių atvaizdus taip sumažinant modelio mokymosi duomenų perteklių. Apjungimo sluoknsiai taip pat turi imlų lauką, dažnai daug mažesnį nei konvoliucinių sluoksnių. Žingsnis, per kurį imlus laukas perkeliamas kiekvieną kartą suaktyvinant konvoliucinio tinklo neuroną, dažnai yra lygus imlaus lauko dydžiui [42].

Visiškai sujungtas arba tankus (angl. *Dense layer*) sluoksnis dažniausiai yra naudojamas kaip paskutinis konvoliucinio neuroninio tinklo architektūros sluoksnis [43]. Sluoksnio, architektūroje esančio prieš visiškai sujungtą sluoksnį, sudarytas savybių žemėlapis yra suplojamas (angl. *Flattened*) į vektorių. Šis vektorius vėliau yra perduodamas į visiškai sujungtą sluoksnį, kurio išvestis yra vienmatis savybių vektorius saugantis duomenų priklausymo klasėms tikimybes [44].

Konvoliuciniai neuroniniai tinklai, kurie yra sudaryti iš trimačių konvoliucijos sluoksnių, dažniausiai yra naudojami trimatiems vaizdo duomenims klasifikuoti. Šiuos sluoksnius galima naudoti vaizdo įrašų ir medicininių duomenų, gautų magnetinio rezonanso ar kompiuterinės tomografijos metu, klasifikavimui ar savybių iš duomenų gavimui [45].

1.2.2. Pasikartojantys neuroniniai tinklai

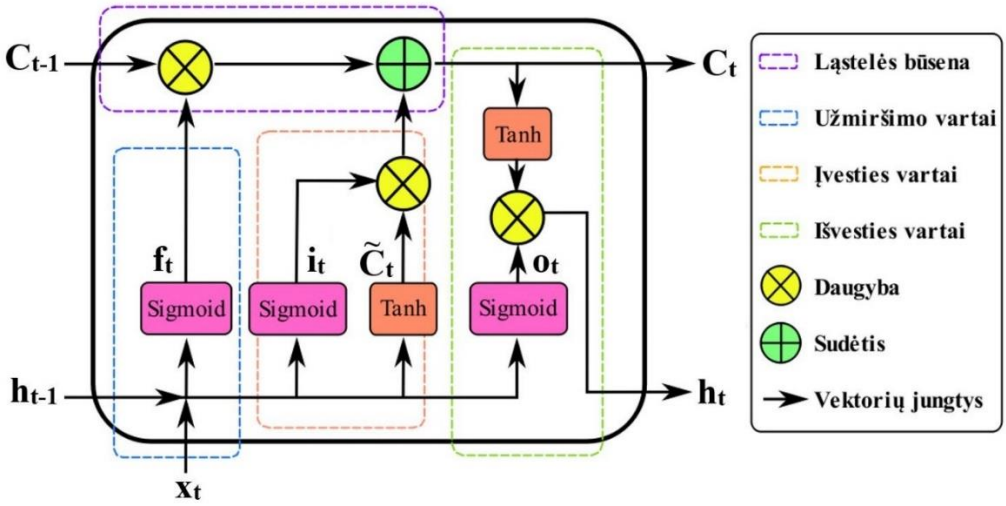
Vienas iš dirbtinių neuroninių tinklų tipų yra pasikartojantys neuroniniai tinklai (RNN), kurie dažnai yra naudojami kalbos ir stebėjimo sekų atpažinimui. Šie tinklai pasižymi gebėjimu atpažinti nuoseklias duomenų charakteristikas ir panaudoti jas įvariems scenarijams nuspėti [46]. Ilgalaikės trumpalaikės atminties tinklas (LSTM) priklausantis RNN tinklų šeimai, parodė esantis veiksmingas sprendžiant sekų numatymo problemas tokioms užduotis, kaip rašysenos atpažinimas, kalbos modeliavimas ir mašininis vertimas [47].

LSTM tinklai yra sudaryti iš specialių vienetų vadinamų vartais (angl. *Gates*), kurie kontroliuoja informacijos srauto tėkmę į atminties ląstelę (angl. *Memory cell*) ir iš jos. Šis vartų mechanizmas kontroliuoja tinklo įsiminimo procesą, nes informacija tinkle gali būti išsaugoma, užrašoma ar skaitoma, tik tada, kai tinklo vartai atsidaro ar užsidaro. Vartai informaciją saugo analoginiu formatu (angl. *Analog*), nes jis yra diferencijuojamo tipo ir gali būti tinkamas atgaliniui skleidimui (angl. *Backpropagation*) [48].

LSTM tinkluose yra naudojamos *Tanh* ir *Sigmoid* aktyvinimo funkcijos. *Tanh* aktyvinimo funkcija ilgalaikės trumpalaikės atminties tinklo architektūroje atlieka reikšmių, tekančių per tinklą, reguliavimą, palaikant jų reikšmes intervale nuo –1 iki 1. Tinklo vartuose naudojama *Sigmoid*

aktyvinimo funkcija palaiko reikšmes intervale nuo 0 iki 1 ir padeda tinklui nuspręsti, kuriuos duomenis reikėtų atnaujinti arba pamiršti. Reikšmės, kurios yra lygios 0, pamirštamos, o reikšmės lygios 1 yra paliekamos [48].

LSTM tinkle yra trijų tipų vartai: įvesties, užmiršimo (angl. *Forget*) ir išvesties. Kiekvienas LSTM tinklo vienetas taip pat turi vidinę atmintį arba būseną, kuri yra papildoma nuskaitant įvesties seką. Vieneto būseną LSTM tinklas gali naudoti kaip vietinį kintamąjį arba atminties registro tipą.



1.4 pav. LSTM tinklo ląstelės sandara [R5]

1.4 paveiksle yra pavaizduota LSTM tinklo ląstelės struktūra. Informacija į ląstelės užmiršimo vartus patenka per įvestį x_t laiko momentu t ir ląstelės išvestį iš praeito laiko momento h_{t-1}. Šiems vektoriams yra taikoma *Sigmoid* aktyvinimo funkcija. Ji nurodo informacijos dalį, kuri ląstelėje turi būti užmiršta arba palikta. *Sigmoid* funkcija grąžina užmiršimo vartų t momentu vektorių f_t. Užmiršimo vartams taikoma (1.7) formulė [49]:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \; ; \tag{1.7}$$

čia t – laiko momentas; f_t – užmiršimo vartų rezultatas; σ – *Sigmoid* funkcija; h_{t-1} – ląstelės išvesties iš praeito laiko momento vektorius; x_t – ląstelės įvesties vektorius; b_f – užmiršimo vartų nuokrypis (angl. *Biase*); W_f – užmiršimo vartų neuronų svorių matrica.

Pagal (1.8) ir (1.9) formules [49] ląstelės įvesties vartuose x_t ir h_{t-1} vektoriams yra tai taikomos *Sigmoid* ir *Tanh* funkcijos, kurių išvesties vektoriai i_t ir $\tilde{C}_t$ yra sudauginami (1.10) formulėje [49].

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \; ; \tag{1.8}$$

čia t – laiko momentas; i_t – įvesties vartų rezultatas; σ – *Sigmoid* funkcija; h_{t-1} – ląstelės išvesties iš praeito laiko momento vektorius; x_t – ląstelės įvesties vektorius; b_i – įvesties vartų nuokrypis; W_i – įvesties vartų neuronų svorių matrica;

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \; ; \tag{1.9}$$

čia t – laiko momentas; $\tilde{C}_t$– reprezentuoja kandidatą kitai ląstelės būsenai laiko momentu t; $tanh$ – *Tanh* funkcija; h_{t-1} – ląstelės išvesties iš praeito laiko momento vektorius; x_t – ląstelės įvesties vektorius; b_C – ląstelės būsenos nuokrypis; W_C – ląstelės būsenos svorių matrica;

Taikant (1.10) formulę [49] ląstelės būsenos laiko momentu *t* vektoriaus gavimui yra sudedamas įvesties vartų rezultatas su užmiršimo vartų rezultatu sudaugintu su C_{t-1} vektoriumi, kuris saugo ląstelės būseną iš praeito laiko momento.

$$C_t = (f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t) ; \quad (1.10)$$

čia *t* – laiko momentas; C_t–ląstelės būsena laiko momentu *t*; f_t – užmiršimo vartų rezultatas; C_{t-1} – ląstelės būsena praeitame laiko momente; i_t – įvesties vartų rezultatas; $\tilde{C}_t$– reprezentuoja kandidatą kitai ląstelės būsenai laiko momentu *t*.

Išvesties vartų rezultatas o_t yra apskaičiuojamas pagal (1.11) formulę [49], kai x_t ir h_{t-1} vektoriams yra taikoma *Sigmoid* funkcija.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) ; \quad (1.11)$$

čia *t* – laiko momentas; o_t – išvesties vartų rezultatas; σ – *Sigmoid* funkcija; h_{t-1} – ląstelės išvesties iš praeito laiko momento vektorius; x_t – ląstelės įvesties vektorius; b_o – išvesties vartų nuokrypis; W_o – išvesties vartų neuronų svorių matrica;

Taikant (1.12) formulę [49], išvesties vartų rezultatas yra sudauginamas su *Tanh* funkcijos taikytos C_t vektoriui išvestimi. Šios operacijos rezultatas h_t yra ląstelės išvestis laiko momentu *t*.

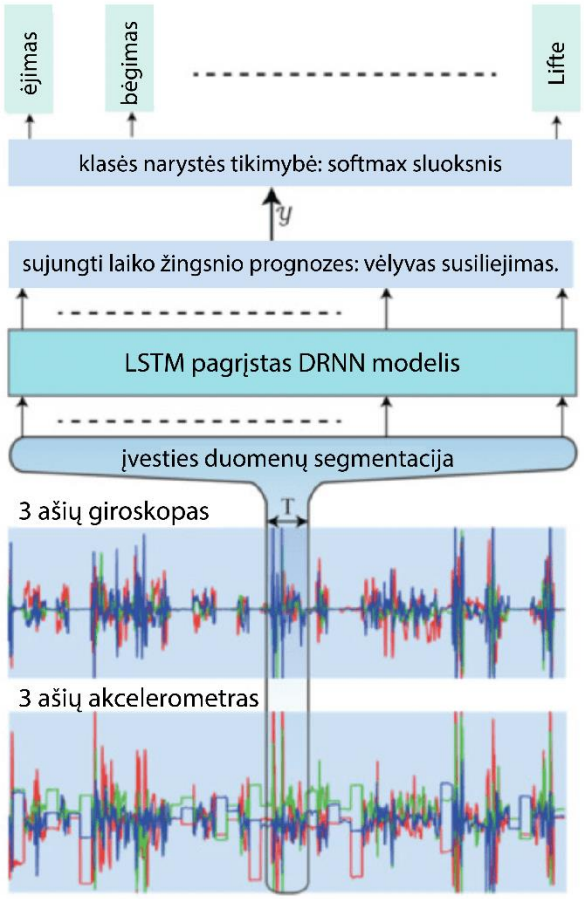
$$h_t = o_t \cdot tanh(C_t) ; \quad (1.12)$$

čia *t* – laiko momentas; C_t – ląstelės būsena laiko momentu *t*; tanh – *Tanh* funkcija; o_t – išvesties vartų rezultatas; h_t – ląstelės išvestis laiko momentu *t*.

Kaip ir CNN tinklai, kurie gali skaityti duomenis įvesties sekoje, LSTM nuskaito įvesties stebėjimų seką ir sukuria savo vidinį įvesties sekos atvaizdą. Skirtingai nei konvoliucinis neuroninis tinklas, ilgalaikis trumpalaikės atminties tinklas yra apmokomas taip, kad skirtų ypatingą dėmesį stebėjimams ir klaidų, padarytų per įvesties sekos laiko etapus, prognozavimui [47].

Ilgalaikiai trumpalaikės atminties tinklai gali būti taikomi žmogaus veiklos atpažinimo užduotims. Šie tinklai išmoksta susieti žmogaus veiklą registruojančio jutiklio duomenų langą su tam tikra veikla, kai įvesties sekos stebėjimai skaitomi po vieną ir atsikirus laiko žingsnius gali sudaryti vienas ar keli kintamieji. Abdulmajid Murad ir Jae–Young Pyun straipsnio [50] autoriai siūlo naudoti giliuosius pasikartojančius neuroninius tinklus DRNN (angl. *Deep recurrent neural networks*), paremtus LSTM tinklų architektūra, kuriant žmogaus veiklos atpažinimo modelius, kurie gali užfiksuoti ilgalaikes priklausomybes kintamo ilgio įvesties sekose. Minėtame straipsnyje [50] buvo pateiktos vienkryptės, dvikryptės ir pakopinės LSTM tinklų architektūros, bei įvertintas jų efektyvumas su skirtingais duomenų rinkiniais. Tyrime buvo panaudotas DRNN tinklas, kuris apdoroja sekos duomenis pirmyn (angl. *Forward – normal*) ir abejomis kryptimis (dvikryptis LSTM). Pastebėta, jog pasirinktas DRNN tinklas numato aktyvumą kiekvienam jutiklio duomenų sekos įvesties laiko žingsniui, kurį apjungiant galima numatyti lango veiklą. Straipsnis [50] aprašo pasiūlytą žmogaus veiklos atpažinimo sistemą (žr. 1.5 pav.), kuri atlieka tiesioginį ir nuoseklų atvaizdavimą nuo neapdorotų multimodalinių (angl. *Multi–modal*) jutiklių įvesčių iki veiklos klasifikavimų. Ši sistema klasifikuoja per tam tikrą laikotarpį atliktos veiklos klasę. Sistemos įvestis yra atskirta, vienodai išdėstytų mėginių seka, kurioje kiekvienas duomenų taškas yra atskirų mėginių, kuriuos jutikliai stebi laiko momentu *t*, vektorius. Šie pavyzdžiai yra segmentuojami į laiko indekso T langus ir pateikiami į DRNN modelį pagrįstą LSTM architektūra. Modelis išveda klasės numatymo įverčius kiekvienam laiko žingsniui, kurie yra sujungiami per vėlyvą

suliejimą (angl. *Late–fusion*) ir įvedami į *Softmax* aktyvinimo fukcijos sluoksnį, tam kad būtų galima nustatyti klasės priklausymo tikimybę.



1.5 pav. Sistemos planas žmogaus veiklai atpažinti [R6]

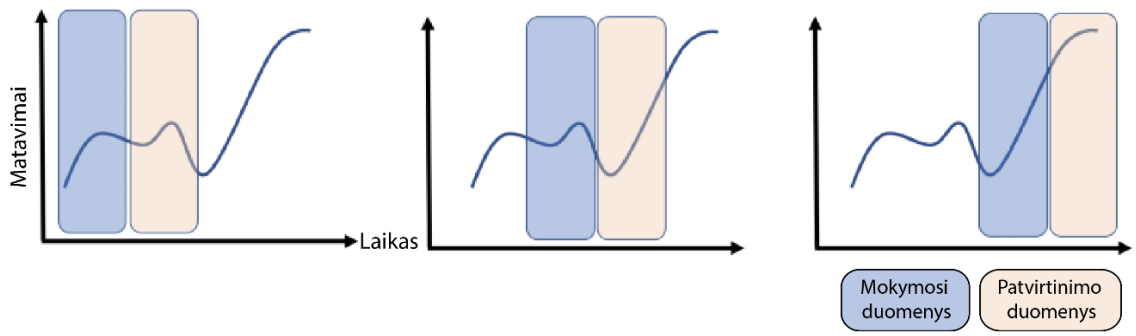
RNN tinklai gali pasinaudoti laiko eilės ryšiu tarp jutiklių surinktų duomenų, tačiau CNN tinklai geriau išmoksta giliųjų savybių taikant rekursyvinius modelius. Dėl šios priežasties RNN tinklai yra rekomenduojami atpažinti trumpą veiklą, kuri turi natūralią tvarką, o CNN modeliai labiau tinkami atpažinti ilgalaikę pasikartojančią veiklą [47].

1.3. Duomenų tvarkymo metodai

Konvoliuciniai ir pasikartojantys neuroniniai tinklai naudojami duomenų klasifikavimui, gali pasikiekti gerus rezultatus, kai duomenims tvarkyti buvo taikomi tam tikri metodai. Šiame skyrelyje yra aprašomi duomenų paruošimo metodai, tokie kaip slankiojantis langas, normalizavimas ir standartizavimas.

1.3.1. Slankiojantis langas

Slankiojančio lango (angl. *Sliding window*) algoritmas yra vienas iš metodų naudojamų operacijoms su duomenų sekomis. Šio metodo taikymas apima įvesties signalo padalijimą į signalų langus, kai viename signalo lange yra stebimi duomenys iš tam tikro laiko momento (žr. 1.6 pav.) [51].



1.6 pav. „Slankiojančio lango“ duomenų paruošimo metodas [R7]

Neuroninių tinklų modeliuose, kurie naudojami žmogaus veiklai atpažinti, dažnai yra taikomas fiksuoto ilgio slankiojančio lango metodas, skirtas savybių iš duomenų išgavimui. Naudojant šį metodą reikia nustatyti lango dydžio ir poslinkio parametrus. Duomenų langą gali sudaryti keli kintamieji, pavyzdžiui akselerometro surinkti duomenys x, y ir z koordinačių ašyse. Jutiklių, tokių kaip akselerometras, surinktų duomenų eilutės dydis, gali priklausyti nuo duomenų registravimo dažnio, išreikšto Hz matavimo vienetu [51].

Slankiojančio lango dydis dažnai priklauso nuo naudojamo mašininio mokymosi modelio, surinktų duomenų pobūdžio ir atpažinamos veiklos. Lango dydžio pasirinkimas gali turėti įtakos atliekamų skaičiavimų greičiui. Lango dydžio mažinimas gali leisti greičiau aptikti veiklą, taip pat sumažinti didelių skaičiavimo resursų poreikį. Dideli duomenų langai įprastai yra naudojami sudėtingoms veikloms atpažinti. 2014 metais Oresti Banos ir kitų autorių straipsnyje [52] yra tiriama lango dydžio įtaka žmogaus veiklai atpažinti. Atliktas tyrimas [52] nustatė, kad signalų langams parinkus dydį mažesnį už 2 sekundžių trukmės intervalą, galima pasiekti gerus modelio klasifikavimo tikslumo ir našumo rezultatus. Tyrimas įrodė, jog tiksliausias žmogaus veiklų atpažinimas gali būti gaunamas esant labai mažiems langams (0,25 – 0,5 sekundžių trukmės) Šis tyrimas parodė, kad didelių langų naudojimas nevisada nulems geresnį modelio tikslumą [52].

Taikant slankiojančio lango metodą egzistuoja tikimybė, kad padalijus jutiklių surinktų duomenų srautą į langus, veiklų pasikeitimo duomenys gali būti prarasti, kai buvo registruojamos daugiau nei viena veiklos [53]. Dėl šios priežasties duomenis galima skaidyti į langus, naudojant persidengimo metodą (angl. *Overlap*), kai pirmoje lango pusėje yra veiklos duomenų iš paskutinio vykdyto stebėjimo, kurie buvo patalpinti į praeito lango antrą pusę.

Persidengimo metodo naudojimas priklauso nuo sprendžiamos problemos ir turimų duomenų tipo, tačiau šis metodas gali neuroninių tinklų mokymo (treniravimo) duomenų kiekį padvigubinti, kai taikomas 50 % persidengimas [53]. 2023 metais Harriet L. Dawson ir kiti autoriai atliko tyrimą [54], kurio tikslas buvo ištirti duomenų rinkinio dydžio įtaką konvoliucinių neuronių tinklų našumui. Šis tyrimas nustatė, jog duomenų rinkinio dydis turi didelę įtaką CNN tinklams. Tyrimo metu modeliai, kurie buvo apmokomi su mažais (mažiau nei 5000 duomenų vienetų) duomenų rinkiniais, parodė didelę tikimybę patirti perkrovimo riziką, kai modelis treniravimo duomenims pateikia aukšto tikslumo spėjimus, bet žemo tikslumo spėjimai yra grąžinami, kai modeliui yra pateikiami nauji duomenys [54].

1.3.2. Normalizavimas ir standartizavimas

Neapdorotų ir skirtingų dydžių duomenų naudojimas mašininio mokymosi modeliuose gali pasireikšti nenuspėjamais rezultatais ir lėtu treniravimo greičiu. Duomenų paruošimui gali būti taikomi normalizavimo metodai, kurie transformuoja duomenis taip, kad jie atitiktų bendrą visų duomenų mastelį (skalę (angl. *Scale*)) ir būtų vienodai paskirstyti [55]. Objektų atpažinimo ir klasifikavimo užduotyse iš dvimačių paveikslų, pikselių vertėms dažnai yra taikomas normalizavimas, paskirstant pikselių reikšmes intervale, pavyzdžiui nuo 0 iki 1, taip užtikrinant vienodą kiekvieno pikselio svarbą atliekamuose skaičiavimuose. Normalizavimas, taip pat taikomas ir balso atpažinimo užduotyse, norint užtikrinti garso įrašų duomenų garsumo ir dažnio suvienodinimą. Įvairiems duomenims gali tikti skirtingi duomenų normalizavimo metodai. Dažniausiai naudojamos duomenų normalizavimo technikos yra:

1. minimumų ir maksimumų (angl. *Min–Max*) normalizavimas;
2. z įverčio (angl. *Z–score*) normalizavimas;
3. dešimtainis mastelio keitimas;
4. logaritminis transformavimas.

Minimumų ir maksimumų normalizavimo, dar vadinamo duomenų rinkinio mastelio keitimo, metu duomenys yra paskirstomi atitinkamame intervale, dažniausiai nuo 0 iki 1. Ši operacija perskaičiuoja reikšmių vertes pagal (1.13) formulę [55]:

$$x_{norm} = \frac{\mathrm{x} - x_{min}}{x_{max} - x_{min}}; \tag{1.13}$$

čia x – pradinė duomenų rinkinio vieneto reikšmė; x_{min} – mažiausia duomenų rinkinio vieneto reikšmė; x_{max} – didžiausia duomenų rinkinio vieneto reikšmė; x_{norm} – normalizuota duomenų rinkinio vieneto reikšmė.

Minimumų ir maksimumų normalizavimo metodas leidžia išsaugoti duomenų pasiskirstymo rinkinyje formą. Tačiau šis metodas yra nerekomenduojamas, kai duomenų rinkinyje yra neįprastai didelių ar mažų reikšmių.

Z įverčio normalizavimas, taip pat dažnai vadinamas duomenų standartizavimu, duomenis perskaičiuoja pagal (1.14) formulę taip, kad jų vidurkis būtų lygus 0, o standartinis nuokrypis lygus 1 [55]. Duomenų vieneto standartizacijos metu iš visų duomenų rinkinio reikšmių vidurkio yra atimama normalizuojamo duomenų vieneto reikšmė, tada šis skirtumas yra padalijamas iš duomenų rinkinio standartinio nuokrypio apskaičiuojamo pagal (1.15) formulę [56].

$$x_{norm} = \frac{\mathrm{x} - \mu}{\sigma}; \tag{1.14}$$

čia x – pradinė duomenų rinkinio vieneto reikšmė; μ – duomenų rinkinio reikšmių vidurkis; σ – duomenų rinkinio standartinis nuokrypis; x_{norm} – normalizuota duomenų rinkinio vieneto reikšmė.

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{\mathrm{N}}}; \tag{1.15}$$

čia x_i –duomenų rinkinio vieneto reikšmė; i – duomenų vieneto eilės numeris rinkinyje; μ – duomenų rinkinio reikšmių vidurkis; σ – duomenų rinkinio standartinis nuokrypis; N – duomenų kiekis rinkinyje.

Šis metodas gali būti naudojamas, kai norima nustatyti neįprastas reikšmes duomenų rinkinyje ar palyginti skirtingus duomenų rinkinio vienetus. Tačiau duomenų standartizavimas neišsaugo duomenų pasiskirstymo rinkinyje formos, kaip minimumų ir maksimumų metodas, bei gali apsunkinti pradinių reikšmių interpretavimą [55] Standartizavimo metodas ir minimumų ir maksimumų metodas yra paliginami 1.2 lentelėje.

1.2 lentelė. Normalizavimo metodų palyginimas

Minimumų ir maksimumų metodas	**Z įverčio (standartizavimo) metodas**
Naudoja mažiausias ir dižiausias duomenų rinkinio reikšmes.	Naudoja duomenų rinkinio reikšmių vidurkio ir standartinio nuokrypio reikšmes.
Rezultatas: rinkinio reikšmės yra pasiskirstę intervale, pavyzdžiui [0,1] ar [–1,1].	Rezultatas: rinkinio reikšmių vidurkis yra lygus 0 ir standartinis nuokrypis lygus 1; Reikšmės nėra paskirstomos specifiniame intervale.
Nerekomenduojamas, kai duomenų rinkinyje yra ekstremaliai didelių ar mažų reikšmių.	Galima naudoti, kai duomenų rinkinyje yra ekstremaliai didelių ar mažų reikšmių.

Dešimtainis mastelio keitimo metodas pakeičia duomenų rinkinio reikšmes padalindamas jas iš skaičiaus 10 pakelto labai mažu laipsniu *j*. Šio metodo (1.16) formulė nurodo, jog *j* reikšmė, turi būti parinkta taip, kad didžiausia normalizuota reikšmė rinkinyje būtų mažensė už 1. Dešimtainį mastelio keitimo metodą galima naudoti tada, kai norima išsaugoti panašų į pradinį verčių dydį, bet taip pat supaprastinti duomenis rinkinyje [55], lengvesniam jų apdorojimui.

$$x_{norm} = \frac{\mathrm{x}}{10^{\,j}}; \tag{1.16}$$

čia x – pradinė duomenų rinkinio vieneto reikšmė; $j -$ konstanta, kurios vertė parenkama taip, kad didžiausia rinkinio x_{norm} reikšmė yra mažesnė už 1; $x_{norm} -$ normalizuota duomenų rinkinio vieneto reikšmė.

Logaritminio transformavimo metodas yra pagrįstas (1.17) formulėje [55] pavaizduotu logaritmavimo veiksmu, kuris apskaičiuoja duomenų reikšmių logaritmus. Šis normalizavimo metodas keičia duomenų rinkinio reikšmių mastelį ir gali būti naudingas, kai duomenys rinkinyje yra iškreipti ir norima suspausti reikšmes mažesniame intervale. Tačiau šis metodas gali netinkamai veikti su neigiamomis ar 0 lygioms reikšmėmis [55].

$$x_{norm} = \log_e x = \ln x; \tag{1.17}$$

čia x – pradinė duomenų rinkinio vieneto reikšmė; $e -$ konstanta, kurios apytikslė reikšmė yra 2,71828; $x_{norm} -$ normalizuota duomenų rinkinio vieneto reikšmė.

1.3 lentelė. Normalizavimo metodų privalumai ir trūkumai [55]

Privalumai	**Trūkumai**
Gali padinti mašininio mokymosi modelių tikslumą.	Normalizacijos metodai gali pakeisti pradines duomenų reikšmes ir jų pasiskirstymo intervalą.
Mašininio mokymosi modeliams gali būti lengviau interpretuoti normalizuotus duomenis.	Normalizacijos metodų taikymas gali apsunkinti ekstremaliai didelių ar mažų reikšmių rinkinyje radimą ir pašalinimą.

Normalizavimas gali sumažinti duomenų perteklių (angl. Redundancy) pašalinant vienodus ar nesvarbius duomenis.	Svarbu pasirinkti tinkamą normalizacijos metodą atsižvelgiant į turimus duomenis. Netinkamo metodo pasirinkimas gali sumažinti mašininio mokymosi modelių tikslumą.
Gali padinti mašininio mokymosi modelių našumą ir sumažinti modelio perkrovimo (angl. Overfitting) riziką.	Normalizacijos metodų taikymas gali apsunkinti duomenų analizės procesą.

Pagal 1.3 lentelėje aprašytus normalizavimo metodų taikymo privalumus ir trūkumus, galima teigti, kad normalizacijos metodai gali pagerinti mašininio mokymosi modelių tikslumą ir našumą, bei sumažinti perkrovimo riziką. Deja, šie metodai taip pat pakeičia pradines duomenų reikšmes ir jų pasiskirstymą bei gali apsunkinti duomenų analizės procesą [55].

1.4. Rankų sekimo technologijos

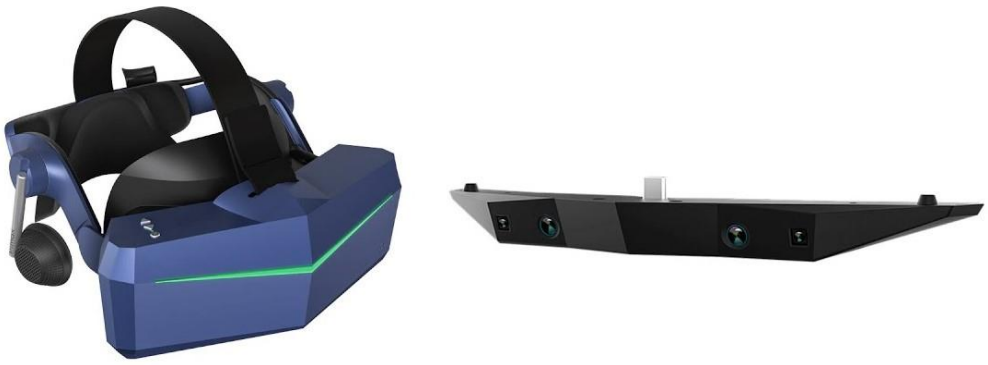
Rankų sekimo technologijos taikymas virtualioje realybėje gali leisti VR sistemų naudotojams sąveikauti su virtualiais objektais ir situacijomis valdant jų pačių rankas. Nors rankų naudojimas įvairiems veiksmams atlikti realiame gyvenime yra įprastas reiškinys daugeliui žmonių, virtualioje realybėje tai yra gana nauja ir pažangi technologija, kuri patirtis virtualioje realybėje gali paversti dar labiau įtraukiančiomis ir įspūdingomis [57].

Rankų sekimas yra procesas, kurio metu virtualios realybės akiniuose esančiomis kameromis, *LiDAR* masyvais ar išorinėmis jutiklių stotimis yra sekama rankų vietos, judėjimo greičio ir orientacijos erdvėje informcija. Duomenys, kurie buvo surinkti tokiu būdu, yra įvertinami ir gali būti paverčiami virtualiomis, realiu laiku atvaizduojamomis rankomis virtualioje aplinkoje. Šis rankų atvaizdavimas yra pateikiamas programinei įrangai, kuri leidžia jos naudotojui intuityviai sąveikauti su aplinka naudojantis savo rankomis [58].

1.4.1. Optinis sekimas

Optinis sekimas (angl. *Optical tracking*) yra procesas, kurio metu rankų sekimas yra atliekamas naudojant kameros užfiksuotų vaizdų masyvą (angl. *Camera array*). Rankų judėjimui erdvėje nustatyti sekimas yra atliekamas apdorojant pikselių pasikeitimus 2D vaizduose. Dėl šios priežasties, virtualių rankų atvaizdavimas gali reikalauti papildomų skaičiavimo resursų. Optinis sekimas gali būti patogus norint klasifikuoti 2D nuotraukų sekose užfiksuotus judesius į skirtingus veiklų tipus taikant mašininio mokymosi modelius. Optinis sekimas priklauso nuo kameros stebimo lauko (kitaip vadinamo aktyvia sekimo sritimi), jautrumo apšvietimui ir gebėjimo užfiksuoti judesius tam tikru kadrų dažniu. Įprastai optinio sekimo technologijos taikymas gali būti pigesnis lyginant jį su kitais rinkoje egzistuojančiais rankų sekimo sprendimais [58].

Pimaxx Vision 8K X virtualios realybės akiniai (žr. 1.7 pav.) naudoja *Leap Motion* optinio sekimo aparatinę įrangą realiu laiku virtualioje erdvėje atvaizduoti rankų judėjimą. Šis įrenginys kameros užfiksuotų pikselių reikšlių pasikeitimus konvertuoja į neapdorotus rankos vietos erdvėje duomenis. Rankų sekimo įrangą sudaro stereoskopinė infraraudonųjų spindulių kamera, kuri stebi rankų judėjimą iki 1 metro atstumo ir turi 160×160 laipsnių kampą sudarantį matymo lauką (aktyvią sekimo sritį) [59].



1.7 pav. *Pimax Vision 8K X* virtualiosios realybės akiniai ir jų optinio sekimo įrenginys [R8]

Nuo 2010 metų JAV įkurta įmonė *Ultraleap* gamina ir pardavinėja aparatinę įrangą, palaikančią rankų ir pirštų judėjimo sekimą. Ši įmonė sukūrė modulinį įrenginį *Leap Motion Controller*, kuris gali būti pritaisomas prie VR akinių ir naudojamas virtualios, papildytos ir mišrios (angl. *Mixed*) realybės sistemų prototipavimui, tyrimui ir kūrimui. Įrenginys gali sekti judėjimą interaktyvioje trimatėje erdvėje iki 80 cm atstumu ir turi 140×120 laipsnių kampą sudarančią aktyvią sekimo sritį. *Leap Motion Controller* įrenginys gali atskirti 27 elementus, tokius kaip pirštų kaulus ir sąnarius, bei gali sekti jų judėjimą erdvėje, net kai jie yra paslėpti kitų elementų. Įrenginį sudaro (žr. 1.8 pav.) dvi 640×240 pikselių raiškos artimojo infraraudonojo spektro kameros, kurios veikia 120 Hz dažniu ir gali užfiksuoti vaizdus per $\frac{1}{2000}$ sekundės [60].

1.8 pav. *Leap Motion Controller* naudojimo pavyzdys ir jį sudarantys elementai [R9]

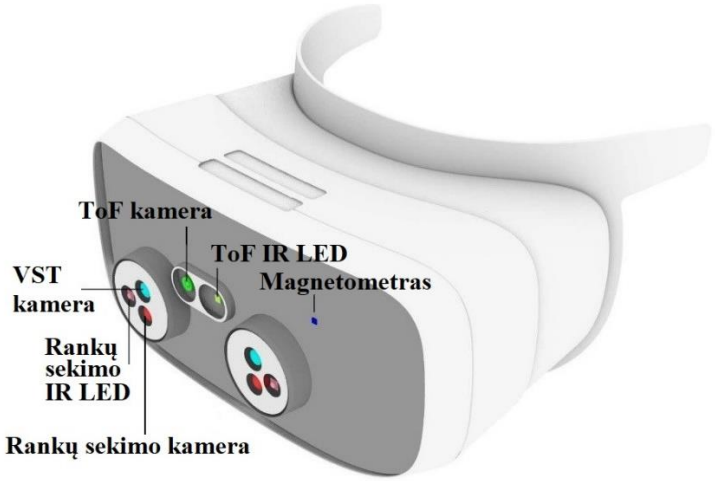
Minimalūs sistemos reikalavimai taikomi šiam įrenginiui:

- USB 2.0 jungtis;
- 2 GB operatyvioji atmintis (RAM);
- *Intel Core i3, Intel Core i5, Intel Core i7* ar *AMD Phenom II* procesorius;
- *Windows 7* ar *Mac OS X 10.7* operacinė sistema.

1.4.2. *LiDAR* sekimas

Šviesos aptikimo ir išdėstymo (*LiDAR* sensorių) technologija paremti įrenginai naudoja lazerio impulsus naudotojo rankų padėties, atstumo nuo įrenginio iki objekto ir judėjimo įvertinimui bei sekimui. Ši technologija, gali tiksliau atpažinti ir sekti rankas, nei optinio sekimo technologija, nes

aptikimas naudojant lazerius yra greitesnis. *LiDAR* sensorių sekimo naudojimas taip pat pasižymi aukštesne sekimo kokybe, nes sekimo tikslumas nepriklauso nuo atstumo iki objekto ilgio. *LiDAR* technologija naudoja atsispindinčius lazerius, panašiai kaip veikia echolokacija. Kadangi optinis sekimas yra paremtas pikselių verčių pasikeitimu, 2D vaizdų raiška prastėja didėjant atstumui nuo kameros iki objekto [61]. 1.9 pav. pavaizduoti *Varjo XR–3* virtualios realybės akiniai, naudoja *LiDAR* sensorių atstumo iki objekto sekimą kartu su optiniu sekimu, tam kad būtų užtikrintas aukšto tikslumo rankų sekimas [62].



1.9 pav. Virtualios realybės akinių *Varjo XR–3* komponentai [R10]

LiDAR sensoriaus skleidžiamų šviesos bangų atsispindėjimo nuo objekto metu surinkti duomenys yra panaudojami trimačio vaizdo generavimui. Infraraudonųjų spindulių šviesos diodai (angl. *Infrared LEDs*), veikiantys panašiai kaip *LiDAR*, taip pat gali būti naudojami objektų sekimui. Aukšto tikslumo objektų sekimas gali būti pasiektas apjungiant optinį ir infraraudonųjų spindulių šviesos diodų sekimą [61].

1.5. Virtualios realybės formos

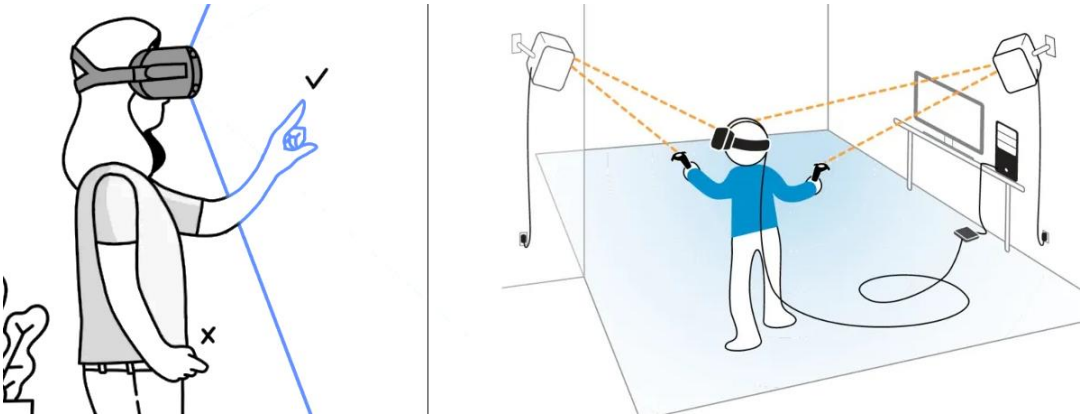
Virtualioji realybė panardina vartotojus į sintetiniu būdu sukurtą virtualią arba imituojamą aplinką. Vienas iš virtualios realybės realizavimo būdų yra ant galvos montuojamo ekrano (VR akinių) panaudojimas, siekiant perkelti naudotojus į uždarą trimatį pasaulį. VR akiniai turi naudotojo padėties sekimo technologiją, garso įvesties ir išvesties įrenginius, bei jutiminio grįžtamojo ryšio priemones. Įvairios virtualiosios realybės sistemos, pavyzdžiui vairavimo ar realaus pasaulio projekcijos imitaciją teikiančios, gali būti įgyvendinamos naudojant VR akinius, kaip pagrindinį duomenų įvesties ir išvesties įrenginį. Simuliacija pagrįstos virtualiosios realybės sistemos imituoja kompiuteriu sukurtą realaus pasaulio aplinką, įvairias aplinkybes ir reiškinius. Populiaru šio tipo VR sistemas naudoti edukacijoje, nes jos gali padėti lavinti įgūdžius įtraukiančiu ir įsimintinu būdu. Projektoriumi pagrįstoje virtualiojoje realybėje naudotojams yra pateikiama aplinka, kuri sieka atkurti realaus pasaulio vaizdą. Projektoriumi grindžiamos VR sistemos tikslas yra panardinti žiūrovus į virtualią, tačiau labai artimą realiam gyvenimui, aplinką [63].

1.6. Virtualioje realybėje naudojami judesių sekimo metodai

Virtualios realybės įrenginiai naudotojo judesių sekimą gali atlikti dviem būdais: „iš vidus į išorę“ (angl. *Inside out*) ir „iš išorės į vidų“ (angl. *Outside in*) [64]. Šis poskyris aprašo šių sistemų ypatybes ir esminius skirtumus.

1.6.1. “Iš vidaus į išorę“ sekimo metodas

„Iš vidaus į išorę“ sekimo metodas dažniausiai yra naudojamas su belaidžiais VR akiniais (žr. 1.10 pav.), kurie gali veikti nepriklausomai nuo kitos aparatinės ar programinės įrangos.



1.10 pav. Sekimo metodų „iš vidaus į išorę“ ir „iš išorės į vidų“ taikymas VR [R11]

Šis sekimo metodas yra įgyvendinamas su optinio sekimo kameromis ir *LiDAR* sensoriais integruotais į VR akinius, kurie seka naudotojo poziciją realioje aplinkoje ir apdoroja sekimo duomenis į padėties erdvėje duomenis. Toks sekimas yra atliekamas tik iš vienos krypties atitinkamo dydžio aktyvios sekimo srities, todėl sekimo kokybė ir tikslumas gali būti mažesnis, nei taikant „iš išorės į vidų“ metodą [63].

1.6.2. “Iš išorės į vidų“ sekimo metodas

“Iš išorės į vidų“ sekimo metodas yra paremtas aparatine įranga, tokia kaip kameros ar sensoriai, kurie naudotojo realioje aplinkoje yra pastatyti stacionarioje pozicijoje ir sudaro stebėjimo zoną, kurioje naudotojo judėjimas bus sekamas ir perteikiamas į virtualią aplinką (žr. 1.10 pav.). Šis metodas nurodo, kad sistemos naudotojas gali judėti tik nustatyto ploto erdvėje, pavyzdžiui *HTC Vive* virtualios realybės įrangos prekės ženklas siūlo sekimo stoteles, kurių sekamas plotas yra nuo 3.5×3.5 m (kai naudojamos dvi *Base Station* aparatinės įrangos stotelės [65]) iki 10×10 m (kai naudojamos keturios *Base Station 2.0* aparatinės įrangos stotelės [66]). Naudotojo judėjimo sekimas yra atliekamas žymeklių (angl. *Markers*), esančių VR akiniuose ar VR valdikliuose, pagalba. Dažnai yra naudojami infraraudonųjų spindulių žymekliai ir kameros, galinčios aptikti tokio tipo skleidžiamą šviesą [64].

1.7. Analizės apibendrinimas

Renkantis dirbtinio intelekto metodus skirtus žmogaus veiklai atpažinti, buvo nuspręsta naudoti konvoliucinius neuroninius tinklus kuriamai VRR sistemai, dėl šių tinklų gebėjimo iš pateiktų duomenų išgauti giliąsias ypatybes ir suskirstyti jas į skirtingus veiklos tipus.

VRR sistemoje renkamų duomenų tvarkymui buvo pasirinktas duomenų normalizavimo metodas, kurio naudojimas gali padinti mašininio mokymosi modelių tikslumą, sumažinti duomenų perteklių, padidinti modelių mokymosi našumą bei sumažinti perkrovimo riziką.

Renkantis rankų judesių sekimo technologijas buvo nuspręsta naudoti *LiDAR* ir optinio sekimo technologijos kartu, tam kad kuriamoje VRR sistemoje būtų užtikrintas aukšto tikslumo rankų judesių sekimas. Šias technologijas naudojantys *Meta Quest 2* ir *Leap Motion Controller* įrenginiai bus taikomi kuriamos virtualios realybės sistemai, kuri naudos „iš vidaus į išorę“ naudotojo judesių sekimo metodą.

2. VRR sistemos projektavimas

2.1. Programos vystymo etapai

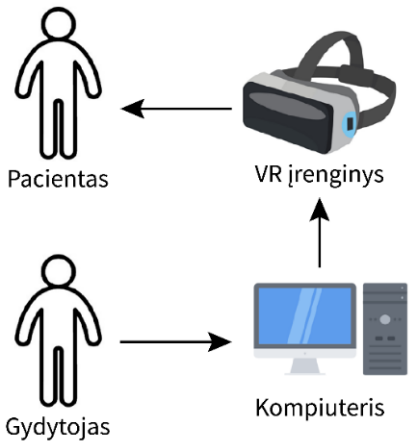
2.1 lentelė pateikia suplanuotus virtualiosios realybės reabilitacijos programos vystymo etapus: planavimą, reikalavimų specifikavimą, projektavimą, mašininio mokymosi modelio kūrimą ir VR sistemos kūrimą.

2.1 lentelė. VRR programos vystymo etapai

Etapas	**Aprašymas**
Planavimas, reikalavimų specifikavimas	Įvardinamas kuriamos programos kontekstas, kokybės vertinimo kriterijai, apribojimai ir kokie funkciniai, ne funkciniai reikalavimai jai yra keliami.
Projektavimas	Nusakoma kuriamos programos architektūra pasinaudojus *MagicDraw* įrankio pagalba.
VR sistemos aplinkos ir funkcijų kūrimas	Sukuriama programa naudojant *Unreal Engine* žaidimų variklį, *Adobe* programų paketus ir *Blender* 3D modeliavimo programinę įrangą.
Mašininio mokymosi modelio kūrimas	Sukuriamas mašininio mokymosi modelis, kuris klasifikuoja virtualioje realybėje atliktus judesius į veiklų klases. Modelis kuriamas naudojantis programa *Jupyter Notebook, Google Colaboratory* ir *Python* programavimo kalba.

2.2. Sistemos kontekstas

Gydytojas paleidžia asmeniniame kompiuteryje išsaugotą virtualiosios realybės reabilitacijos programą ir informuoja pacientą apie pratimą, kurį pacientas turės atlikti virtualioje aplinkoje. Aplinka pacientui yra pasiekiama per virtualiosios realybės įrenginį. Aplinkoje galima judėti ir sąveikauti su aplinkos elementais. Pacientui atliekant pratimą keisis jo virtualių rankų pozicija ir orientacija virtualioje aplinkoje, pagal paciento rankų judėjimą realioje aplinkoje. Pacientui atlikus pratimą kompiuteryje yra išsaugomas atpažintos veiklos rezultatas, nurodantis paciento veiklos klasę ir tikimybę priklausyti šiai klasei. Atpažintos veiklos rezultatą gali matyti pacientas virtualioje realybėje ir gydytojas kompiuterio monitoriuje. Gydytojas gali peržiūrėti ankščiau atliktų paciento reabilitacijos užsiėmimų seansų įvertinimus išsaugotus asmeniniame kompiuteryje. Virtualios realybės reabilitacijos programos veikimas pavaizduotas 2.1 pav.



2.1 pav. Sistemos veikimo kontekstas

2.3. Programos kokybės vertinimas

VRR programos kūrimo pabaigoje atlikto tyrimo metu bus įvertinama jos kokybė pagal 2.2 lentelėje paminėtus kriterijus, tokius kaip panaudojamumą, reikalingą protinio ir fizinio darbo krūvį bei įtraukimą į VR aplinką vertinimas.

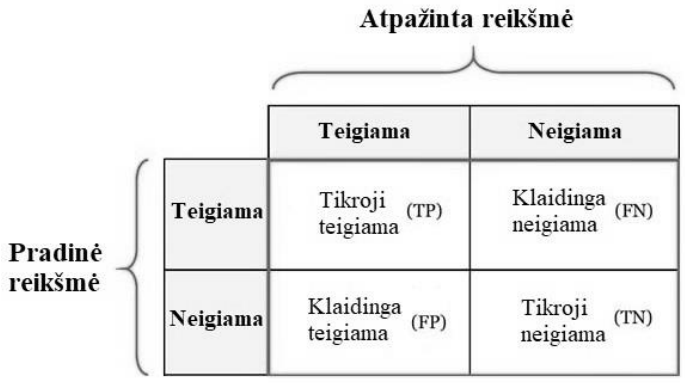
2.2 lentelė. Kokybės vertinimas

Kriterijus	**Pagrindimas**
Panaudojamumas	Kaip greitai naudotojas, neturintis daug patirties su virtualios realybės įranga ir VR programomis, sugebės išmokti dirbti su programa ir teigiamai ją įvertins.
Interaktyvumas	Ar naudotojui programos virtuali aplinka yra įdomi ir įtraukianti.
Protinio ir fizinio darbo krūvis	Kiek sukurta programa naudotojo reikalavo mąstyti, spręsti ir fiziškai judėti.

2.4. Mašininio mokymosi modelio vertinimo metrikos

CNN modelio, naudojamo suformuoti paciento atliktų pratimų rezultatus, įvertinimas bus atliekamas pagal šias metrikas: tikslumą (angl. *Accuray*), preciziškumą (angl. *Precision*), atkūrimą (angl. *Recall*) ir maišos matricą (angl.*Confusion matrix*).

Maišos matrica yra orginalių reikšmių ir modelio nuspėtų reikšmių lentelė. Maišos matrica gali padėti pamatyti reikšmes, kurias modelis dažnai atpažįsta neteisingai. Maišos matricoje yra naudojami “TP” (angl. *True positive*), “FN” (angl. *False negative*), “FP” (angl. *False positive*) ir “TN” (angl. *True negative*) reikšmių tipai pavaizduoti 2.2 pav. Šie tipai žymi pradines ir modelio atpažintas reikšmes, kurios gali būti skaidomos į teigiamas ir neigiamas. Maišos matricoje naudojami reikšmių tipai taip pat yra taikomi tikslumo, preciziškumo ir atkūrimo metrikų formulėse.



2.2 pav. Maišos matricos struktūra

Tikslumo metrika nusako duomenų rinkinio dalį, kurią modelis teisingai atpažino ir yra apskaičiuojama pagal (2.1) formulę [67]. Tikslumo metrika yra tinkama naudoti modelio vertinimui, kai duomenų rinkinys, kuris buvo naudotas modelio treniravimui, yra subalansuotas taip, kad vienodas kiekis duomenų priklausytų visoms rinkinio klasėms. Preciziškumo ir atkūrimo metrikos yra rekomenduojamos naudoti modelio vertinimo metu, kai modeliui treniruoti buvo naudojamas nesubalansuotas duomenų rinkinys, nes šios metrikos yra atskirai skaičiuojamos kiekvienai duomenų rinkinio klasei.

$$\text{Tikslumas} = \frac{TP+TN}{TP+TN+FP+FN}; \quad (2.1)$$

čia TP – tikrųjų teigiamų reikšmių kiekis; TN – tikrųjų neigiamų reikšmių kiekis; FP – klaidingų teigiamų reikšmių kiekis; FN – tikrųjų neigiamų reikšmių kiekis.

Preciziškumo metrika nusako duomenų rinkinio tikrųjų teigiamų (TP) reikšmių dalį, kurią modelis teisingai atpažino iš tikrųjų teigiamų (TP) ir klaidingų teigiamų (FP) reikšmių. Preciziškumo metrika dažnai yra naudojama įvertinti modelio gebėjimą teisingai atpažinti duomenų rinkinio reikšmes priklausančias vienai klasei. Preciziškumas apskaičiuojamas pagal (2.2) formulę [67]:

$$\text{Preciziškumas} = \frac{TP}{TP+FP}; \quad (2.2)$$

čia TP – tikrųjų teigiamų reikšmių kiekis; FP – klaidingų teigiamų reikšmių kiekis.

Atkūrimo metrika nusako duomenų rinkinio tikrųjų teigiamų (TP) reikšmių dalį, kurią modelis teisingai atpažino iš tikrųjų teigiamų (TP) ir klaidingų neigiamų (FP) reikšmių. Atkūrimas matuoja modelio tikslumą teisingai atpažinti pradines teigiamas reikšmes ir yra apskaičiuojamas pagal (2.3) formulę [67]:

$$\text{Atkūrimas} = \frac{TP}{TP+FN}; \quad (2.3)$$

čia TP – tikrųjų teigiamų reikšmių kiekis; FN – klaidingų teigiamų reikšmių kiekis.

2.5. Apribojimai

Šiame poskyryje yra iškeliami sistemai taikomi bendri apribojimai ir įvardinami įrankiai pasirinkti sistemos kūrimui.

2.5.1. Bendri apribojimai

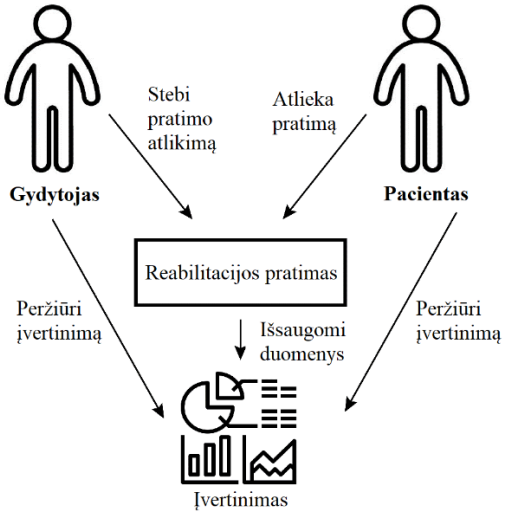
Buvo išsikelti bendri apribojimai kuriamai reabilitacijos programai. Vienas iš apribojimų yra kad, kuriama reabilitacijos programa turi būti palaikoma „Microsoft Windows 10“ operacinės sistemos platformoje ir veikti asmeniniuose kompiuteriuose. Įvykus naudotojo sąveikai su programos trimatės virtualios aplinkos elementais, programa turi sureaguoti ne ilgiau nei per 0,5 sekundės, nes ilgesnis reagavimo laikas priverčia naudotoją laukti ir gali sudaryti nesklandaus programos veikimo įspūdį. Taip pat, programa turi veikti šešis laisvės laipsnius palaikančiuose *Oculus* virtualios realybės įrenginių platformose. Virtualios realybės įrangos naudojimui yra reikalingas 2,25 m^2 pratimų atlikimo plotas patalpoje.

2.5.2. Naudojami programinės įrangos paketai

Sistema yra kuriama naudojantis *Unreal Engine* žaidimų varikliu. Trimačiai grafiniai elementai yra modeliuojami naudojantis atviro kodo programine įranga *Blender*, o šie elementai yra tektūruojami naudojantis *Adobe Substance 3D Painter* programinės įrangos pagalba. Programų *Jupyter Notebook, Google Colaboratory* ir *Python* programavimo kalbos pagalba yra kuriama konvoliucinio neuroninio tinklo architektūra, vykdomas tinklo apmokymas ir atliekamas VR sistemos užregistruotų duomenų klasifikavimas.

2.6. Veiklos sudėtis

2.3 paveikslas pavaizduoja kuriamos programos veiklos konteksto diagramą. Šiame paveiksle pavaizduotos veiklos yra pateikiamos veiklos įvykių sąraše 2.3 lentelėje. Lentelė nurodo, kokie yra kiekvienos veiklos įeinantis arba išeinantis informacijos srautai.



2.3 pav. Veiklos kontekstas

2.3 lentelė. Veiklos įvykių sąrašas

Eil. Nr.	Įvykio pavadinimas	Įeinantys/Išeinantys informacijos srautai
1.	Pacientas atlieka pratimą	Paciento judėjimo duomenys *(in)*
2.	Gydytojas stebi pratimo atlikimą	Paciento matomas vaizdas virtualioje aplinkoje *(out)*
3.	Gydytojas ir pacientas peržiūri pratimo įvertinimą	Pratimo atlikimo metu sugeneruotų duomenų klasifikavimo rezultatas*(out)*

2.6.2. Sistemos panaudojamumo ir išvaizdos reikalavimai

Programa turi būti lengvai suprantama ir naviguojama tų žmonių, kurie turi mažai patirties su darbalaukio programomis. Virtualios realybės sąsaja turi būti lengva naudoti tiems žmonėms, kurie neturi patirties naudojant virtualios realybės įrangą.

Programos virtuali aplinka yra pakankamai tikroviška ir įtraukianti, kad programos naudotojas galėtų pritaikyti išmoktus su judėjimu susijusius įgūdžius kasdienybėje, pasiekti pažangą motorinių ir kognityvinių įgūdžių reabilitacijos procese.

2.7. Funkciniai ir nefunkciniai reikalavimai

Prieš pradedant virtualios realybės reabilitacijos programos kūrimo procesą, svarbu žinoti jai keliamus reikalavimus. Reikalavimai nurodo, kokias funkcijas turi atlikti programa ir pagal kokius kriterijus ji turėtų veikti.

Funkciniai reikalavimai:

- Pacientas gali judėti virtualios realybės aplinkoje.
- Pacientas gali atlikti reabilitacijos pratimus virtualioje realybėje.

- Pacientas gali sąveikauti su 3D objektais virtualioje realybėje.
- Gydytojas gali peržiūrėti išsaugotus duomenis apie paciento veiklos įvertinimą asmeniname kompiuteryje.
- Gydytojas gali stebėti pratimų atlikimą kompiuterio ekrane.

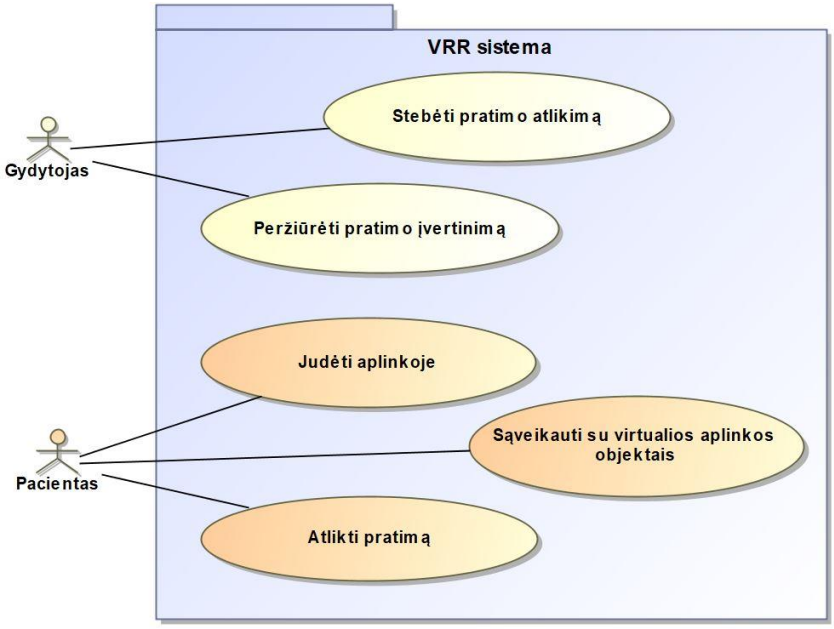
Nefunkciniai reikalavimai:

- Programa turi būti optimizuota, kad veiktų optimaliu kadrų dažniu (70–90 Hz).
- Virtualios realybės programoje turi būti rankų sąveikos mechanizmas.
- Naudotojas gali sąveikauti su objektais VR aplinkoje juos sugriebdamas ir liesdamas.
- VR sąsają lengva suprasti ir naudoti tiems, kurie neturi VR įrangos naudojimo patirties.
- Pratimus pacientas gali atlikti sėdėdamas.
- Nėra apribojimo, per kiek laiko pacientas gali atlikti vieną pratimą.
- Pacientas gali pataisyti savo veiksmus, jei buvo padaryta klaida pratimo atlikimo metu.
- Duomenų klasifikavimo rezultatas kompiuteryje yra išsaugomas kaip tekstinis failas.
- Pacientas gali atlikti pratimus tiek dešine, tiek kaire ranka.
- Pacientas gali matyti savo rankas virtualioje aplinkoje.

2.8. Sistemos sudėtis

2.8.1. Sistemos ribos ir panaudojimo atvejai

2.4 paveiksle pavaizduota panaudojimo atvejų diagrama, kuri apibūdina programą iš naudotojo perspektyvos ir jai iškeltus funkcinius reikalavimus.



2.4 pav. VRR sistemos panaudojimo atvejų diagrama

2.8.2. Panaudojimo atvejų sąrašas

Virtualios realybės reabilitacijos programos panaudojimo atvejų aprašymai pateikti šiame skyrelyje lentelių forma. Kiekviena lentelė apibūdina atskirų panaudojimo atvejų naudotoją (aktorių), aprašymą, prieš, po ir sužadinimo sąlygas.

2.4 lentelė. Atliekamo pratimo stebėjimo panaudos atvejo aprašymas

PANAUDOJIMO ATVEJIS: Stebėti pratimo atlikimą	
Naudotojas / Aktorius:	Gydytojas
Aprašas:	Apima procesą, kurio metu naudotojas stebi pratimo atlikimą kompiuterio ekrane.
Prieš sąlyga:	Naudotojas yra paleidęs programą savo asmeniniame kompiuteryje.
Sužadinimo sąlyga:	Naudotojas nori stebėti kaip kitas naudotojas / pacientas atlieka pratimą.
Po–sąlyga:	Pratimo įvertinimas yra sugeneruojamas.

Atliekamo pratimo stebėjimo operacija leidžia gydytojui per kompiuterio ekraną stebėti vaizdą kurį mato pacientas iš pirmo asmens perspektyvos (2.4 lentelė). Ši operacija gali įvykti tik gydytojui paleidus VRR programą savo asmeniniame kompiuteryje. Paciento atliekamos veiklos stebėjimas leis gydytojui matant paciento regimą vaizdą padėti jam geriau suvokti virtualią aplinką, pakonsultuoti dėl teisingo pratimo atlikimo.

2.5 lentelė. Pratimo atlikimo įvertinimo peržiūros panaudos atvejo aprašymas

PANAUDOJIMO ATVEJIS: Peržiūrėti pratimo įvertinimą	
Naudotojas / Aktorius:	Gydytojas
Aprašas:	Apima procesą, kurio metu naudotojas peržiūri kompiuterio atmintyje išsaugotą pratimo įvertinimą.
Prieš sąlyga:	Kitas naudotojas / pacientas atliko pratimą.
Sužadinimo sąlyga:	Naudotojas nori peržiūrėti pratimo įvertinimą.
Po–sąlyga:	Pratimo įvertinimas yra išsaugomas tekstiniame faile.

Pratimo įvertinimo peržiūros operacija leidžia gydytojui peržiūrėti asmeniniame kompiuteryje išsaugotą paciento atlikto pratimo įvertinimą (2.5 lentelė). Operacija gali įvykti tik pacientui baigus pratimą ir įvykus paciento judėjimo virtualioje aplinkoje duomenų išsaugojimui. Šie duomenys yra išsaugomi PNG formato failu asmeniniame kompiuteryje kartu su pratimo įvertinimu išsaugotu TXT formato failu.

2.6 lentelė. Judėjimo aplinkoje panaudos atvejo aprašymas

PANAUDOJIMO ATVEJIS: Judėti aplinkoje	
Naudotojas / Aktorius:	Pacientas
Aprašas:	Apima procesą, kurio metu naudotojas virtualioje aplinkoje gali keisti savo rankos poziciją fiziškai kiek jam leidžia jo fizinė aplinka ir su VR įranga nustatytos ribos.
Prieš sąlyga:	Naudotojas per virtualios realybės įrangą mato virtualią aplinką.
Sužadinimo sąlyga:	Naudotojas nori keisti savo rankos poziciją virtualioje aplinkoje, tam kad atliktų reabilitacijos pratimą.
Po–sąlyga:	Naudotojas pakeitė savo rankos poziciją pratimo atlikimo metu.

Judėjimo virtualios realybės aplinkoje funkcija leidžia pacientui savo fizinėje aplinkoje atliekamą rankos vietos ir pasisukimo pozicijos pasikeitimą matyti VRR programos aplinkoje (2.6 lentelė). Operacija gali įvykti tik gydytojui paleidus reabilitacijos programą ir pacientui matant virtualią aplinką per virtualios realybės įrangą. Judėjimo aplinkoje funkcija yra reikalinga pacientui atlikti reabilitacijos pratimą.

2.7 lentelė. Sąveikavimo su aplinkos objektais panaudos atvejo aprašymas

PANAUDOJIMO ATVEJIS: Sąveikauti su aplinkos objektais	
Naudotojas / Aktorius:	Pacientas
Aprašas:	Apima procesą, kurio metu naudotojas virtualioje aplinkoje gali liesti, sugriebti ir keisti trimačių objektų vietos poziciją virtualioje erdvėje.
Prieš sąlyga:	Naudotojas gali judėti virtualioje aplinkoje.
Sužadinimo sąlyga:	Naudotojas nori keisti objektų poziciją virtualioje aplinkoje, tam kad atliktų reabilitacijos pratimą.
Po–sąlyga:	3D objektas virtualioje aplinkoje pakeitė savo vietos ir pasisukimo poziciją.

Gebėjimo sąveikauti su 3D aplinkos objektais operacija pacientui leidžia liesti, sugriebti, pernešti virtualius objektus aplinkoje. Pratimo atlikimo įvertinimas yra sudaromas atsisžvelgiant į tai, kaip pacientas keičia aplinkos objektų vietos ir pasisukimo padėtis, todėl ši operacija yra viena svarbiausių naudotojui (2.7 lentelė).

2.8 lentelė. Pratimo atlikimo panaudos atvejo aprašymas

PANAUDOJIMO ATVEJIS: Atlikti pratimą	
Naudotojas / Aktorius:	Pacientas
Aprašas:	Apima procesą, kurio metu naudotojas virtualioje aplinkoje keisdamas savo rankos ir aplinkoje esančio objekto poziciją atlieka reabilitacijos pratimą.
Prieš sąlyga:	Naudotojas gali judėti virtualioje aplinkoje ir liesti, sugriebti, pajudinti aplinkoje esančius objektus.
Sužadinimo sąlyga:	Naudotojas nori atlikti reabilitacijos pratimą ir gali judėti virtualioje aplinkoje.
Po–sąlyga:	Naudotojas atliko reabilitacijos pratimą, pratimo atlikimo duomenys buvo užregistruoti.

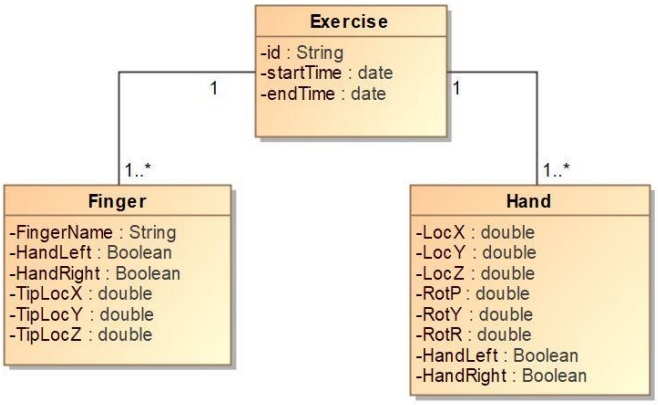
Pratimo atlikimo operacija leidžia virtualios realybės reabilitacijos programai surinkti paciento judėjimo aplinkoje duomenis ir atlikti pratimo įvertinimą. Šios operacijos sąlyga yra sužadinama jei pacientas gali judėti programos aplinkoje ir sąveikauti su joje esančiais trimačiais objektais (2.8 lentelė).

2.9. Reikalavimai duomenims

2.5 pav. pavaizduoja sukurtą duomenų klasių diagramą. Diagramos viršuje yra pavaizduota „Exercise“ klasė, kuri nurodo jog pratimo metu yra registruojami pratimo identifikacijos numerio, pradžios ir pabaigos laikų duomenys.

„Finger“ klasė saugo pratimo metu naudojamo piršto pavadinimą, jo galiuko vietos x, y ir z koordinačių ašyse reikšmes bei ranką, kuriai pirštas priklauso. „Hand“ klasė saugo naudotojo rankos objekto vietos pagal x y ir z koordinačių ašių reikšmes, rankos objekto pasisukimo laipsnių aplink x, y ir z koordinačių ašis reikšmes bei rankos, kuria atliekamas pratimas, reikšmes.

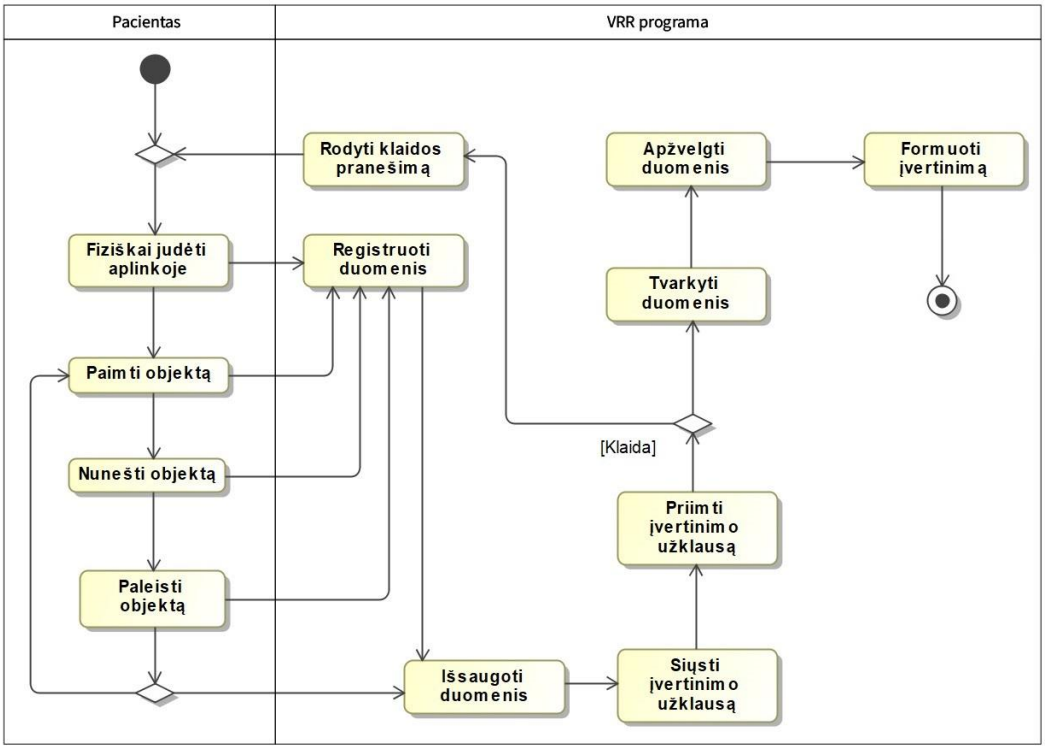
2.5 pav. pavaizduotoje diagramoje „Finger“ ir „Hand“ klasės yra sujungtos su „Exercise“ klase skaitiniais daugybiškumo (angl. *Multiplicity*) ryšiais. Toks klasių sujungimas nurodo objektų, galinčių dalyvauti sukurtame ryšyje tarp dviejų klasių, skaičių. „Finger“ ir „Hand“ klasių objektai gali priklausyti tik vienam „Exercise“ klasės objektui. „Exercise“ klasės objektas gali turėti vieną arba daugiau „Finger“ ir „Hand“ klasių objektų.



2.5 pav. Duomenų klasių diagrama

2.10. VRR programos dinaminis vaizdas

Šiame poskyryje yra pateikiamas sistemos dinaminis vaizdas pavaizduotas su UML veiklos diagrama [68]. Diagrama nurodo objektų atliekamas veiklas ir perėjimus tarp skirtingų veiklų, tam kad būtų galima įvykdyti iškeltus VRR programos funkcijų scenarijus.



2.6 pav. Pratimo įvertinimo formavimo veiklos diagrama

2.6 paveiklas vaizduoja pratimo atlikimo ir jo įvertinimo suformavimo veiklos diagramą. Gydytojui paleidos VRR sistemą ir pacientui matant pratimo vykdymo aplinką, pacientas gali fiziškai judėti, nešti aplinkos objektus tam, kad pratimas būtų pabaigtas. Pratimo atlikimo metu yra registruojami duomenys, kurie vėliau yra išsaugomi, tvarkomi ir apžvelgiami, tam kad būtų galima suformuoti pratimo atlikimo įvertinimą ir jį išsaugoti. Klaidos pranešimas veiklos diagramoje vaizduoja atsiradusią klaidą susijusią su duomenų tvarkymu.

3. Sistemos kūrimo priemonės ir metodai

3.1. Vartotojo sąsajos grafika

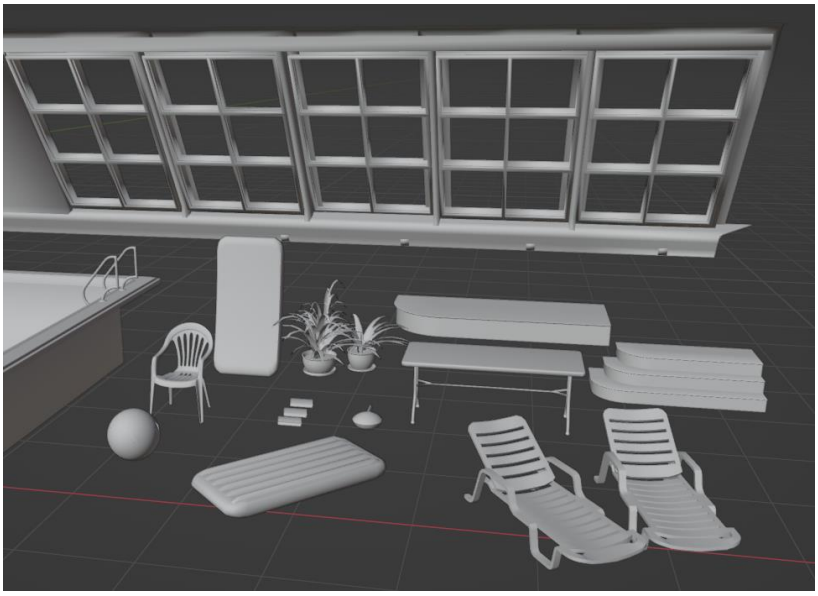
Virtualios realybės aplinkų grafikos kūrimui yra reikalingi 3D objektai, tekstūrų rinkinių paveikslai ir medžiagos. Sudarant sistemos vartotojo sąsają buvo pasinaudota *Unreal Engine* žaidimų variklio medžiagų mazgų sistemomis, *Blender* ir *Krita* programinės įrangos paketais. Šiame poskyryje yra aprašomas atliktas trimačio modeliavimo, tekstūravimo, medžiagų kūrimo, modelių optimizavimo, UV išklotinių žemėlapių sudarymo, eksportavimo ir kolizijos geometrijų kūrimo procesas.

3.1.1. 3D modeliavimas

Trimačio objekto kūrimas, dar vadinamas 3D modeliavimu, yra svarbus VR sistemų grafikos kūrimo proceso etapas. Sistemos trimačių modelių kūrimui buvo pasinaudota atviro kodo 3D modeliavimo programa *Blender*. Ši programinė įranga gali būti naudojama 3D modelių kūrimui, jų importavimui ir eksportavimui dažniausiai žaidimų varikliuose naudojamais 3D objektų formatais *(*BLEND, OBJ ir FBX), modelių dydžio, vietos padėties, pasisukimo kampo 3D erdvėje, topologijos (modelio briaunų pasiskirstymo ir struktūros) redagavimui, modelių medžiagų ir tekstūrų paveikslų kūrimui [69].

Atliekant 3D modelių modeliavimą buvo pasinaudota *Blender* paketo redagavimo režimu, kuriame galima keisti modelių tinkelio tipo paviršių, kuris yra sudarytas iš individualių elementų: plokštumų (angl. *Faces*), briaunų (angl. *Edges*) ir viršūnių (angl. *Vertex*) [70]. Keičiant šių individualių elementų padėtis 3D erdvėje buvo išgauti tam tikros formos 3D objektai. Modeliavimo metu buvo panaudoti modifikatoriai (angl. *Modifiers*), kurie leidžia keisti objektų geometriją ne destruktyviu būdu ir nepakeičiant jos pradinės formos [71].

VRR sistemos grąfinės sąsajos formavimui buvo sukurti septyni trimačiai modeliai ir parsisiųsti du patalpos [R12] ir kėdės [R13] modeliai, kurie yra platinami *Creative Commons Attribution* licenzijos sąlygomis (žr. 3.1 pav.). 3D modeliai buvo imporuoti į *Blender* programinės įrankos paketo projektą FBX formatu. Parsisiųstas patalpos modelis apytiksliai turėjo 24 tūkstančius trikampių plokšumų, 14 tūkstančių viršūnių, 3 tekstūrų paveikslus ir 2 medžiagas. Kėdės 3D modelis apytiksliai turėjo 26,6 tūkstančius trikampių plokšumų, 13 tūkstančių viršūnių, 2 tekstūrų paveikslus ir vieną medžiagą.



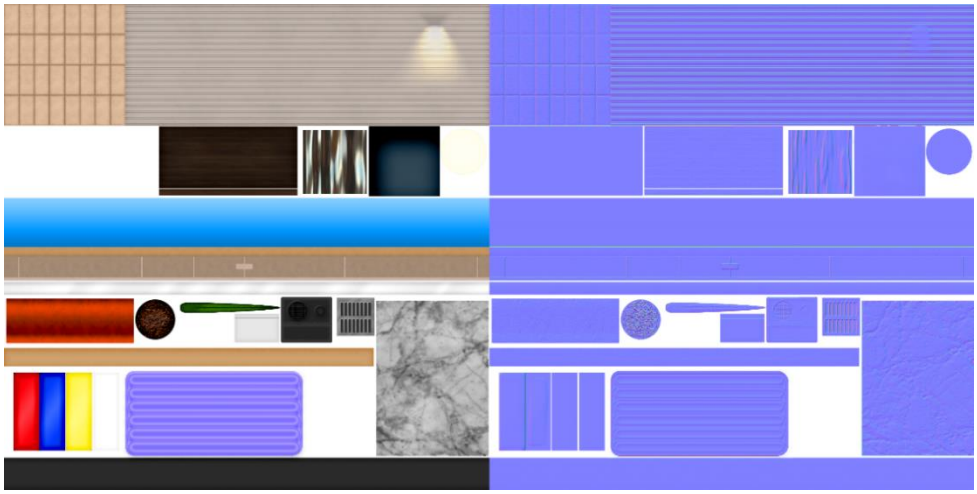
3.1 pav. Trimačiai modeliai *Blender* vartotojo sąsajoje

3.1.2. Optimizacijos metodų taikymas

VR programų kūrimo metu svarbu atsižvelgti į įrenginio atminties ir vaizdo plokštės apkrovimą. Sistemos greitaveika ir grafinės sąsajos atvaizdavimo dažnis gali būti žemas, jei yra naudojami aukšto detalumo trimatės grafikos modeliai ir jų elementai, tokie kaip medžiagos ir tekstūrų paveikslai [72]. Norint išlaikyti gerą kuriamos sistemos greitaveiką buvo svarbu panaudoti optimizacijos metodus, tokius kaip poligonų skaičiaus mažinimas ir tekstūrų paveikslų atlasų naudojimas.

Taikant trimačių modelių optimizacijos metodus, parsisiųstiems modeliams buvo mažinamas juos sudarančių poligonų skaičius. Buvo panaikintos topologijos plokštumos, kurios nepateks į sistemos naudotojo matymo lauką, taip pat panaikintos modelių paviršiaus viršūnės, kurios nebuvo būtinos norimai modelių formai išgauti.

Tekstūrų paveikslų atlasų optimizavimo metodas yra skirtas 3D scenoje naudojamų modelių medžiagų skaičiaus mažinimui. Taikant šį metodą yra naudojami tekstūrų paveikslų atlasai, kurie įprastai yra skirti vienai medžiagai sudaryti. Ši medžiaga gali būti naudojama daugiau nei vieno modelio atvaizdavimui [73]. Naudojant tekstūrų paveikslų atlasų optimizacijos metodą, kėdės modelio medžiaga buvo panaikinta ir jai priskirta patalpos modelio medžiaga, kuri naudoja dviejų tekstūrų paveikslų atlasą (žr. 3.2 pav.), reikalingą objekto medžiagos spalvos parametro ir plokštumų normalių informacijai užpildyti.



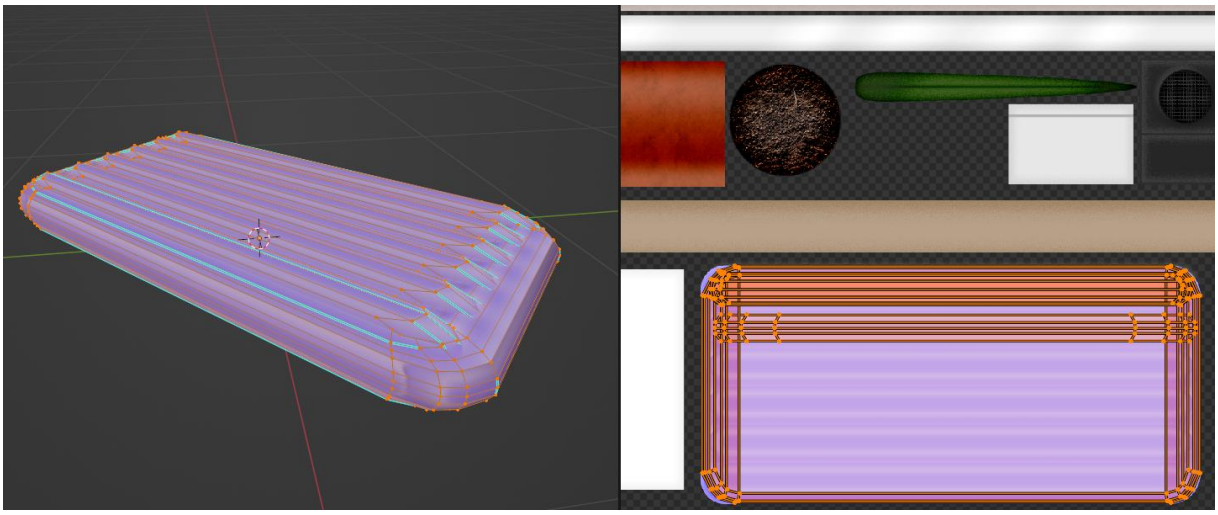
3.2 pav. 3D modelio tekstūrų paveikslų atlasas

3.1.3. 3D objektų medžiagos

Praeitame skyrelyje aprašytam tekstūrų paveikslų atlasui buvo atlikta spalvų korekcija pasinaudojus skaitmeninės tapybos programinės įrangos paketu *Krita*. Šis paketas taip pat buvo panaudotas trijų modelių tekstūros paveikslo sukūrimui. Šiame tekstūros paveiksle yra patalpinta medžiagos atspalvio informacija.

Visiems modeliamas buvo atliktas UV (dvimačio paveikslo u ir v koordinačių ašių) išklotinių žemėlapių sudarymas (žr. 3.3 pav.). Tokio tipo žemėlapiai yra reikalingi tinkamam tekstūrų atvaizdavimui ant 3D modelio. UV išklotinė yra gaunama pažymint modelio paviršių briaunas siūlėmis (angl. *Seams*), taip kad objekto paviršių būtų galima išskleisti ant plokštumos, kitaip vadinamos dvimačiu paveikslu [74]. Modelių išklotinės ant UV žemėlapio buvo išskirstytos pagal tekstūrų

paveikslo atlasą, taip kad skirtingos modelio dalys turėtų skirtingus spalvos parametrus, bet naudotų tik vieną medžiagą.



3.3 pav. 3D modelis ir jo UV išklotinė patalpinta ant tekstūrų paveikslų atlaso

3.1.4. Modelių paruošimas žaidimų varikliui

Modeliams buvo parinkti pradžios taškai (angl. *Origin point*), kurie nurodė objekto vietą trimatėje erdvėje. Objektams, kuriuos sistemos naudotojas galėsi kilnoti virtualioje realybėje pradžios taškai buvo nurodyti objektų tūrio centruose. Kitiems objektams buvo pritaikomi pradžios taškai paviršiaus apačioje, dėl patogesnio objektų talpinimo į žaidimų variklio trimatę sceną. Modeliai ir jų elementai buvo eksportuojami FBX failų formatu ir importuojami į *Unreal Engine* žaidimų variklį sistemos scenos sudarymui.

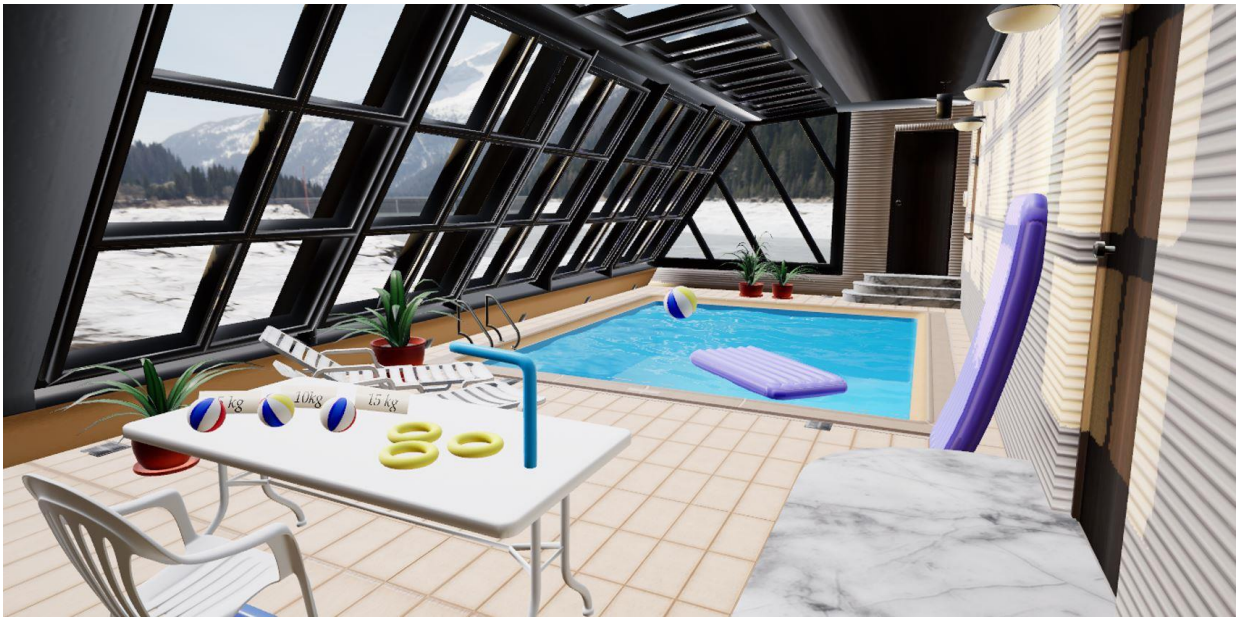
3.1.5. Kolizijos geometrijų kūrimas

Kolizijos geometrijos yra naudojamos išvengti trimačių objektų persidengimo. Iškilios geometrijos trimačių objektų kolizijos gali būti automatiškai generuojamos *Unreal Engine* žaidimų variklyje. Tačiau įgaubtų objektų kolizijos geometrijos generavimas nepakankamai tiksliai atkartoja objekto formą, todėl sistemos interaktyvumas gali būti neišpildytas. *Unreal Engine* turi įrankius kurti kolizijos geometrijas rankiniu būdu [75]. Tam įprastai yra naudojami stačiakampio gretasienio formos objektai, kurių mastelį, vietą ir pasisukimą trimatėje erdvėje galima koreguoti ir išgauti norimą kolizijos formą.

Daugumos grafinės sąsajos 3D objektų, tokių kaip kamuolys, plaustas ar durys, kolizijos geometrijos buvo sugeneruotos automatiškai žaidimų variklyje, tačiau išgaubtų objektų geometrijas pasirinkta kurti rankiniu būdu su *Blender* programinės įrangos paketu. Kolizijos geometrijos buvo eksportuojamos iš *Blender* programinės įrangos paketo kartu su 3D modeliu, kuriam kolizija buvo kuriama. Kolizijos geometrijoms reikėjo nurodyti atitinkamus pavadinimus, kad žaidimų variklis galėtų jas atpažinti. Pavyzdžiui, “Torus” 3D modelio kolizijos geometrijos objektui suteikus pavadinimą „UCX_Torus_01“, *Unreal Engine* žaidimų variklis atpažins šį objektą, kaip pirmąją kolizijos geometriją “Torus” modeliui.

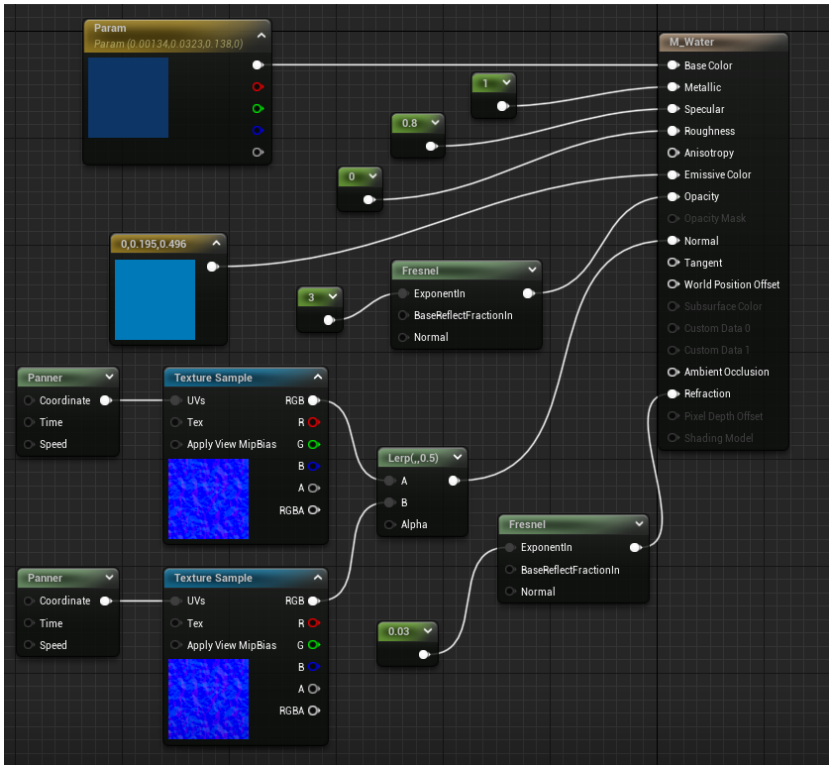
3.2. 3D scenos kūrimas

Atlikus 3D modelių ir jų elementų eksportavimą į *Unreal Engine* žaidimų variklį, objektai buvo talpinami trimatėje scenoje (žr. 3.4 pav.). Objektams buvo pritaikomos *Unreal Engine* mazgų sistemoje suformuotos medžiagos.



3.4 pav. VRR programos scena

“M_Water” medžiagai norint išgauti bangavimo efektą buvo panaudotas dviejų normalių žemėlapio mazgų persidengimas *Panner* mazgo nurodytu greičiu. Sukurtos vandenį imituojančios medžiagos mazgų sistema yra pavaizduota 3.5 pav.



3.5 pav. “M_Water” medžiagos mazgų sistema

Scenos apšvietimui sukurti buvo panaudoti trijų tipų šviesos komponentai: „Directional Light“, „Point Light“ ir „HDRI Backdrop“. Kryptinės šviesos komponentui „Directional Light“ buvo parinkti stacionarumo, spalvos, krypties, 4500 K temperatūros ir 4 lx intensyvumo parametrai. Stacionarus kryptinės šviesos komponentas negali keisti vietos padėties 3D aplinkoje. Šio komponento taikymas scenoje nurodo programai šešėlius generuoti iš anksto o ne realiu laiku. Taškiniam šviesos komponentui „Point Light“ buvo parinktas 5 cd šviesos šaltinio stipris, spalva ir statinis judėjimas. „HDRI Backdrop“ šviesos komponentui buvo parinktas HDR paveikslas [R14], šviesos atstumo faktorius (angl. *Lightning distance factor*), 152 m HDR paveikslą projektuojančios geometrijos dydis ir 1 cd/m^2 intensyvumas.

Scenos objektams, kuriuos naudotojas galės paimti, buvo pridėtas *Grab* komponentas bei parinkti mobilumo, fizikos simuliavimo, masės, gravitacijos ir kolizijos persidengimo parametrai. Įjungtas kolizijos persidengimo parametras nurodo dviejų objektų persidengimo įvykio metu nustatyti tam tikrą šių objektų elgseną.

3.3. Naudotojo objekto įvestis

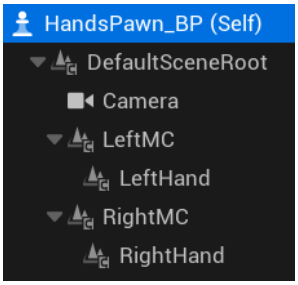
Šiame poskyryje yra aprašomas sistemai iškeltų funkcinių ir nefunkcinių reikalavimų išpildymas.

3.3.1. Rankų sąveikos mechanizmas

VRR programos kūrimui ir testavimui buvo pasirinktas *Meta Quest 2* įrenginys, kuris turi *Inside–out* ir rankų sekimo mechanizmus. *Meta* kompanijos VR įrenginių platforma *Oculus* yra palaikoma *Unreal Engine* žaidimų variklyje. Aplikacijų, kurios naudoja minėtus *Meta Quest 2* įrenginio sekimo mechanizmus, kūrimui su *Unrea Engine* yra reikalingas *Oculus VR* arba *OpenXR* įskiepis. Kai kūrimui yra naudojamas *OpenXR* įskiepis, aplikacijos gali būti paleidžiamos bet kuriame VR įrenginyje, kuris naudoja *OpenXR* taikomųjų programų programavimo sąsają. Jei aplikacijos kūrimui naudojamas *Oculus VR* įskiepis, ji galės būti paleista tik *Oculus* platformoje [76].

Sistemos naudotojo rankų judesių sekimui taip pat buvo naudojamas *Ultraleap* įmonės *Leap Motion Controller* įrenginys. Šio įrenginio surinktų duomenų naudojimui *Unreal Engine* žaidimų variklyje buvo instaliuotas *Ultraleap Tracking* įskiepis [77].

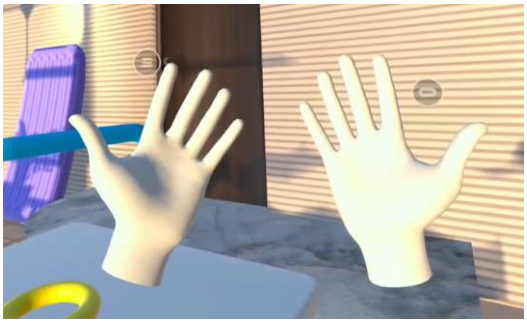
Sistemos levelio, kuris naudos *Oculus* programavimo sąsają, paruošimo metu buvo įjungtas rankų sekimo palaikymas *Oculus VR* įskiepio nustatymų skydelyje. Naudotojo objekto įvesties sukūrimui buvo panaudota *Blueprint* klasė „HandsPawn_BP“. Šiai klasei buvo nurodyta *Pawn* tipo tėvinė klasė. *Pawn* klasė *Unreal Engine* žaidimų variklyje nurodo aktoriaus objektą, kurį programos naudotojas galės kontroliuoti. *Pawn* klasė yra fizinė programos naudotojo reprezentacija, kuri nurodo naudotojo sąveiką su aplinkos objektais [78].



3.6 pav. „HandsPawn_BP“ klasę sudarantys komponentai

„HandsPawn_BP“ klasei buvo sukurti „Camera“, „LeftMC“, „LeftHand“, „RightMC“, „RightHand“ komponentai, turintys kilnojamo (angl. *Movable*) mobilumo parametrą (žr. 3.6 pav.). „Camera“ komponentas nurodo fotoaparato peržiūros tašką ir nustatymus, tokius kaip projekcijos tipas ir matymo laukas. Numatytasis aktorius naudos šio komponento vietos ir pasisukimo informaciją [79]. „LeftMC“ ir „RightMC“ yra dešinės ir kairės rankos judesio valdikliai. „LeftHand“ ir „RightHand“ yra *Oculus* įskiepio dešinės ir kairės rankos komponentai, jų tėviniai komponentai yra „LeftMC“ ir „RightMC“ valdikliai.

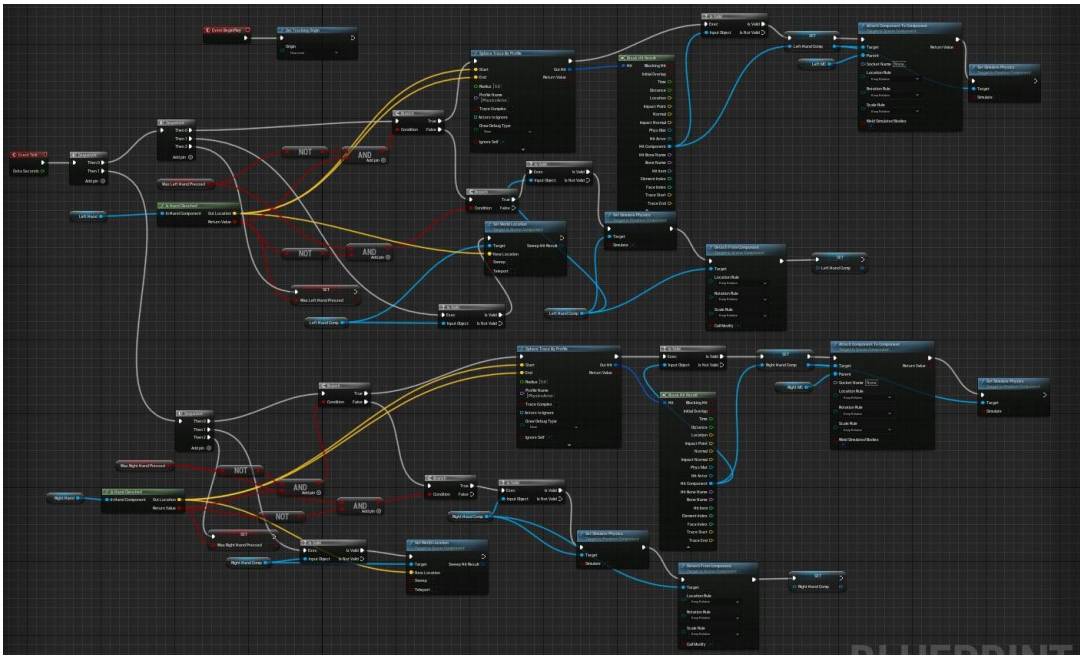
„LeftHand“ ir „RightHand“ komponentams (žr. 3.7 pav.) buvo sukurta „M_Hands“ medžiaga su atitinkamais spalvos ir šiurkštumo (angl. *Roughness*) parametrais. Medžiagai reikėjo pritaikyti funkciją, kuri nurodo medžiagos naudojimą su skeleto objektu (angl. *Skeletal mesh*), tam kad būtų galima naudoti šią medžiagą su *Oculus* įskiepio dešinės ir kairės rankos komponentais.



3.7 pav. Naudotojo rankų atvaizdavimas

3.3.2. Sąveika su aplinkos objektais

Veikėjo objekto sąveikai su aplinkos objektais sukurti buvo panaudota *Unreal Engine* žaidimų variklio mazgų sistema. Ji leido „HandsPawn_BP“ klasei sukurti įvykių grafiką, pavaizduotą 3.8 paveiksle. Įvykių grafike yra panaudotos funkcijos, kurios tikrina ar buvo atliktas rankos pirštų suspaudimas ir atleidimas, kai yra paimamas ar paleidžiamas VR aplinkos objektas.



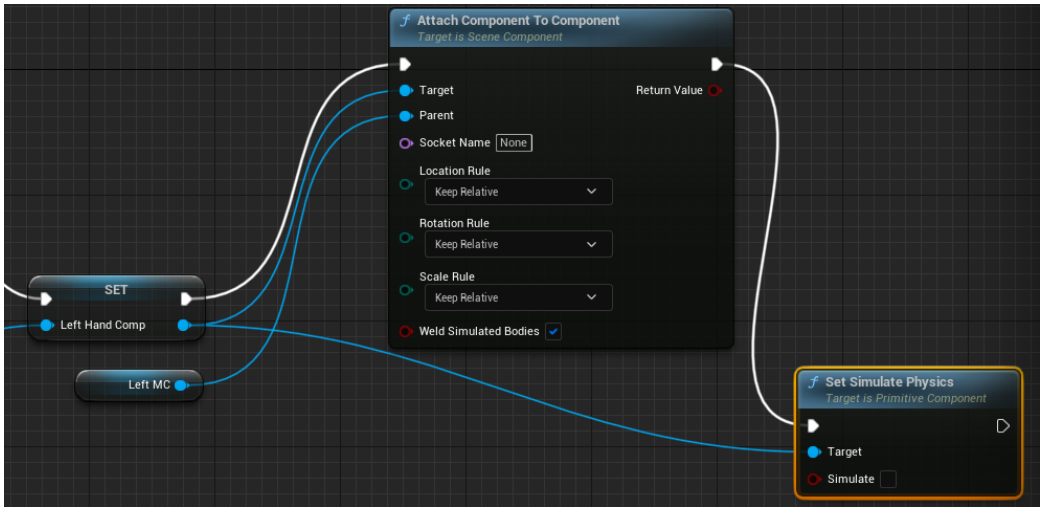
3.8 pav. „HandsPawn_BP“ klasės įvykių grafikas

Pasinaudojus C++ programavimo kalba buvo sukurta „IsHandClenched“ (žr. 3.3 pav.) funkcija, kuri rankos pirštų galiukų vietos trimatėje erdvėje informaciją naudoja rankos suspaudimo ir atleidimo įvykių nustatymui. Funkcijos iškvietimo metu yra surandamas mažylio, bevardžio, didžiojo ir smiliaus pirštų galiukų vietos vidurkis, kuris yra palyginamas su nykščio piršto galiuko vietos reikšme. Jei atstumas tarp nykščio ir kitų pirštų yra mažesnis nei 5 cm, ranka yra laikoma suspausta. Jei šis dydis mažesnis – ranka atleista.

```
bool UHandsFunctionLibrary::IsHandClenched(class UOculusHandComponent* InHandComponent, FVector& OutLocation)
{
    FName ThumbName = FName(UOculusInputFunctionLibrary::GetBoneName(EBone::Thumb_Tip));
    FName IndexName = FName(UOculusInputFunctionLibrary::GetBoneName(EBone::Index_Tip));
    FName MiddleName = FName(UOculusInputFunctionLibrary::GetBoneName(EBone::Middle_Tip));
    FName RingName = FName(UOculusInputFunctionLibrary::GetBoneName(EBone::Ring_Tip));
    FName PinkyName = FName(UOculusInputFunctionLibrary::GetBoneName(EBone::Pinky_Tip));

    FVector ThumbTipLoc = InHandComponent->GetBoneLocationByName(ThumbName, EBoneSpaces::WorldSpace);
    FVector IndexTipLoc = InHandComponent->GetBoneLocationByName(IndexName, EBoneSpaces::WorldSpace);
    FVector MiddleTipLoc = InHandComponent->GetBoneLocationByName(MiddleName, EBoneSpaces::WorldSpace);
    FVector RingTipLoc = InHandComponent->GetBoneLocationByName(RingName, EBoneSpaces::WorldSpace);
    FVector PinkyTipLoc = InHandComponent->GetBoneLocationByName(PinkyName, EBoneSpaces::WorldSpace);

    FVector FingersLoc = (ThumbTipLoc + IndexTipLoc + MiddleTipLoc + RingTipLoc) / 4.0F;

    OutLocation = (ThumbTipLoc + IndexTipLoc + MiddleTipLoc + RingTipLoc + PinkyTipLoc) / 5.0F;

    float Distance = FVector::Distance(ThumbTipLoc, FingersLoc);

    return Distance <= 5.0F;
}
```

3.9 pav. „IsHandClenched“ funkcija

„IsHandClenched“ funckija priima rankos komponentą kaip įvestį. Funckija grąžina rankos komponento vietos vektoriaus (angl. *Location vector*) x, y ir z koordinačių ašių reikšmes ir *Boolean* tipo kintamąjį, kuris nurodo suspaudimo įvykio buvimą. Vietos vektoriaus reikšmės „HandsPawn_BP“ klasės įvykių grafike yra naudojamos „Attach Component To Component“ funkcijos įvestyje. Ši funkcija nurodo rankos komponento priskyrimą kitam VR scenos komponentui, kai buvo užfiksuotas rankos suspaudimo įvykis. Toks priskyrimas nurodo, kad naudotojo paimtas objektas kopijuos rankos vietos ir pasisukimo informaciją.



3.10 pav. „Set Simulate Physics“ ir „Attach Component To Component“ funkcijų mazgai

„Set Simulate Physics“ funkcija priima „Attach Component To Component“ funkcijos išvestį (žr. 3.10 pav.), kuri yra *Boolean* tipo kintamasis nurodantis sėkmingą rankos komponento ir scenos objekto

priskyrimo įvykį. Jei priskyrimas yra sėkmingas, „Set Simulate Physics“ funkcija vykdo scenos objekto fizikos imitaciją.

„IsHandClenched“ funckijai grąžinus rankos atleidimo įvykį yra kviečiama „Detach From Component“ funkcija. Ši funkcija atskiria scenos objektą nuo rankos komponento, jei objektas buvo priskirtas „Attach Component To Component“ funkcijos vykdymo metu.

3.3.3. Judesių registravimas ir projektavimas į dvimatį vaizdą

Unreal Engine žaidimų variklyje buvo sukurtos dvi scenos, kuriose naudotojas gali atlikti simbolių piešimo pratimą. Pratimo atlikimas buvo skaidomas į dvi dalis: pratimo atlikimas naudojant rankų judėjimo sekimo technologijas (*Ultraleap* įmonės *Leap Motion Controller* įrenginys) ir pratimo atlikimas naudojant VR įrenginio valdiklius. Šios dalys buvo įgyvendintos atskirose žaidimų variklio scenose.

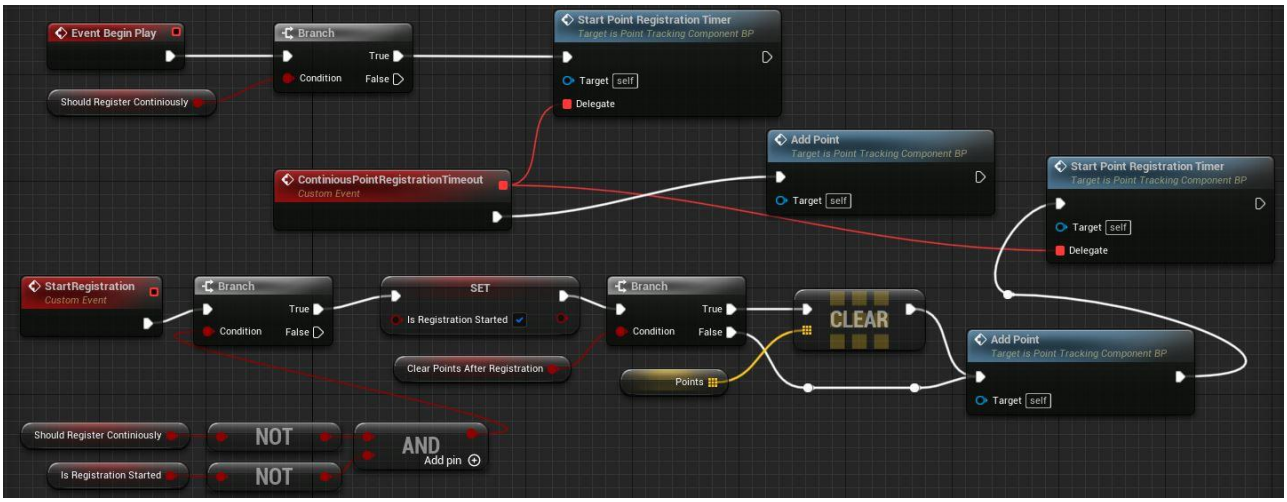
Pawn tipo klasės buvo reikalingos sistemos naudotojo interaktyvumui su aplinka ir pratimo vykdymo logikai aprašyti. Buvo sukurta tėvinė „PointRegPawn_BP“ klasė, kurią paveldėjo kitos *Blueprint* klasės naudojamos skirtingose sistemos scenose. Šią klasę sudaro „Camera“, „LeftMC“, „RightMC“, ir „PointTracker“ komponentai.

„PointTracker“ komponentas buvo naudojamas registruoti taškus virtualioje erdvėje, kai naudotojo rankos juda ir yra įgyvendinamos tam tikros sistemoje aprašytos sąlygos. „PointTracker“ komponentui buvo priskirti šie parametrai:

- „TrackedComponent“, nurodo scenos komponentą, kuris yra sekamas (VR įrenginio valdikliai, *Meta Quest 2* ar *Leap Motion Controller* įrenginių atpažinti sistemos naudotojo rankos kaulai);
- „Points“, vektorių masyvas saugantis užregistruotų taškų informaciją;
- „MaxPoints“, maksimalus skaičius taškų, kuriuos galima registruoti;
- „HandRegistrationTimer“, nurodo laiko momentą, kurio metu yra registruojami taškai;
- „PointRegistrationTimeout“, nurodo laiko momentą, tarp taškų registravimo įvykių;
- „bShouldRegisterContinously“, nurodo sąlygą, kada taškų registravimas yra atliekamas nuolatos;
- „blsRegistrationStarted“, nurodo taškų registravimo sąlygos atitikimo būseną;
- „TrackedSocketName“, nurodo rankos kaulą, kurį reikia sekti, kai naudojamas *Leap Motion Controller* įrenginys pratimo atlikimo metu;
- „bShouldCapPoints“, nurodo registruoti taškų skaičių parinktą „MaxPoints“ parametre;
- „bClearPointsAfterRegistration“, nurodo išvalyti taškų vektoriaus masyvą pasibaigus taškų registravimo įvykiui.

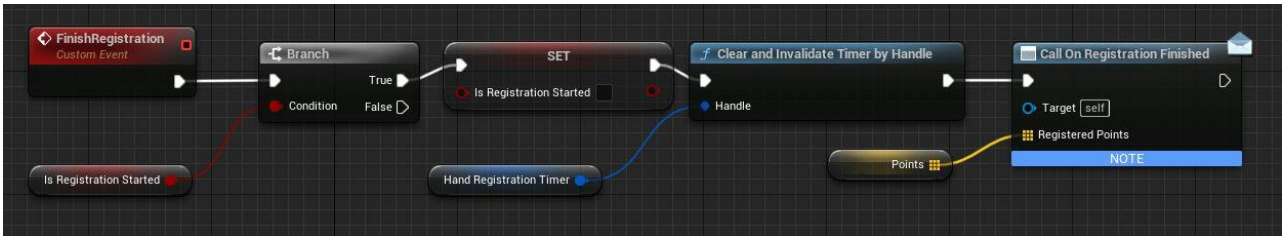
„PointTracker“ komponentui buvo sukurtas įvykių grafikas (žr. 3.11 pav.), kuris nurodo, kad „Begin Play“ įvykis yra įvykdomas, kai sistemos naudotojas pradeda simbolių piešimo pratimą. Prasidėjus pratimui yra kviečiama „Start Point Registration Timer“ funkcija, kuri nurodo laiko momentą, kada yra vykdymas taško registravimo įvykis. Ši funkcija turi „looping“ parametrą, todėl yra vykdomas nuolatinis funkcijos kvietimas kiekvieną kartą jai pasibaigus. Jei nėra naudojamas nuolatinis registravimas ir nėra pradėtas registravimo įvykis, tada vietoje „Starp Point Registration Timer“ funkcijos yra kviečiama „Start Registration“ funkcija, kuri pradeda registracijos įvykį. Prasidėjus taško registravimo įvykiui yra kviečiama „Add Point“ funkcija, kuri patikrina ar scenos komponentas „TrackedComponent“ yra aktyvus. Esant aktyviam scenos komponentui yra pridedamas taškas į vektorių masyvą. Šiam taškui yra priskiriama scenos komponento vietos erdvėje informacija taško

registravimo laiko momentu. „Call Point Added“ įvykis yra naudojamas registruoti taškų pridėjimą į vektorių masyvą ir perduoti užregistruotą informaciją kitiems sistemos objektams. Kai vektorių masyvo ilgis pasiekia „MaxPoints“ parametro vertę, taškų informacija yra pakeičiama į naujausiai užregistruotų taškų informaciją.



3.11 pav. „PointTracker“ komponento įvykių grafikas

Taškų pridėjimas į vektorių masyvą yra atliekamas tol, kol nėra tenkinamas „FinishRegistration“ įvykio sąlyga. Šio įvykio mazgas yra pavaizduotas „PointTracker“ komponento įvykių grafike (žr. 3.12 pav.). Sistemos simbolių piešimo pratimas yra atliekamas tam skirtoje virtualios scenos vietoje, kurioje naudotojo rankų komponentai turi judėti. Jei naudotojo rankų komponentai nėra numatytoje pratimo atlikimo vietoje „Begin Play“ įvykis nėra pradedamas, tačiau jei pratimas buvo atliekamas ir naudotojo rankų komponentai yra patraukiami iš pratimo atlikimo vietos, tada yra tenkinama „FinishRegistration“ įvykio sąlyga. Šio įvykio metu yra pažymimas taškų registravimo pabaigimas ir kviečiama „Call On Registration Finished“ funkcija, kuri perduoda užregistruotus taškus kitiems sistemos objektams.

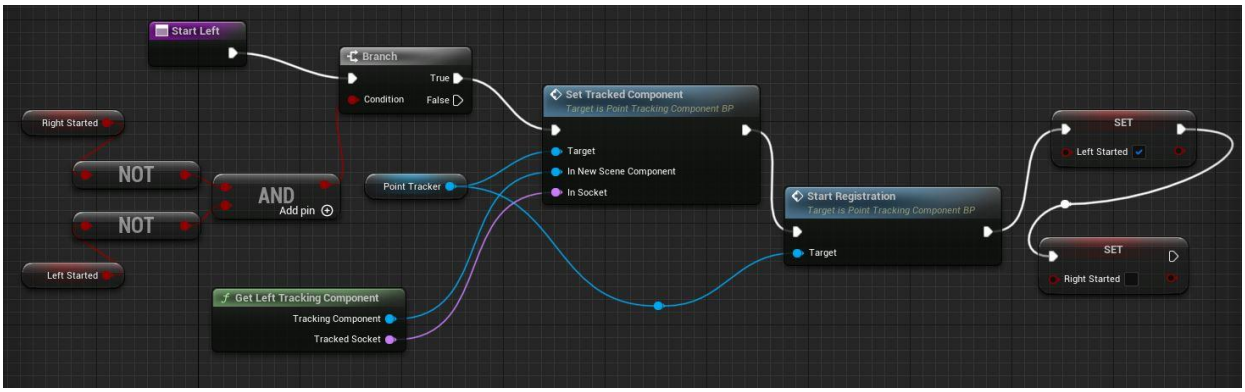


3.12 pav. „FinishRegistration“ įvykio grafikas

„PointRegPawn_BP“ klasės įvykių grafike yra „BeginPlay“ įvykio mazgas, kuris naudoja funkciją „Set Tracking Origin“. Ši funkcija nurodo virtualios realybės įrenginiui, kad sistemos naudotojo realios aplinkos plokštuma atitinka virtualios aplinkos plokštumą. „Set Tracking Origin“ funkcijos panaudojimas leidžia sistemos naudotojui stebėti virtualios aplinkos vaizdą iš tokio aukščio, kuris jam būtų natūralus realioje aplinkoje.

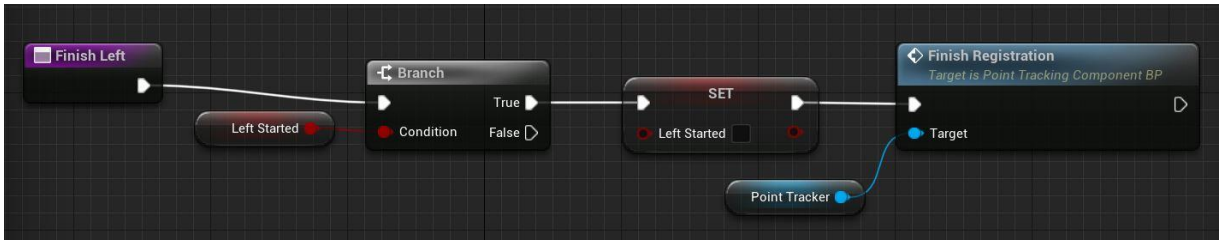
„PointRegPawn_BP“ klasėje yra aprašomi „RightStarted“ ir „LeftStarted“ kintamieji, kurie nusako, kuri sistemos naudotojo ranka yra naudojama simbolių piešimo pratimo atlikimo metu. Šie kintamieji yra naudojami „PointRegPawn_BP“ klasės funckijose. „Start Left“ (žr. 3.13 pav.) ir „Start Right“

funkcijos perduoda kairės ir dešinės rankos komponentus taškų registravimo pradėjimo funkcijai „Start Registration“, kuri yra naudojama „PointTracker“ komponento įvykių grafike (žr. 3.11 pav.). Funkcijai vienu metu gali būti perduodamas tik vienos rankos komponentas. 3.13 pav. yra pavaizduotas kairės rankos komponento perdavimas taškų registravimo pradėjimui.



3.13 pav. „PointRegPawn_BP“ klasės „Start Left“ funkcija

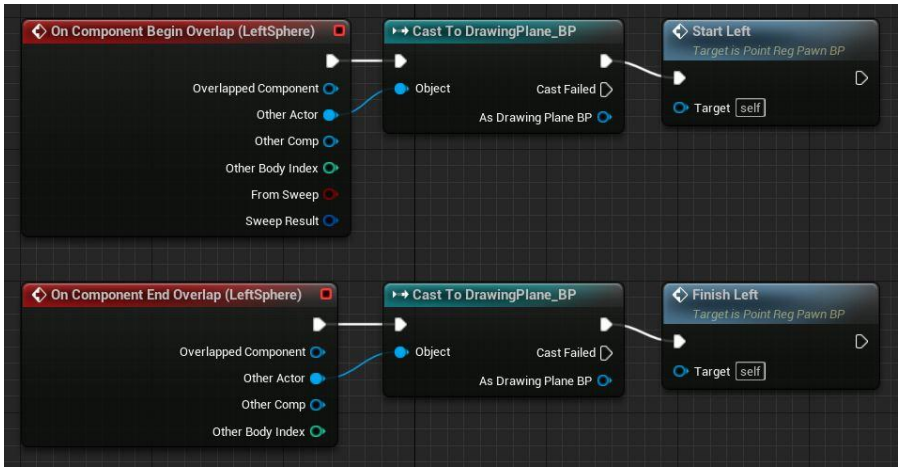
„PointRegPawn_BP“ klasės „Finish Left“ ir „Finish Right“ funkcijos yra naudojamos nurodyti, sistemos naudotojo rankos komponento nebenaudojimą taškų registravimui ir taškų registravimo įvykio pabaigimą. 3.14 pav. yra pavaizduota „Finish Left“ funkcija, kuri patikrina ar kairės rankos komponentas buvo naudojamas taškų registravimui ir nurodo, kad „PointTracker“ komponento „Finish Registration“ funkcijai turi būti vykdomas kairės rankos komponento taškų registravimo pabaigimas.



3.14 pav. „PointRegPawn_BP“ klasės „Finish Left“ funkcija

„MotionControllerPawn_BP“ *Blueprint* klasė paveldi „PointRegPawn_BP“ tėvinės klasės struktūrą ir yra naudojama, kai sistemos naudotojo rankų sekimas yra atliekamas su VR įrenginio valdikliais. Ši klasė iš „PointRegPawn_BP“ tėvinės klasės paveldi „Camera“, „LeftMC“, „RightMC“, ir „PointTracker“ komponentus. Klasei buvo pridėti du komponentai, kurie nurodo, kad rankų sekimui naudojant VR įrenginio valdiklius, sistemos naudotojo rankų komponentams atvaizduoti virtualioje aplinkoje bus naudojami 3D sferų objektai.

„MotionControllerPawn_BP“ klasės įvykių grafike yra aprašomas „PointRegPawn_BP“ klasės „Start Left“, „Start Right“, „Finish Left“ ir „Finish Right“ funkcijų kvietimas kai yra naudojami VR įrenginio valdikliai bei yra tenkinamos šių funkcijų sąlygos. Įvykių grafike taip pat yra aprašomi kairės ir dešinės rankos komponentų persidengimo su scenos objektu įvykiai (žr. 3.15 pav.). „On Component Begin Overlap“ įvykio metu yra patikrinama ar rankos komponentas persidengė su specialiu scenos objektu. Jei ši sąlyga yra tenkinama yra pradedamas rankos komponento taškų registravimas. „On Component End Overlap“ įvykio metu, kai rankos komponentas nustojo persidengti su specialiu scenos objektu, bus sustabdomas rankos komponento taškų registravimas.



3.15 pav. „MotionControllerPawn_BP“ klasės įvykių grafikas

„HandPointRegPawn_BP“ *Blueprint* klasė paveldi „PointRegPawn_BP“ tėvinės klasės struktūrą ir yra naudojama, kai sistemos naudotojo rankų sekimas yra atliekamas su *Meta Quest 2* įrenginio rankų sekimo technologija. Šios klasės įvykių grafike yra aprašomas 3D sferos objekto priskyrimas prie įrenginio atpažintos kairės ir dešinės naudotojo rankos smiliaus piršto galo. Šie sferų objektai atlieka panašią funkciją kaip „MotionControllerPawn_BP“ klasės 3D sferos objektai, kurie yra naudojami sistemos naudotojo rankoms atvaizduoti. 3D sferos ant rankų pirštų galų yra naudojamos patikrinti persidengimo su specialiu scenos objektu įvykio salygas ir pagal tai registruoti arba nustoti registruoti taškus, bei saugoti juos į vektorių masyvą.

„LeapPointRegPawn_BP“ *Blueprint* klasė paveldi „PointRegPawn_BP“ tėvinės klasės struktūrą ir yra naudojama, kai sistemos naudotojo rankų sekimas yra atliekamas su *Leap Motion Controller* įrenginio rankų sekimo technologija. Šioje klasėje yra naudojamas „Leap“ komponentas kuris apskaičiuojama įrenginio sekamų rankų kaulų judėjimo erdvėje informaciją. „LeapPointRegPawn_BP“ klasė taip turi „ChildActor“ komponentą su „LeapLowPolyHand“ reikšme, kuris 3D rankų modelius taiko sistemos naudotojo rankų atvaizdavimui. Panašiai kaip „HandPointRegPawn_BP“ klasėje, „LeapPointRegPawn_BP“ klasėje yra naudojami 3D sferų modeliai, kurie yra priskiriami prie 3D rankų smiliaus pirštų galų ir jų vietos informacija yra naudojama persidengimo su specialiu scenos objektu įvykio sąlygai tikrinti.

3.16 pav. Scenos objektas, kuris naudoja „DrawingPlane_BP“ klasę

„DrawingPlane_BP“ *Blueprint* klasę sudaro kūbo formos kolizijos komponentas, kuris stebi įvykį, kai naudotojo rankų komponentai persidengia su šios klasės specialiuoju scenos objektu. 3D Scenos specialusis objektas, kuris naudoja „DrawingPlane_BP“ klasę yra atvira kūbo formos trimatė dėžė pavaizduota 3.16 paveiksle. Naudotojo rankos komponentui patekus į dėžės objekto vidų yra vykdomi šiame skyrelyje aprašyti su specialuoju objektu susiją įvykiai.

„PointProcessor_BP“ *Blueprint* klasė turi „ProjectDirection“ rodyklės tipo komponentą, kuris yra naudojamas nustatyti užregistruotų taškų projektavimo erdvėje vietą. Šiai klasei buvo sukurti keletas parametrų:

- „PointRegPawn“, nurodo naudotojo rankų komponentą;
- „PlaneVis“, nurodo plokštumos tekstūros paveikslą, ant kurio yra projektuojami taškai;
- „AreaHalfSize“, nurodo plokštumos kraštinės ilgio padalinto iš dviejų reikšmę;
- „ProjectionTransform“, nurodo projekcijos transformacijos specifiškumą;
- „bShouldDrawOnFinish“, nurodo taškų projektavimą jų registravimo pabaigoje arba projektavimą, kai vykdomas nuolatinis taškų registravimas.

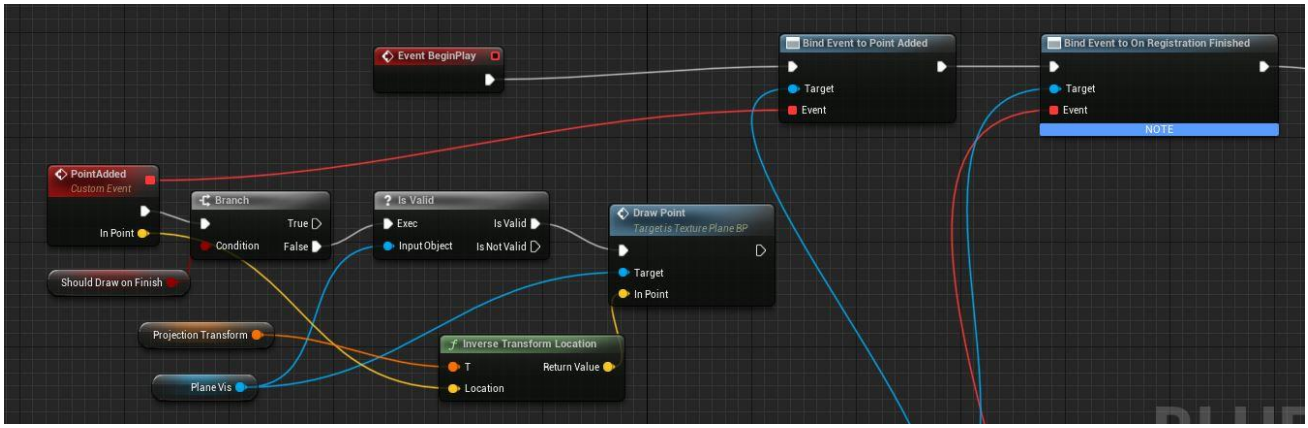
„PointProcessor_BP“ *Blueprint* klasės įvykių grafike „Begin Play“ įvykio metu yra nustatomas rankų sąveikos ir „Point Tracker“ komponentas. „Point Tracker“ komponentui yra priskiriami klausytojai (angl. *Event listener*), kurie klausysis taško pridėjimo į vektorių masyvą ir taškų registravimo pabaigos įvykių. Taškų registravimo pabaigimui įvykus yra atliekama projekcijos transformacija.

Projekcijos transformacija yra apskaičiuojama vykdant „Get Projection Transform“ funkciją (žr. 3.17 pav.), kurioje yra naudojamos projekcijos krypties ir „AreaHalfSize“ parametrų reikšmės. Projekcijos transformacijos metu sistemos naudotojo rankų vietos informacija yra projektuojama dvimatės plokštumos tekstūros paveikslo erdvėje. „Get Projection Transform“ funkcijos vykdymo metu skaičiavimams yra panaudojamos rankų komponento krypties x, y ir z koordinačių ašimis vektorių reikšmės. Funkcija atlieka ašių transformacijas ir pritaiko jas tekstūros paveikslo plotui: x koordinačių ašis yra paverčiama tektūros paveikslo z koordinačių ašimi (ši ašis nėra naudojama taškų projekcijai ant plokštumos), y koordinačių ašis yra paverčiama tektūros paveikslo x koordinačių ašimi, z koordinačių ašis yra paverčiama tektūros paveikslo apversta y koordinačių ašimi. Naujos tektūros paveikslo koordinačių ašys yra paslenkamos taip, kad jos prasidėtų tekstūros paveikslo viršutiniame kairiajame krašte. „PointProcessor_BP“ klasės įvykių grafike yra grąžinama „Get Projection Transform“ funkcijos išvestis, kuri yra išsaugoma, kaip nekintantis klasės kintamasis.

```
FTransform UShapeDetectionFunctionLibrary::GetProjectionTransform(const UObject* WorldContextObject, const FTransform& OriginTransform, float HalfSize)
{
    FVector Fwd = OriginTransform.GetRotation().GetForwardVector();
    FVector Right = OriginTransform.GetRotation().GetRightVector();
    FVector Up = OriginTransform.GetRotation().GetUpVector();

    return FTransform(Right * HalfSize * 2,
        -Up * HalfSize * 2,
        Fwd * HalfSize * 2,
        OriginTransform.GetLocation() + Up * HalfSize - Right * HalfSize);
}
```

3.17 pav. „Get Projection Transform“ funkcija

„TexturePlane_BP“ klasės „Draw Point“ funkcija yra kviečiama, kai yra vykdomas taško pridėjimo į vektorių masyvą įvykis (žr. 3.18 pav.). Užregistruotui taškui yra taikoma „Inverse Transform Location“ funkcija, kuri taškui atlieka apvertimo transformaciją ir naudotojo rankos pasaulio erdvėje informaciją paverčia į lokalios erdvės informaciją. „Draw Point“ funkcija taškus projektuoja į juodos spalvos tekstūros paveikslą ir priskiria jiems baltą spalvą.



3.18 pav. „PointProcessor_BP“ klasės įvykių grafikas

„TexturePlane_BP“ *Blueprint* klasei buvo sukurtas plokštumos, į kurios medžiagos tekstūros paveikslą yra projektuojami užregistruoti taškai, komponentas ir „RenderTargerSender“ komponentas, kuris gali siųsti tekstūros paveikslo informaciją. „TexturePlane_BP“ klasės įvykių grafike yra sukuriamas 64×64 pikselių raiškos ir RGBA spalvų modelio formato tekstūros paveikslas. Įvykių grafike plokštumai buvo parinkta dinamiška (angl. *Dynamic*) medžiaga, kurios atvaizdavimas gali būti keičiamas pratimo atlikimo metu.

3.4. Duomenų rinkinys

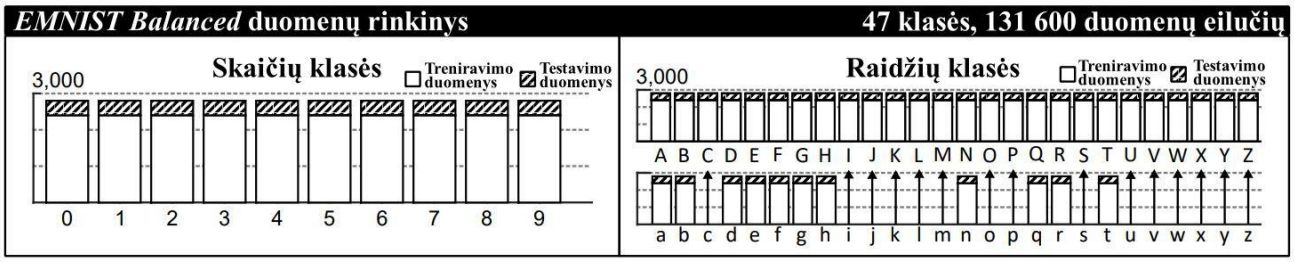
VRR sistemoje naudojamo, mašininio mokymosi modelio treniravimui buvo panaudotas *EMNIST Balanced* duomenų rinkinys [80] ir jį papildančios trys klasės. *EMNIST* duomenų rinkinys yra sukurtas iš *NIST Special Database 19* duomenų rinkinio [81], kurį sudaro 3600 rašytojų 810 000 ranka parašytų simbolių nuotraukos. Šių nuotraukų raiška buvo pakeista į 28×28 pikselių raišką ir jų formatas pateiktas *MNIST* duomenų rinkinio [82] formatu. *EMNIST* duomenų rinkinį sudaro 6 atskiri duomenų rinkiniai: *By_Class, By_Merge, Balanced, Digits, Letters* ir *MNIST*. Šių rinkinių struktūra yra pavaizduota 3.1 lentelėje. VRR sistemos naudojamo modelio treniravimui buvo pasirinktas *Balanced* duomenų rinkinys, kurį sudaro 47 klasės ir jame yra 131 600 duomenų eilučių. *EMNIST* duomenų rinkinio autoriai aprašo *Balanced* rinkinį, kaip labiausiai pritaikomą įvairioms klasifikavimo problemoms spręsti, nes šis rinkinys yra subalansuotas taip, kad visoms klasėms tektų vienodas kiekis treniravimo ir testavimo duomenų eilučių.

3.1 lentelė. *EMNIST* rinkinį išskaidančių duomenų rinkinių struktūra [R15]

Duomenų rinkinio pavadinimas	**Klasių skaičius**	**Treniravimo duomenų kiekis**	**Testavimo duomenų kiekis**	**Ar turi validacijai skirtus duomenis?**	**Bendras duomenų kiekis**
EMNIST By_Class	62	697 932	116 323	Ne	814 255
EMNIST By_Merge	47	697 932	116 323	Ne	814 255
EMNIST Balanced	47	112 800	18 800	Taip	131 600
EMNIST Digits	10	240 000	40 000	Taip	280 000
EMNIST Letters	37	88 800	14 800	Taip	103 600
EMNIST MNIST	10	60 000	10 000	Taip	70 000

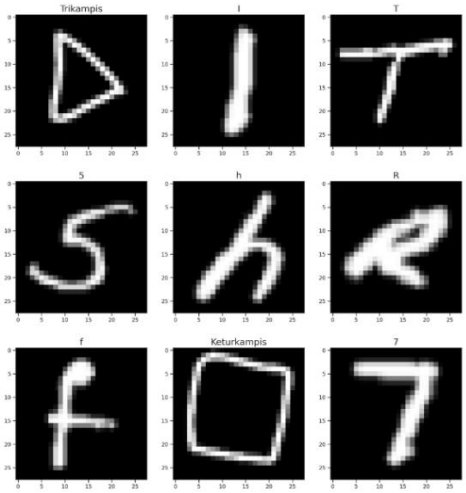
EMNIST Balanced duomenų rinkinio klasių pavadinimus ir jų duomenų eilučių pasiskirstymą galima matyti 3.19 paveiksle. Šis paveikslas parodo, kad duomenų kiekis yra vienodai paskirstytas kiekvienai

klasei ir yra padalintas į treniravimo bei testavimo duomenų rinkinius. 3.19 paveiksle pavaizduotame raidžių klasių pasiskirstyme rodyklė nurodo mažųjų ir didžiųjų raidžių klasių apjungimą į vieną bendrą klasę. Atviros prieigos *EMNIST Balanced* duomenų rinkinys buvo atsisiųstas kaip CSV formato failas [R15], kuriame kiekvieną duomenų eilutę sudaro 785 reikšmės (viena klasės reikšmė ir 784 pikselių reikšmės).



3.19 pav. *EMNIST Balanced* rinkinio klasių pavadinimai ir duomenų pasiskirstymas [R16]

Šiam duomenų rinkiniui buvo sukurtos trys naujos klasės pasinaudojus VRR programoje išpildytu simbolių piešimo mechanizmu ir *Meta Quest 2* įrenginiu. „Rankiniu būdu“ iš viso buvo sukurta 7200 pilkos skalės nuotraukų su trikampio, spiralės ir keturkampio formos simboliais. Nuotraukos buvo konvertuotos į *EMNIST Balanced* duomenų eilučių formatą ir pridėtos į rinkinio CSV formato failą. Papildytą *EMNIST Balanced* rinkinį sudaro 50 klasių 140 000 duomenų, iš kurių 100 000 buvo priskirta treniravimo duomenims, 20 000 – validacijos duomenims ir 20 000 – testavimo duomenims. Dalis rinkinį sudarančių vaizdų yra pavaizduoti 3.20 paveiksle.



3.20 pav. Papildyto *EMNIST Balanced* rinkinio duomenų pavyzdžiai

Duomenų paruošimui naudoti su neuroniniais konvoliuciniais tinklais buvo panaudotos *Pandas, Numpy*, *Keras* ir *Sklearn* bibliotekų funkcijos. Duomenys buvo nuskaityti iš CSV formato failo, normalizuoti ir konvertuoti į *Numpy* bibliotekos 28×28×1 formato duomenų tipą *ndarray*.

3.5. Konvoliucinių neuroninių tinklų architektūrų kūrimas

Mašininio mokymosi modelių kūrimui buvo pasinaudota *Jupyter Notebook* ir *Google Colab* programomis, kurios leidžia konstruoti neuroninius tinklus, juos apmokyti ir testuoti *Python* programavimo kalbos pagalba. *Keras* biblioteka leido kurti konvoliucinius neuroninius tinklus ir aprašyti jų sluoksniams taikomus parametrus. Buvo sukurti trys konvoliuciniai neuroniniai tinklai

priklausantys *Keras* bibliotekos *Sequential* klasei. Ši klasė nurodo jog konvoliucinis neuroninis tinklas priimtus įvesties duomenis perduos nuoseklia tvarka per visus modelio sluoksnius. Tai reiškia, kad įvesties duomenys keliaus nuo modelio architektūroje aprašyto viršutinio (pirmo) sluoksnio iki apatinio (paskutinio) sluoksnio. Modeliams buvo skirti pavadinimai „model_1“, „model_2“, „model_3“.

Pirmam CNN modeliui „model_1“ buvo sukurta architekūra, kurią sudaro keturi konvoliuciniai sluoksniai *Conv2D*, keturi maksimizavimo apjungimo sluoknsiai *MaxPooling2D*, keturi duomenų grupės neuroniniame tinkle verčių normalizavimo sluoksniai *BatchNormalization*, vienas duomenų formos pakeitimo į vienmatę sluoksnis *Flatten*, vienas atsitiktinių reikšmių pakeitimo į nulį sluoksnis *Dropout* ir du visiškai sujungti sluoksniai *Dense*. Paskutiniam architektūros sluoksniui yra taikoma *Softmax* aktyvinimo funkcija. Šis sluoksnis išveda 50 tikimybių reikšmių nuo 0 iki 1, kurios nurodo įvesties duomenų priklausymą kiekvienai iš 50 galimų klasių. Šis neuroninis tinklas turi 333 362 parametrų, iš kurių 332 658 yra naudojami modelio mokymui. Sluoksniams taikomų filtrų skaičių, jų dydį, sluoksnių išvesties formą ir parametrų skaičių galima matyti 3.2 lentelėje.

3.2 lentelė. Pirmo konvoliucinio neuroninio tinklo modelio architektūros struktūra

Sluoksnio pavadinimas	**Filtrų / neuronų skaičius**	**Filtro dydis**	**Išvesties forma**	**Parametrų skaičius**
Conv2D	32	3x3	28×28×32	320
MaxPooling2D	–	2x2	14×14×32	0
BatchNormalization	–	–	14×14×32	128
Conv2D	64	3x3	14×14×64	18 496
MaxPooling2D	–	2x2	7×7×64	0
BatchNormalization	–	–	7×7×64	256
Conv2D	128	3x3	7×7×128	73 856
MaxPooling2D	–	2x2	3×3×128	0
BatchNormalization	–	–	3×3×128	512
Conv2D	128	3x3	3×3×128	147 584
MaxPooling2D	–	2x2	1×1×128	0
BatchNormalization	–	–	1×1×128	512
Flatten	–	–	128	0
Dropout	50 % (išmetimo procentas)	–	128	0
Dense	512	–	512	66 048
Dense	50	–	50	25 650

Antram CNN modeliui „model_2“ buvo sukurta architekūra, kurią sudaro keturi konvoliuciniai sluoksniai *Conv2D*, du maksimizavimo apjungimo sluoknsiai *MaxPooling2D*, du duomenų grupės neuroniniame tinkle verčių normalizavimo sluoksniai *BatchNormalization*, vienas duomenų formos pakeitimo į vienmatę sluoksnis *Flatten*, vienas atsitiktinių reikšmių pakeitimo į nulį sluoksnis *Dropout* ir du visiškai sujungti sluoksniai *Dense*. Paskutiniam architektūros sluoksniui yra taikoma *Softmax* aktyvinimo funkcija. Šis neuroninis tinklas turi 898 194 parametrų, iš kurių 989 002 yra naudojami modelio mokymui. Sluoksniams taikomų filtrų skaičių, jų dydį, sluoksnių išvesties formą ir parametrų skaičių galima matyti 3.3 lentelėje.

3.3 lentelė. Antro konvoliucinio neuroninio tinklo modelio architektūros struktūra

Sluoksnio pavadinimas	**Filtrų / neuronų skaičius**	**Filtro dydis**	**Išvesties forma**	**Parametrų skaičius**
Conv2D	32	5 x 5	28×28×32	832
Conv2D	32	5 x 5	28×28×32	25 632
MaxPooling2D	–	2 x 2	14×14×32	0
BatchNormalization	–	–	14×14×32	128
Conv2D	64	3 x 3	14×14×64	18 496
Conv2D	64	3 x 3	14×14×64	36 928
MaxPooling2D	–	2 x 2	7×7×64	0
BatchNormalization	–	–	7×7×64	256
Dropout	25 % (išmetimo procentas)	–	7×7×64	0
Flatten	–	–	3136	0
Dense	256	–	256	803 072
Dense	50	–	50	12 850

Trečiam CNN modeliui „model_3“ buvo sukurta architekūra, kurią sudaro trys konvoliuciniai sluoksniai *Conv2D*, trys maksimizavimo apjungimo sluoknsiai *MaxPooling2D*, trys duomenų grupės neuroniniame tinkle verčių normalizavimo sluoksniai *BatchNormalization*, vienas duomenų formos pakeitimo į vienmatę sluoksnis *Flatten*, vienas atsitiktinių reikšmių pakeitimo į nulį sluoksnis *Dropout* ir du visiškai sujungti sluoksniai *Dense*. Paskutiniam architektūros sluoksniui yra taikoma *Softmax* aktyvinimo funkcija. Šis neuroninis tinklas turi 511 002 parametrų, iš kurių 510 618 yra naudojami modelio mokymui. Sluoksniams taikomų filtrų skaičių, jų dydį, sluoksnių išvesties formą ir parametrų skaičių galima matyti 3.4 lentelėje.

3.4 lentelė. Trečio konvoliucinio neuroninio tinklo modelio architektūros struktūra

Sluoksnio pavadinimas	**Filtrų / neuronų skaičius**	**Filtro dydis**	**Išvesties forma**	**Parametrų skaičius**
Conv2D	64	3 x 3	28×28×64	640
MaxPooling2D	–	2 x 2	14×14×64	0
BatchNormalization	–	–	14×14×64	256
Conv2D	64	5 x 5	14×14×64	102 464
MaxPooling2D	–	2 x 2	7×7×64	0
BatchNormalization	–	–	7×7×64	256
Conv2D	64	9 x 9	7×7×64	331 840
MaxPooling2D	–	2 x 2	3×3×64	0
BatchNormalization	–	–	3×3×64	256
Dropout	25 % (išmetimo procentas)	–	3×3×64	0
Flatten	–	–	576	0
Dense	120	–	120	69 240
Dense	50	–	50	6050

3.6. Modelių parametrų derinimas

Apmokytų modelių geriausiui rezultatui gauti buvo derinami jų hiperparametrai, kurie kontroliuoja modelių mokymo (treniravimo) procesą. Modeliai buvo treniruojami taikant skirtingus hiperparametrus: partijos dydį (angl. *Batch size*), aktyvinimo funkciją, optimizatorių (angl. *Optimizer*) ir duomenų pakeitimo metodą. Kitų hiperparametrų, tokių kaip treniravimo epochų ir mokymosi greičio (angl. *Learning rate*) reikšmės buvo keičiamos taikant *Keras* bibliotekos grįžtamųjų ryšių (angl. *Callbacks*) objektus. Taikant *Keras* bibliotekos funkciją „Evaluate“ apmokytiems modeliams ir validacijos duomenų rinkiniui, buvo gautas modelių įvertinimas pagal antrame skyriuje aprašytas metrikas. Modelių tikslumo metrika buvo pasirinkta, kaip pati svarbiausia ir pagrindinė vertinimo metrika, nes modeliai buvo treniruojami su subalansuotu duomenų rinkiniu.

Ankstyvo stabdymo (angl. *Early stopping*) objektas leidžia nurodyti didelį modelio treniravimo epochų skaičių treniravimo pradžioje. Šis objektas sustabdo treniravimo procesą epochoje, kurioje modelio tikslumas nustoja didėti. Epocha yra vienas modelio mokymosi ciklas su visu mokymosi duomenų rinkiniu. Treniravimo pradžioje visiems modeliams buvo skirta 100 epochų, tačiau ankstyvo stabdymo objektas nutraukdavo treniravimo procesą vidutiniškai ties 20 epocha. Ankstyvo stabdymo objektui buvo taikomi šie parametrai [83]:

- „min_delta = 0“, nurodo mokymosi nutraukimą epochoje, kurioje modelio tikslumo pokyčio reiškmė yra lygi 0;
- „patience = 0“, nurodo epochų, po kurių mokymasis bus nutrauktas, skaičių, jei tikslumo pokytis „min_delta“ bus pasiektas;
- „restore_best_weights = True“, nurodo atsatyti modelio gautus svorius epochoje, kurioje buvo pasiektas geriausias tikslumas;
- „monitor = val_accuracy“, nurodo modelio tikslumo stebėjimą, kai jis yra taikomas validacijos duomenų rinkiniui.

Keras bibliotekos grįžtamųjų ryšių *ReduceLROnPlateau* objektas buvo naudojamas kontroliuoti modelių mokymosi greitį. Per mažas modelio mokymosi greitis gali lemti ilgą mokymosi procesą, kuris gali užstrikti ir neduoti gerų rezultatų. Per didelis mokymosi greitis gali nulemti neoptimalų modelio svorių rinkinį ir nestabilų mokymosi procesą [84]. *ReduceLROnPlateau* objektui buvo taikomi šie parametrai [85]:

- „monitor = val_loss“, nurodo modelio našumo (angl. *Performance*) stebėjimą, kai jis yra taikomas validacijos duomenų rinkiniui;
- „patience = 3“, nurodo epochų, po kurių mokymosi greitis bus mažinamas, skaičių;
- „factor = 0,2“, nurodo mokymosi greičio daugiklio reikšmę;
- „min_lr = 0,0001“, nurodo mažiausią, galimą pasiekti mokymosi greičio reikšmę.

Grįžtamųjų ryšių *ModelCheckpoint* objektas [86] buvo naudojamas išsaugoti modelio svorių rinkinį skirtinguose treniravimo etapuose. Treniravimo metu pasiekus naują geriausią modelio tikslumą validacijos duomenų rinkiniui, modelio svoriai buvo išsaugomi.

Prieš pradedant modelių parametrų derinimo procesą atsitiktiniu būdu buvo parinkta tinklų konvoliuciniuose sluoksniuose taikoma *LeakyReLU* aktyvinimo funkcija ir *RMSprop* optimizacijos algoritmas.

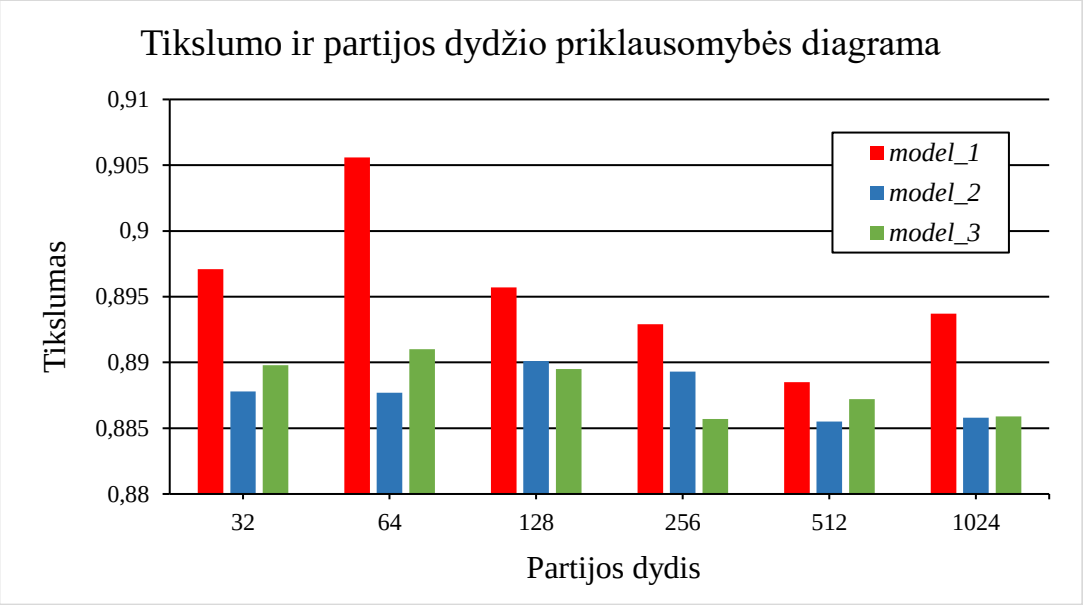
Partijos dydis yra modelių treniravimo hiperparametras, kuris nurodo duomenų rinkinio eilučių (nuotraukų) kiekį vienoje mokymo iteracijoje. Kiekvienos iteracijos pabaigoje modelio svoriai yra atnaujinami. Vieną epochą sudaro bendras iteracijų kiekis, reikalingas mokymosi ciklui su visu treniravimo duomenų rinkiniu. 3.5 lentelėje yra pavaizduotas modelių įvertinimas validacijos duomenų rinkiniui, kai treniravimo metu modeliams buvo taikomi skirtingi partijos dydžiai.

3.5 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingus partijos dydžius

Modelio pavadinimas	Partijos dydis	Praradimas	Tikslumas	Preciziškumas	Atkūrimas
„model_1“	32	0,3104	0,8971	0,9148	0,8789
	64	0,2797	0,9056	0,9201	0,8917
	128	0,2996	0,8957	0,915	0,8801
	256	0,3059	0,8929	0,9126	0,8758
	512	0,3159	0,8885	0,9015	0,8778
	1024	0,2999	0,8937	0,9065	0,883
„model_2“	32	0,3293	0,8878	0,9085	0,8711
	64	0,3315	0,8877	0,9031	0,876
	128	0,3282	0,8901	0,9051	0,8798
	256	0,3235	0,8893	0,904	0,8771
	512	0,3294	0,8855	0,9052	0,8683
	1024	0,3321	0,8858	0,9087	0,8647
„model_3“	32	0,3157	0,8898	0,9061	0,8772
	64	0,3172	0,891	0,9078	0,8748
	128	0,3334	0,8895	0,9021	0,8775
	256	0,3288	0,8857	0,9074	0,8668
	512	0,3256	0,8872	0,904	0,8742
	1024	0,327	0,8859	0,9075	0,8666

3.5 lentelės žalios spalvos laukai nurodo geriausias įvertinimo metrikų reikšmes. Mažiausias praradimo (angl. *Loss*) įvertinimas ir didžiausi tikslumo, preciziškumo ir atkūrimo įvertinimai gauti modeliui „model_1“, kai buvo taikoma partijos dydžio reikšmė lygi 64. Didžiausi tikslumo ir atkūrimo įvertinimai gauti modeliui „model_2“, kai buvo taikoma partijos dydžio reikšmė lygi 128. Didžiausi tikslumo ir preciziškumo įvertinimai gauti modeliui „model_3“, kai buvo taikoma partijos dydžio reikšmė lygi 64. Šios partijos dydžio reikšmės buvo naudojamos likusiame modelių parametrų derinimo procese.

3.21 paveikslas vaizduoja modelių tikslumo ir jų treniravimo metu taikytų partijos dydžių priklausomybės diagramą. Ši diagrama parodo, kad geriausią tikslumą lygų 0,9056 (90,56 %) pasiekė modelis „model_1“. Blogiausias tikslumas, kurio vertė yra 0,8855 (88,55 %), buvo pasiektas modelį „model_2“ treniruojant su partijos dydžiu lygiu 512.



3.21 pav. Modelių tikslumo ir treniravimui naudoto partijos dydžio priklausomybės diagrama

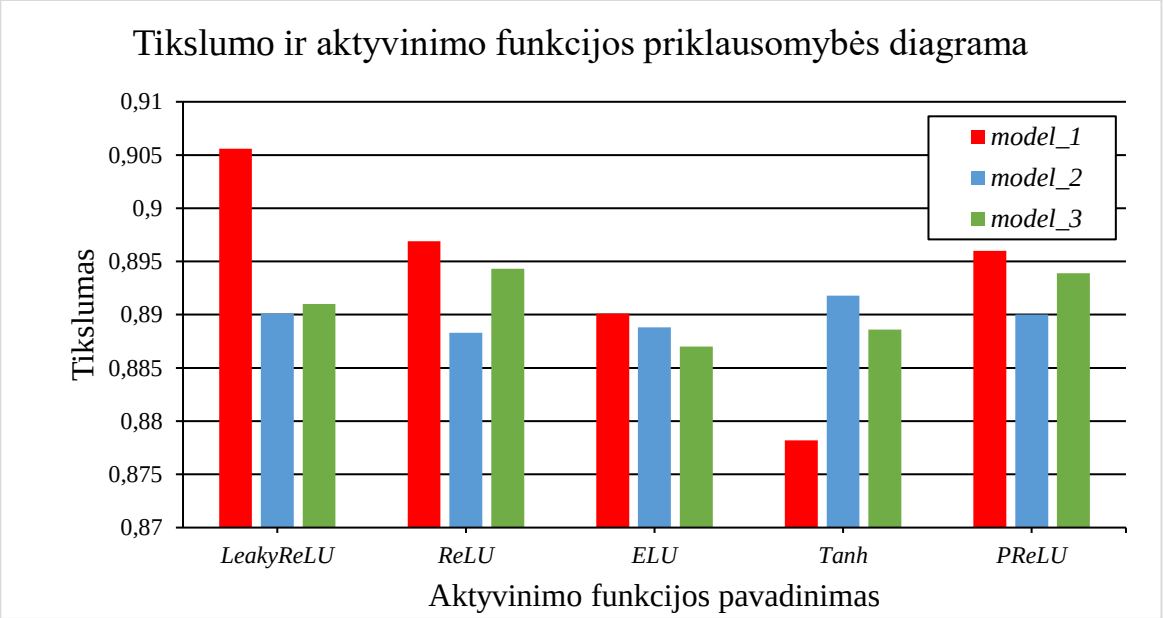
3.6 lentelėje yra pavaizduotas modelių įvertinimas validacijos duomenų rinkiniui, kai modelių konvoliuciniams sluoksniams buvo taikomos *LeakyReLU*, *ReLU*, *ELU*, *Tanh* ir *PReLU* aktyvinimo funkcijos.

3.6 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingas aktyvinimo funkcijas

Modelio pavadinimas	**Aktyvinimo funkcija**	**Praradimas**	**Tikslumas**	**Preciziškumas**	**Atkūrimas**
„model_1“	*LeakyReLU*	0,2797	0,9056	0,9201	0,8917
	ReLU	0,2989	0,8969	0,9136	0,8813
	ELU	0,3108	0,8901	0,9119	0,8685
	Tanh	0,3623	0,8782	0,8909	0,8678
	PReLU	0,2996	0,896	0,9146	0,8792
„model_2“	*LeakyReLU*	0,3282	0,8901	0,9051	0,8798
	ReLU	0,3235	0,8883	0,908	0,8699
	ELU	0,3299	0,8888	0,9058	0,8741
	Tanh	0,3091	0,8918	0,9132	0,8752
	PReLU	0,3156	0,89	0,9067	0,8733
„model_3“	*LeakyReLU*	0,3172	0,891	0,9078	0,8748
	ReLU	0,2992	0,8943	0,9105	0,881
	ELU	0,3321	0,887	0,901	0,8751
	Tanh	0,318	0,8886	0,9058	0,8726
	PReLU	0,3033	0,8939	0,9099	0,8792

3.6 lentelės žalios spalvos laukai nurodo geriausias įvertinimo metrikų reikšmes. Mažiausias praradimo įvertinimas ir didžiausi tikslumo, preciziškumo ir atkūrimo įvertinimai gauti modeliui „model_1“, kai buvo taikoma *LeakyReLU* aktyvinimo funkcija. Mažiausias praradimo įvertinimas ir

didžiausi tikslumo ir preciziškumo įvertinimai gauti modeliui „model_2“, kai buvo taikoma *Tanh* aktyvinimo funkcija. Mažiausias praradimo įvertinimas ir didžiausi tikslumo, preciziškumo ir atkūrimo įvertinimai gauti modeliui „model_3“, kai buvo taikoma *ReLU* aktyvinimo funkcija. Šios aktyvinimo funkcijos buvo naudojamos likusiame modelių parametrų derinimo procese.



3.22 pav. Modelių tikslumo ir skirtingų aktyvinimo funkcijų taikymo priklausomybės diagrama

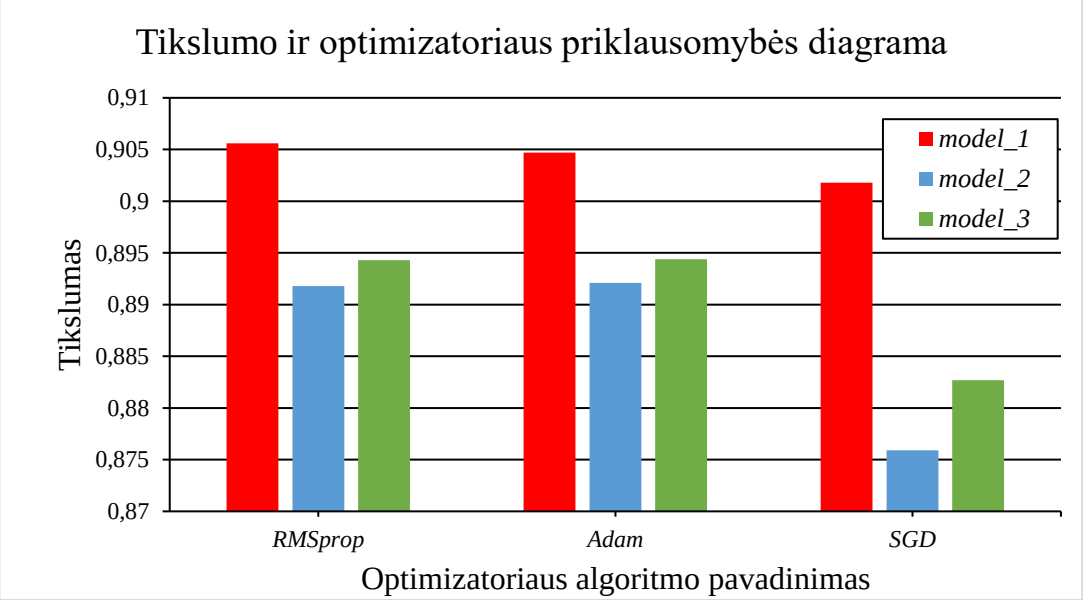
3.22 paveikslas pavaizduoja modelių tikslumo ir jų konvoliuciniams sluoksniams taikytų aktyvinimo funkcijų priklausomybės diagramą. Ši diagrama parodo, kad geriausią tikslumą lygų 0,9056 (90,56 %) pasiekė modelis „model_1“. Blogiausias tikslumas, kurio vertė yra 0,8782 (87,82 %), buvo pasiektas modeliui „model_1“ taikant *Tanh* aktyvinimo funkciją.

3.7 lentelėje yra pavaizduotas modelių įvertinimas validacijos duomenų rinkiniui, kai modelių treniravimo metu buvo taikomi *RMSprop*, *Adam* ir *SGD* optimizatoriai. Optimizatoriai yra algoritmai naudojami sumažinti modelio treniravimo metu gaunamas praradimo reikšmes [87].

3.7 lentelė. Modelių įvertinimo metrikų reikšmės taikant skirtingus optimizatorius

Modelio pavadinimas	**Optimizatorius**	**Praradimas**	**Tikslumas**	**Preciziškumas**	**Atkūrimas**
„model_1“	*RMSprop*	0,2797	0,9056	0,9201	0,8917
	Adam	0,2889	0,9047	0,9168	0,8936
	SGD	0,2792	0,9018	0,9166	0,8875
„model_2“	*RMSprop*	0,3091	0,8918	0,9132	0,8752
	Adam	0,309	0,8921	0,9125	0,8749
	SGD	0,3697	0,8759	0,9112	0,8455
„model_3“	*RMSprop*	0,2992	0,8943	0,9105	0,881
	Adam	0,3005	0,8944	0,9093	0,8813
	SGD	0,3444	0,8827	0,9088	0,8546

3.7 lentelės žalios spalvos laukai nurodo geriausias įvertinimo metrikų reikšmes. Geriausi tikslumo ir preciziškumo įvertinimai modeliui „model_1“ buvo gaunami taikant *RMSprop* optimizatoriaus algortimą. Taikant *Adam* optimizatoriaus algoritmą modeliui „model_2“ buvo gautos geriausios praradimo ir tikslumo vertės. Nors modeliui „model_3“ buvo gautos geriausios praradimo ir preciziškumo reikšmės taikant *RMSprop* optimizatoriaus algoritmą, geriausias modelio tikslumas buvo pasiektas su *Adam* algoritmu. Šie optimizatoriai buvo naudojami likusiame modelių parametrų derinimo procese.



3.23 pav. Modelių tikslumo ir optimizatorių taikymo priklausomybės diagrama

3.23 paveikslas pavaizduoja modelių tikslumo ir jų treniravimo metu taikytų optimizatorių algoritmų priklausomybės diagramą. Ši diagrama parodo, kad geriausią tikslumą lygų 0,9056 (90,56 %) pasiekė modelis „model_1“. Blogiausias tikslumas, kurio vertė yra 0,8759 (87,59 %), buvo pasiektas modeliui „model_2“ taikant *SGD* optimizatoriaus algoritmą.

Keras bibliotekos *ImageDataGenerator* klasės objektai buvo naudojami pagal atitinkamus parametrus pakeisti pradines duomenų rinkinio reikšmes. Šie objektai priima partijos duomenis (nuotraukas) ir pritaiko jiems įvarias transformacijas, tokias kaip pasukimas, pavertimas ar dydžio pakeitimas. Transformacijoms dažniausiai yra būdingas atsitiktinių reikšmių esančių tam tikrame intervale parinkimas. *ImageDataGenerator* klasės objektai pradinius partijos duomenis pakeičia naujais, transformuotais duomenimis, kurie yra naudojami modelio treniravime [88]. Modelio parametrų derinimo procese buvo taikomi du *ImageDataGenerator* klasės objektai.

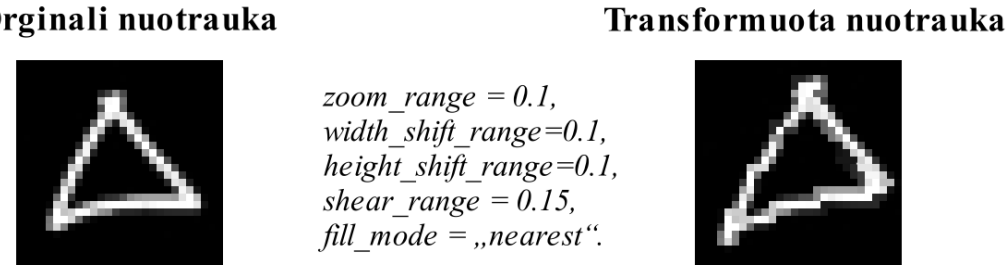
Pirmajam objektui buvo priskirtos šios nuotraukų transformacijos operacijos:

- „zoom_range = 0,1”, nurodo priartinti nuotrauką nekeičiant jos pradinio dydžio pagal atsitiktinai parinktą procento reikšmę, kuri daugiausiai gali priartinti nuotrauką nuo –10 % iki 10 %;
- „width_shift_range = 0,1”, nurodo pastumti nuotraukos pikselius horizontaliai (į kairę arba į dešinę pusę) pagal atsitiktinai parinktą procento reikšmę. Daugiausiai pikselius galima pastumti 10 % į kairę arba į dešinę pusę;

– „height_shift_range = 0,1“, nurodo pastumti nuotraukos pikselius vertikaliai (į viršų arba į apačią) pagal atsitiktinai parinktą procento reikšmę, pagal kurią daugiausiai galima pastumti pikselius 10 % į viršų arba į apačią pusę.

Antrajam objektui buvo priskirtos šios nuotraukų transformacijos operacijos:

– „zoom_range = 0,05“, nurodo priartinti nuotrauką nekeičiant jos pradinio dydžio pagal atsitiktinai parinktą procento reikšmę, kuri daugiausiai gali priartinti nuotrauką nuo –5 % iki 5 %;
– „width_shift_range = 0,1“ (vienoda pirmam objektui);
– „height_shift_range = 0,1“ (vienoda pirmam objektui);
– „shear_range = 0,15“, nurodo šlyties (angl. *Shear*) kampą, pagal atsitiktinai parinktą laipsnį nuo –15º iki 15º. Šis parametras nurodo kad vaizdas bus iškraipomas išilgai x arba y koordinačių ašimis;
– „fill_mode = nearest“, nurodo, kad tušti (neužpildyti) pikseliai po transformacijų taikymo būtų užpildyti artimiausių (kaimynų) pikselių reikšmėmis;
– „rotation_range = 10“, nurodo pasukti nuotrauką atsitiktinai parinktu laipsniu nuo –10º iki 10º.



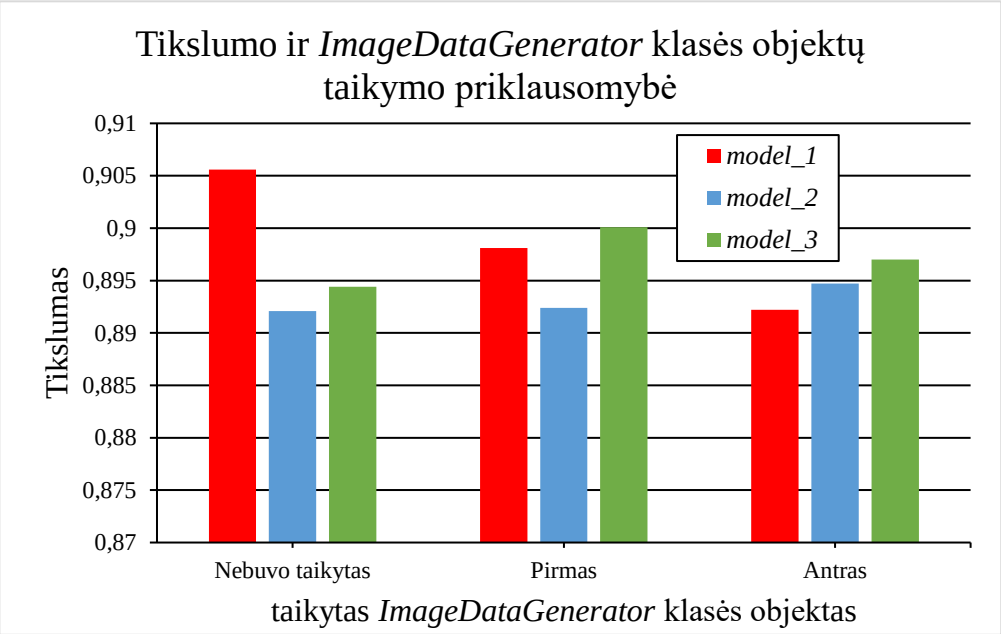
3.24 pav. *ImageDataGenerator* klasės objekto su atitinkamais parametrais taikymo pavyzdys

3.24 paveikslas vaizduoja nuotrauką kuriai buvo taikytos „zoom_range“, „width_shift“, „height_shift“, „shear_range“, „fill_mode“ transformacijos. Transformuota nuotrauka yra 10 % priartinta, 10 % pastumta į dešinę, 10 % pastumta į apačią, iškreipta –15º x ir y koordinačių ašimis. Nuotraukos tuštiems pikseliams užpildyti buvo pritaikytas „nearest“ metodas.

3.8 lentelė. Modelių įvertinimas taikant skirtingus *ImageDataGenerator* klasės objektus

Modelio pavadinimas	***ImageDataGenerator* klasės objektas**	**Praradimas**	**Tikslumas**	**Preciziškumas**	**Atkūrimas**
„model_1“	Nebuvo taikytas	0,2797	0,9056	0,9201	0,8917
	Pirmas	0,292	0,8981	0,9186	0,878
	Antras	0,3165	0,8922	0,914	0,8707
„model_2“	Nebuvo taikytas	0,309	0,8921	0,9125	0,8749
	Pirmas	0,3066	0,8924	0,9156	0,8697
	Antras	0,2948	0,8947	0,917	0,8724
„model_3“	Nebuvo taikytas	0,3005	0,8944	0,9093	0,8813
	Pirmas	0,2775	0,9001	0,9158	0,8852
	Antras	0,2843	0,897	0,9155	0,8817

3.8 lentelėje yra pavaizduotas modelių įvertinimas validacijos duomenų rinkiniui, kai modelių treniravimo metu buvo taikomi *ImageDataGenerator* klasės objektai. Lentelės žalios spalvos laukai nurodo geriausias įvertinimo metrikų reikšmes. Geriausi praradimo, tikslumo, preciziškumo ir atkūrimo įvertinimai modeliui „model_1“ buvo gaunami kai nebuvo taikomi *ImageDataGenerator* klasės objektai. Taikant antrąjį *ImageDataGenerator* klasės objektą modeliui „model_2“ buvo gautos geriausios praradimo, tikslumo ir preciziškumo vertės. Geriausi praradimo, tikslumo, preciziškumo ir atkūrimo įvertinimai modeliui „model_3“ buvo gaunami, kai buvo taikomas pirmasis *ImageDataGenerator* klasės objektas.



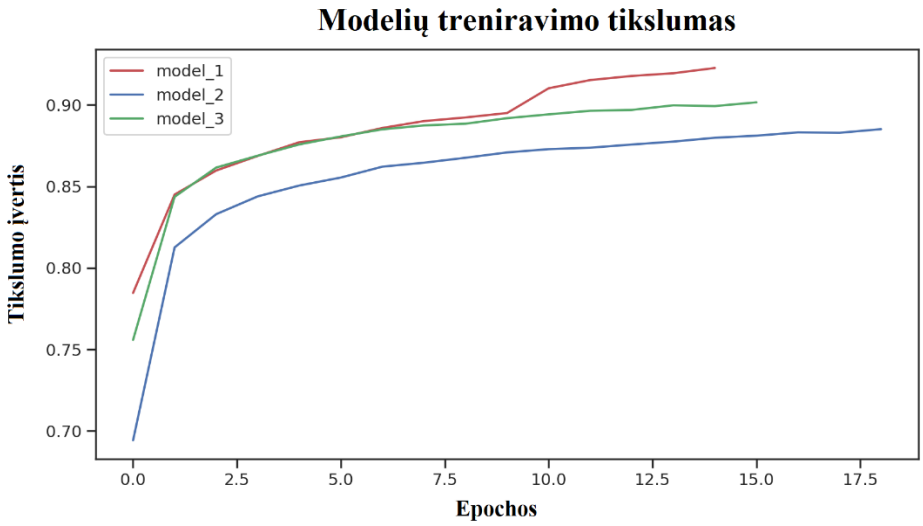
3.25 pav. Modelių tikslumo ir *ImageDataGenerator* klasės objekto priklausomybės diagrama

3.25 paveikslas pavaizduoja modelių tikslumo ir jiems taikytų *ImageDataGenerator* klasės objektų priklausomybės diagramą. Ši diagrama parodo, kad geriausią tikslumą lygų 0,9056 (90,56 %) pasiekė modelis „model_1“. Blogiausias tikslumas, kurio vertė yra 0,8921 (89,21 %), buvo pasiektas modeliui „model_2“ netaikant *ImageDataGenerator* klasės objektų.

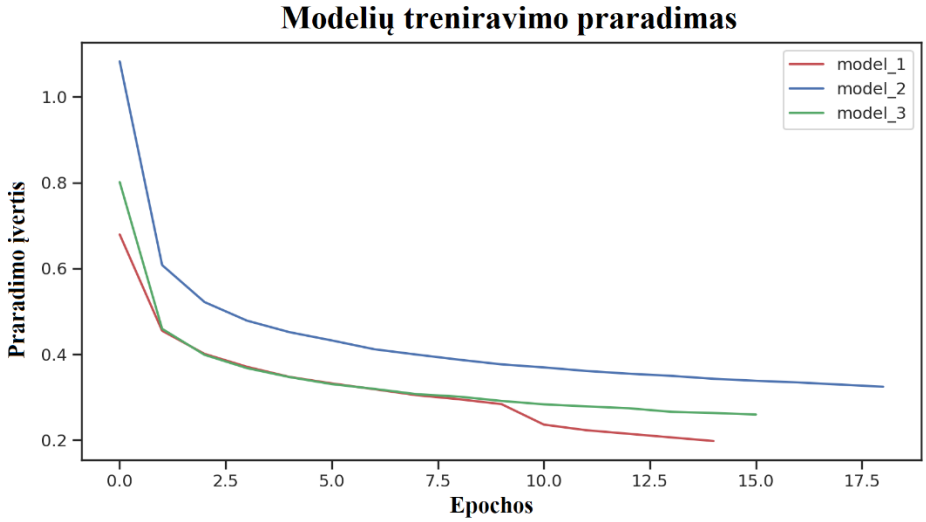
3.9 lentelė pavaizduoja parametrų derinimo procese modeliams taikytus hiperparametrus, su kuriais buvo gauti geriausi modelių rezultatai. Taikant lentelėje nurodytus hiperparametrus buvo sugeneruoti modelių tikslumo ir praradimo treniravimo ir validacijos duomenų rinkiniams reikšmių pasikeitimo kiekvienoje epochoje grafikai. Grafikų generavimui buvo pasinaudota *Matplotlib* bibliotekos *Pyplot* funkcijų rinkiniu. Šie grafikai yra pavaizduoti 3.26, 3.27, 3.28 ir 3.29 paveiksluose.

3.9 lentelė. Modelių parametrų derinimo rezultatai

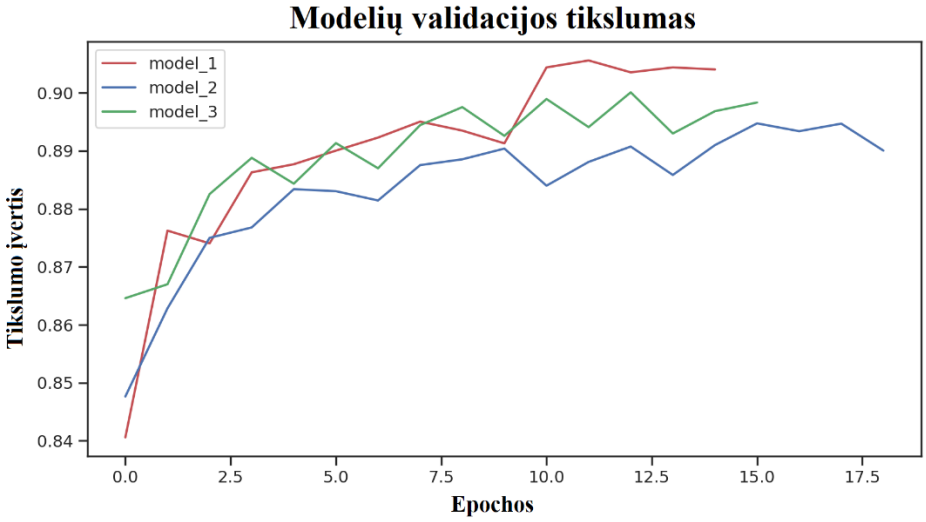
Modelio pavadinimas	**Partijos dydis**	**Aktyvinimo funkcija**	**Optimizatorius**	***ImageDataGenerator* klasės objektas**
„model_1“	64	*LeakyReLU*	*RMSprop*	Nebuvo taikytas
„model_2“	128	*Tanh*	*Adam*	Antras
„model_3“	64	*ReLU*	*Adam*	Pirmas



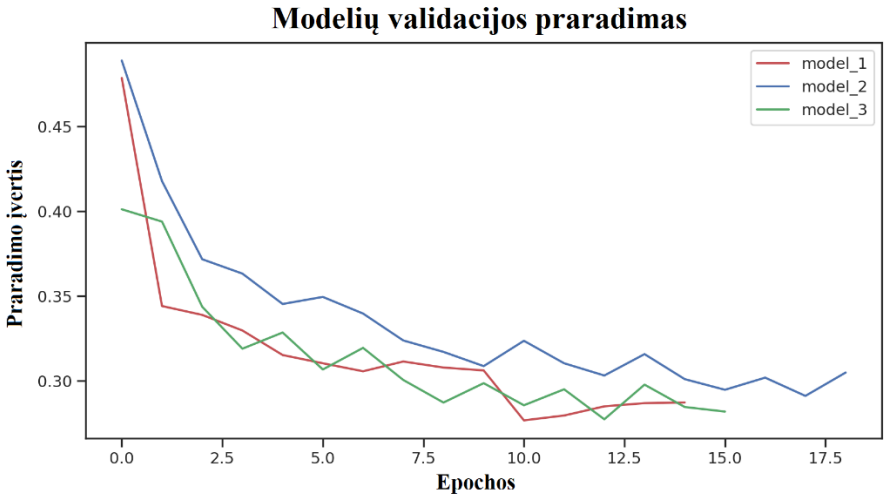
3.26 pav. Modelių treniravimo tikslumo reikšmių pasikeitimo grafikas



3.27 pav. Modelių treniravimo praradimo reikšmių pasikeitimo grafikas



3.28 pav. Modelių validacijos tikslumo reikšmių pasikeitimo grafikas



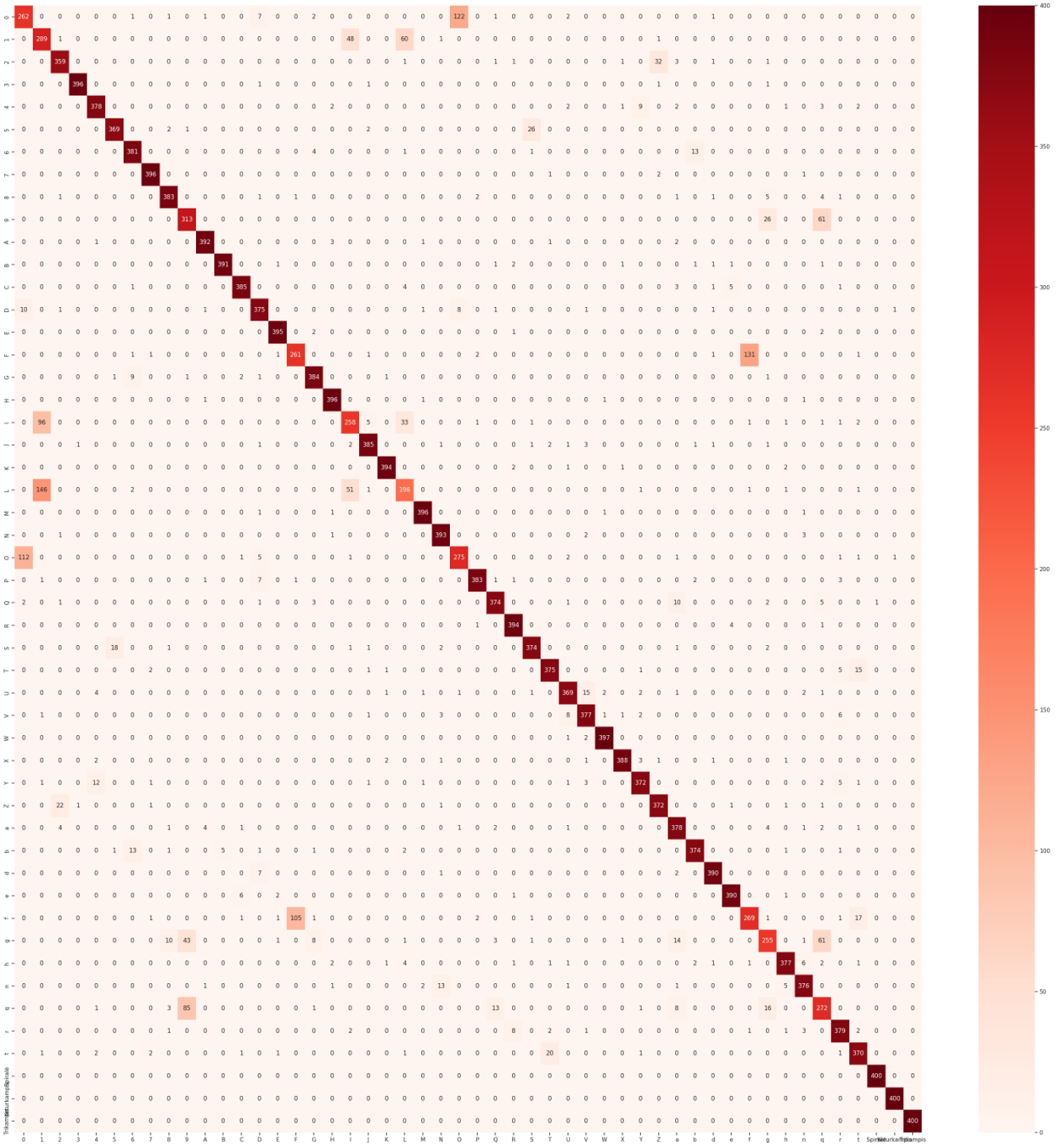
3.29 pav. Modelių validacijos praradimo reikšmių pasikeitimo grafikas

Modelių parametrų derinimo metu buvo pastebėta, kad geriausias modelis iš trijų buvo „model_1“. Šio modelio klasifikavimo našumui atvaizduoti buvo panaudota maišos matrica pavaizduota 3.30 pav. Paveiksle yra matomi modelio „model_1“ klasifikavimo rezultatai su nematytais duomenimis iš testavimo duomenų rinkinio ir orginalios duomenų klasės. Kiekviena klasė minėtame testavimo rinkinyje turi po 400 nuotraukų. Maišos matrica gali padėti pamatyti klases, kurias modelis dažnai atpažįsta neteisingai. Iš maišos matricos galima pastebėti, kad modelis labiausiai klydo šiose situacijose:

- 122 nuotraukas, kuriose buvo pavaizduotas simbolis “O”, atpažino kaip nuotraukas vaizduojančias simbolį “0” (tikslumas “O” klasei yra 68,75 %);
- 131 nuotraukas, kuriose buvo pavaizduotas simbolis “f”, atpažino kaip nuotraukas vaizduojančias simbolį “F” (tikslumas “f” klasei yra 67,25 %);
- 96 nuotraukas, kuriose buvo pavaizduotas simbolis “1”, atpažino kaip nuotraukas vaizduojančias simbolį “I” (tikslumas “1” klasei yra 72,25 %);
- 146 nuotraukas, kuriose buvo pavaizduotas simbolis “1”, atpažino kaip nuotraukas vaizduojančias simbolį “L”;
- 112 nuotraukas, kuriose buvo pavaizduotas simbolis “0”, atpažino kaip nuotraukas vaizduojančias simbolį “O” (tikslumas “0” klasei yra 65,5 %);
- 105 nuotraukas, kuriose buvo pavaizduotas simbolis “F”, atpažino kaip nuotraukas vaizduojančias simbolį “f” (tikslumas “F” klasei yra 65,25 %).

Maišos matrica parodo, kad modeliui *model_1* geriausiai sekėsi atpažinti “Spiralės”, “Trikampio”, “Keturkampio” (šių trijų klasių atpažinimas buvo 100 % teisingas), “M”, “3”, “7”, “H” (šių keturių klasių atpažinimas buvo 99 % teisingas), “W” (šios klasės atpažinimas buvo 99,25 % teisingas), “e” ir “d” (šių klasių atpažinimas buvo 97,5 % teisingas) klases.

Modelį „model_1“ įvertinus su testavimo duomenų rinkiniu buvo gautas jo atpažinimo tikslumas lygus 90,04 %, preciziškumas lygus 91,68 % ir atkūrimas lygus 88,84 %.

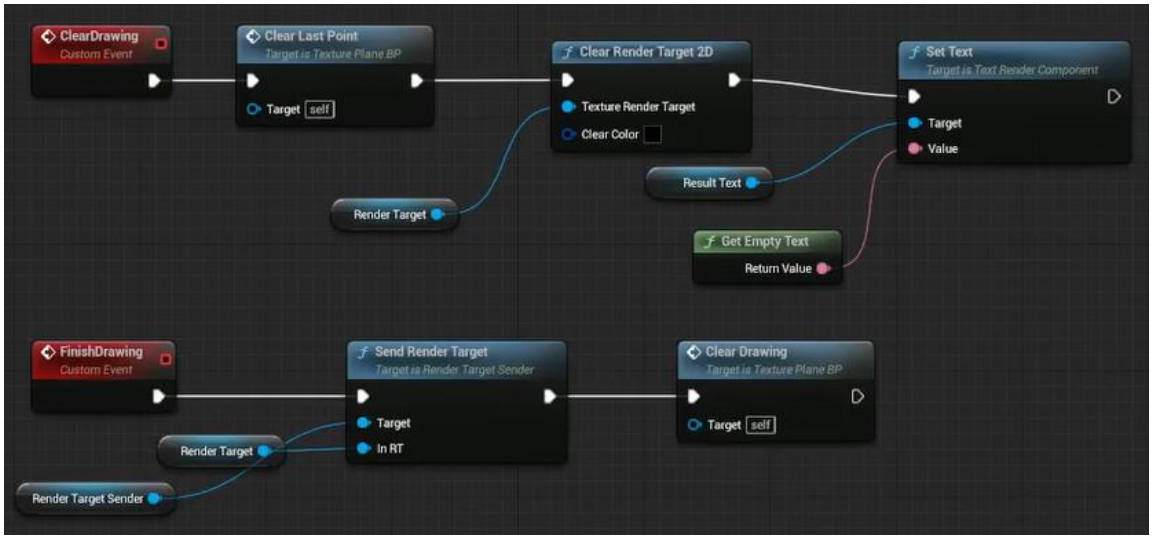


3.30 pav. Modelio „model_1“ klasifikavimo rezultatų maišos matrica

3.7. Duomenų siuntimas

Unreal Engine žaidimų variklio *Blueprint* klasėje „TexturePlane_BP“ buvo sukurtas virtualios realybės scenoje esančių mygtukų funkcionalumas su „Clear Drawing“ ir „Finish Drawing“ įvykiais (žr. 3.31 pav.). „Clear Drawing“ įvykio metu yra patikrinama ar buvo nuspaustas raudonos spalvos mygtukas. Jei ši sąlyga yra tenkinama tekstūros paveikslo informacija yra išvaloma. „Finish Drawing“ įvykio metu yra patikrinama ar buvo nuspaustas žalios spalvos mygtukas. Jei ši sąlyga yra tenkinama tekstūros paveikslo informacija bus išsiunčiama kitiems objektams ir po to išvaloma.

Konvoliucinio neuroninio tinklo modelis gautą informaciją klasifikuoja į tam tikrą klasę ir grąžina atpažintos klasės ir jos tikslumo tikimybės reikšmes, kurios yra pavaizduojamos scenos aplinkoje teksto formatu, kai buvo nuspaustas žalias mygtukas.



3.31 pav. „TexturePlane_BP“ klasės įvykių grafikas

„TexturePlane_BP“ klasės „RenderTargerSender“ komponentas turi IP adreso ir tinklo prievado numerio parametrus. Nors dažniausiai šie parametrai yra naudojami interneto protokolo perdavimo lygyje, tačiau šių parametrų reikšmės buvo naudojamos sukurti jungtį tarp kompiuterio atmintyje išsaugoto PY formato failo ir virtualios realybės sistemos.

Duomenys iš sistemos į failą yra siunčiami TCP protokolu. Kitaip nei UDP protokolas, TCP užtikrina patikimą duomenų siuntimą tarp dviejų tinklo taškų. „RenderTargerSender“ komponentui buvo sukurta privati „TcpSocket“ klasė, kuri nurodo vieną dvipusio ryšio tarp dviejų tinkle veikiančių programų galinį tašką ar lizdą. Lizdas yra susiejamas su tinklo prievado numeriu, kad TCP sluoksnis galėtų identifikuoti siunčiamos informacijos gavėją.

„RenderTargerSender“ komponento „Begin Play“ įvykio metu, kai VRR sistema yra paleidžiama, lizdas yra atidaromas ir sukuriamas 64×64 dydžio paketo buferis, kuris bus naudojamas siųsti tekstūros paveikslo informaciją konvertuotą į baitus. „End Play“ įvykio metu lizdas yra uždaromas ir paketo buferis yra sunaikinamas.

„RenderTargerSender“ komponento „Send Render Target“ funkcija patikrina ar „TcpSocket“ klasės lizdas buvo sukurtas. Jei ši sąlyga yra tenkinama, tada siunčiamo tekstūros paveikslo baitams yra priskiriamas buferis, kuris yra išsiunčiamas į kompiuteryje išsaugotą PY formato failą. Šiame faile CNN modelis yra užkraunamas iš kompiuteryje išsaugoto h5 formato failo. Modeliui yra pateikiama gauta informacija iš buferio ir modelis grąžina aptiktos simbolių klasės ir priklausymo šiai klasei tikimybės reikšmes. PY formato faile yra naudojama *Socket* biblioteka informacijos gavimui ir siuntimui TCP protokolu į VR sistemą, nurodant IP adresą ir tinklo prievado numerį.

PY formato faile iš VRR sistemos gauti baitai yra konvertuojami į 2D paveikslą. Kompiuterio atmintyje yra išsaugomas 2D paveikslas kaip PNG formato failas ir modelio grąžinti rezultatai yra išsaugomi kaip TXT formato failas. „RenderTargerSender“ komponento „TickComponent“ funkcija yra vykdoma kiekvieno kadro metu ir ji nuolatos laukia CNN modelio grąžintų reikšmių siunčiamų baitais. „TickComponent“ funkcijoje yra kviečiama „ReceivedBuffer“ funkcija, kuri nuskaito gautus baitus ir konvertuoja juos į simbolių eilutę (String formato kintamąjį). Funckijos grąžinama simbolių eilutė yra atvaizduojama naudotojui virtualioje scenoje.

4. Virtualios realybės ir dirbtinio intelekto metodų taikymo tyrimas

Buvo atliktas tyrimas siekinat įvertinti virtualios realybės ir dirbtinio intelekto metodų taikymą kognityvinių įgūdžių ir judesių lavinimui su VRR sistemomis. Tyrimo metu buvo atliekamas:

- šiame projekte pasiūlyto CNN modelio rezultatų palyginimas su kitais klasikiniais klasifikavimo modeliais, kai modeliai buvo apmokomi atpažinti virtualios realybės sistemoje surinktus duomenis.
- šiame projekte pasiūlyto CNN modelio rezultatų palyginimas, kai VRR sistemos naudotojo judesių sekimui buvo taikomos skirtingos technologijos.
- Sukurtos VRR sistemos panaudojamumo, protinio ir fizinio darbo krūvio bei įtraukimo į VR aplinką vertinimas.

Tyrime dalyvavo 24 respondentai, kurių amžiaus vidurkis buvo 28,88 metai. Tyrimo tipas buvo kiekybinis. Tyrime naudojamos anketos buvo sudarytos su *Google Forms*.

4.1. Tyrimo priemonės

Šiame poskyryje yra pateiktos tyrime naudotos priemonės, tokios kaip neuroninių tinklų architektūros, reabilitacijos pratimas, rankų judesių sekimo technologijos ir klausimynai.

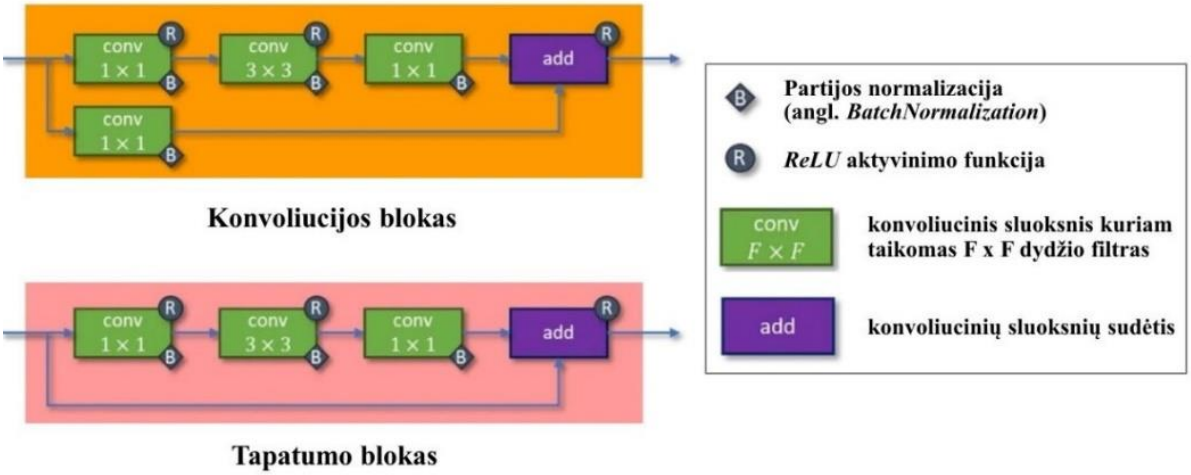
4.1.1. Neuroninių tinklų architektūros

Tyrimui buvo pasirinktos dažniausiai klasifikavimo problemoms spręsti taikomos neuroninių tinklų architektūros naudojamos *ResNet50, VGG16, VGG19, AlexNet* ir *LeNet–5* modeliuose.

ResNet tinkluose (angl. *Residual neural network*) yra naudojami dviejų tipų liekamieji blokai (angl. *Residual blocks*): identiškumo (angl. *Identity*) ir konvoliucijos (žr. 4.1 pav.). Liekamųjų blokų konvoliuciniams sluoksniams yra taikomos *ReLU* aktyvinimo funkcijos ir partijos normalizavimo operacijos. *ResNet* architektūroje yra naudojamos trumposios jungtys, kurios sprendžia nykstančio gradiento problemą giliuosiuose tinkluose [89]. Tapatumo bloko struktūroje (žr. 4.1 pav.), trumpoji jungtis peršoka bloko konvoliucinius sluoksnius ir pateikia pradinius įvesties duomenis jų sudėčiai su paskutinio konvoliucinio sluoksnio išvestimi. Šios sudėties rezultatas yra tapatumo bloko gąžinama išvestis. Konvoliucijos bloko struktūroje yra pavaizduotas panašus trumposios jungties panaudojimas, tik čia jis yra naudojamas peršokti du paskutinius konvoliucinius sluoksnius ir pateikia vieno konvoliucinio sluoksnio išvesties reikšmes sudėties operacijai. 2015 metais sukurto *ResNet 50* tinklo architektūrą sudaro 4 konvoliucijos blokai ir 12 tapatumo blokų. Architektūra yra sudaryta iš 50 sluoksnių. Standartiškai *Resnet 50* tinklas įvestyje priima 224×244×3 formato duomenis ir išveda 1×1000 formato vektorių, kuris saugo 1000 klasių tikimybių reikšmes [89].

VGGNet yra konvoliucinis neuroninis tinklas, kurio architektūra buvo sukurta 2014 metais dėl poreikio mažinti konvoliucinių sluoksnių parametrų kiekį, bei sutrumpinti modelių treniravimosi laiką. Egzistuoja įvairios šio tinklo versijos, tokios kaip *VGG11, VGG13, VGG16* ir *VGG19,* kurios skiriasi jų architektūrose naudojamu sluoksnių skaičiumi. Šiam tyrimui buvo pasirinkti *VGG16* ir *VGG19* modeliai. *VGG16* turi giliojo konvoliucinio neuroninio tinklo architektūrą, kurią sudaro 3 pilnai sujungti sluoksniai ir 13 konvoliucinių sluoksnių. *VGG19* tinklo architektūra skiriasi nuo *VGG16* architektūros tik tuo, kad turi 16 konvoliucinių sluoksnių. Šių tinklų konvoliuciniams sluoksniam yra taikomas 3×3 formato filtro dydis. Tinklų maksimizavimo apjungimo sluoknsiai naudoja 2×2 formato imlų lauką. Paskutiniai abiejų tinklų sluoknsiai taiko *Softmax* aktyvinimo funkciją. Standartiškai

VGG16 ir *VGG19* tinklai įvestyje priima 224×244×3 formato duomenis ir išveda 1×1000 formato vektorių, kuris saugo 1000 klasių tikimybių reikšmes [90].



4.1 pav. *ResNet* tinklo liekamųjų blokų struktūros [R17]

AlexNet tinklas yra gilusis konvoliucinis tinklas, kurio autoriai buvo vieni iš pirmųjų pasiūlyti erdvinės koreliacijos vaizde nagrinėjimą atlikti naudojant konvoliucinius sluoksnius ir imliuosius laukus. Šio 2012 metais sukurto tinklo architektūrą sudaro 5 konvoliuciniai sluoksniai ir 3 pilnai sujungti sluoksniai, kurie turi net 62 milijonus parametrų. Prieš *AlexNet* tinlo atsiradimą, dažniausiai CNN tinkluose buvo naudojamos *Sigmoid* ir *Tanh* aktyvinimo funkcijos. *AlexNet* tinklo sluoksniams buvo taikoma *ReLU* aktyvinimo funkcija, kuri vėliau tapo viena iš populiariausių aktyvinimo funkcijų. Paskutiniam architektūros sluoksniui yra taikoma *Softmax* aktyvinimo funkcija. Architektūroje taip pat yra naudojami du maksimizavimo apjungimo sluoknsiai, kurių imlaus lauko dydis yra 3×3. Standartiškai *AlexNet* tinklas įvestyje priima 224×244×3 formato duomenis ir išveda 1×1000 formato vektorių, kuris saugo 1000 klasių tikimybių reikšmes [91].

LeNet–5 yra konvoliucinis neuroninis tinklas, kurį 1989 metais sukūrė Yann LeCun kartu su bendraautoriais. Neuroninis tinklas *LeNet–5* buvo skirtas ranka parašytiems ir spaudoje naudojamiems simboliams atpažinti. Orginaliai šis tinklas buvo apmokytas su pilkos skalės 32×32 pikselių raiškos skaitmeniniais vaizdais. Tinklo architektūrą sudaro du konvoliuciniai sluoksniai ir trys visiškai sujungti sluoksniai. Konvoliucinių sluoksnių branduoliai šioje architektūroje yra 5×5 dydžio. Po kiekvieno konvoliucinio sluoksnio seka 2×2 dydžio vidurkinimo apjungimo operacijos. Visiems sluoksniams išskyrus paskutinįjį yra taikoma *Tanh* aktyvinimo funkcija. Paskutinis sluoksnis naudoja *Softmax* aktyvinimo funkciją tinklo išvesties gavimui. Standartiškai *LeNet–5* tinklas išveda 1×10 formato vektorių, kuris saugo 10 klasių tikimybių reikšmes [92].

Visų šiame skyrelyje paminėtų architektūrų įvesties ir išvesties formatai buvo pakoreguoti taip, kad atitiktų VRR sistemoje naudojamą duomenų rinkinį.

4.1.2. Reabilitacijos pratimas

Tyrimui buvo paruoštas rankos judėjimo ir kognityvinių įgūdžių lavinimo pratimas, kuris buvo atliekamas virtualiosios realybės aplinkoje. Taikant pasiūlytą VRR sistemą respondentai turėjo atlikti simbolių sekos piešimo užduotį, kurioje yra taikomas analizės dalyje apžvelgtas grįžtamojo ryšio motorinio mokymosi ir kontrolės principas bei atliekamas kognityvinių įgūdžių, tokių kaip dėmėsio, atminties ir kalbos, lavinimas. Respondentai tyrimo pradžioje žodine forma buvo informuoti apie

simbolių seką, kurią jiems reikia atlikti kinestetiniu būdu VRR sistemos aplinkoje. Simbolių seka buvo sudaryta iš raidžių ir skaičių. Respondentai galėjo patys pasirinkti didžiųjų arba mažųjų raidžių simbolių formą.

4.1.3. Rankų judesių sekimo technologijos

Reabilitacijos pratimo atlikimui buvo pasirinktos dvi skirtingos rankos judesių sekimo technolgijos: *Meta Quest 2* įrenginio valdikliai, kuriuos respondentai turėjo laikyti rankose ir *Leap Motion Controller* įrenginys, kuris buvo tvirtinamas ant *Meta Quest 2* virtualiosios realybės akinių. Respondentai reabilitacijos pratimą turėjo atlikti du kartus, kiekvieną kartą taikant skirtingą judesių sekimo tecnhologiją.

4.1.4. Klausimynai

Respondentams pateikiant *Google Forms* platformoje sukurtą klausimyną buvo siekama išsiaiškinti respondentų nuomonę apie tyrimo metu naudotą VRR sistemą. Klausimyno pradžioje respondentai turėjo nurodyti savo amžių ir kaip dažnai jie naudoja virtualią realybę. Tyrime naudotas klausimynas apjungė *SUS* (angl. *System usability scale*), *NASA TLX* (angl. *NASA task load index*) ir *IPQ* (angl. *Igroup presence questionnaire*) klausimynus.

1986 metais John Brooke sudarė *SUS* klausimyną [R18], kuris tyrime buvo naudojamas išmatuoti VRR sistemos panaudojamumą. *SUS* klausimyną sudaro 10 teiginių, kuriuos respondentai turi įvertinti skalėje nuo 1 iki 5, kai 1 žymi „visiškai nesutinku“ ir 5 – „visiškai sutinku“. *SUS* klausimyno naudojimas tapo industrijos standartu vertinti sistemų panaudojamumą ir jį mini daugiau nei 1300 mokslinių straipsnių ir publikacijų.

NASA TLX klausimynas [R19] tyrime buvo naudojamas nustatyti respondetų protinį ir fizinį darbo krūvį VRR sistemos naudojimo metu. Klausimynas yra skaidomas į 6 klausimus, kurie respondentų prašo skalėje nuo 1 iki 20 įvertinti:

- protinį poreikį, kuris nurodo, kiek užduotis reikalavo mąstyti, spręsti ar apskaičiuoti;
- fizinį poreikį, kuris nurodo, kiek fiziškai aktyvi ir intensyvi buvo užduotis;
- laikinąjį poreikį, kuris nurodo, kiek reikėjo skubėti tam, kad užduotis būtų įvykdyta;
- pastangas, kuris nurodo, kaip sunkiai reikėjo stengtis, kad užduotis būtų įvykdyta;
- pasisekimą, kuris nurodo, kaip sekėsi įvykdyti užduotį;
- nusivylimą, kuris nurodo, kaip respondetas savimi nepasitikėjo užduoties atlikimo metu.

IPQ klausimynas [R20] buvo naudojamas nustatyti, kiek tyrime naudota eksperimentinė virtualioji aplinka buvo įtraukianti. Klausimynas padeda išmatuoti tris faktorius:

- erdvinį buvimą, kuris nurodo fizinio buvimo jausmą virtualioje aplinkoje;
- įsitraukimą, kuris nurodo virtualioje aplinkoje skirtą dėmesį;
- realistiškumo jausmą, kuris nurodo, kiek reali buvo virtuali aplinka.

IPQ klausimyną sudaro 14 teiginių, kuriuos respondentai turi įvertinti skalėje nuo 1 iki 5.

4.2. Tyrimo eiga

Šiame poskyryje yra pateikiama vykdyto tyrimo eiga, kurią sudaro neuroninių tinklų modelių apmokymas, eksperimentų su VRR sistema vykdymas ir surinktų duomenų apdorojimas.

4.2.1. Neuroninių tinklų modelių apmokymas

Tyrime naudotų modelių apmokymui buvo naudojamas papildytas *EMNIST Balanced* duomenų rinkinys. Modelių treniravimo procese buvo taikomi parametrai aprašyti 4.1 lentelėje. Šie parametrai taip pat buvo naudojami projekte sukurtų CNN modelių parametrų derinimo procese ir leido pasiekti geriausius rezultatus modeliui „model_1“.

4.1 lentelė. Modelių treniravimo metu taikyti parametrai

Parametras	**Parametro vertės**	**Parametro aprašymas**
grįžtamojo ryšio objektas *EarlyStopping*	„min_delta = 0“; „patience = 0“; „restore_best_weights = True“; „monitor = val_accuracy“.	Sustabdo treniravimo procesą epochoje, kurioje modelio tikslumas nustoja didėti.
grįžtamojo ryšio objektas *ReduceLROnPlateau*	„monitor = val_loss“; „patience = 3“; „factor = 0,2“; „min_lr = 0,0001“.	Naudojamas kontroliuoti modelių mokymosi greitį.
grįžtamojo ryšio objektas *ModelCheckpoint*	“monitor = val_accuracy”	Naudojamas išsaugoti modelio svorių rinkinį skirtinguose treniravimo etapuose.
Optimizatorius	„rmsprop“	Naudojamas sumažinti modelio treniravimo metu gaunamas praradimo reikšmes.
Praradimo (angl. Loss) funkcija	„categorical_crossentropy“	Naudojama įvertinti modelio aptiktų ir tikrų reikšmių skirtumus (nepanašumus) ir juos sumažinti.
Partijos dydis	64	Nurodo duomenų rinkinio eilučių (nuotraukų) kiekį vienoje mokymo iteracijoje.

4.2.2. Eksperimentai su VRR sistema

Kauno technologijos universiteto patalpose buvo atliekami tyrimo eksperimentai, kurių pradžioje respondentai buvo supažindinami su tyrimo procesu ir tvarka. Respondentams buvo papasakota apie rankų sekimo įrenginius ir kaip juos reikės naudoti eksperimentų metu. Respondentams buvo leidžiama pasisžvalgyti virtualioje aplinkoje prieš pradedant eksperimentus, taip pat buvo papasakojama apie aplinkoje esančius mygtukus bei kaip juos bus galima naudoti.

Aplinkoje buvo raudonos ir žalios spalvos mygtukai. Raudonas mygtukas leisdavo pradėti simbolį piešti nuo pradžių, jei dėl kokios nors priežasties nepavyko nupiešti simbolio iš vieno karto. Žalias mygtukas respondetui leisdavo pažymėti simbolio piešimo pabaigimą.

Respondentams buvo papasakota, kurioje virtualios aplinkos vietoje jie galės stebėti CNN modelio grąžinamus rezultatus apie jų nupieštus simbolius ir šių rezultatų prasmę. CNN modelis grąžindavo dvi reikšmes apie nupieštą simbolį: atpažintą simbolio klasę ir priklausymo šiai klasei tikimybės reikšmę išreikštą procentais.

Respondentai buvo supažindinti su „RESPUBLIKA0512“ simbolių seka, kurią jiems reikės nupiešti virtualioje aplinkoje. Simbolių piešimo užduotį respondentai atliko sėdimoje padėtyje, jų judėjimui buvo skirta 2×2 metrų ploto erdvė. Prieš pradedant eksperimentus respondentai galėjo pabandyti nupiešti daugiausiai tris jų pasirinktus simbolius ir išmėginti aplinkoje esančių mygtukų veikimą. VRR sistema veikė asmeniniame kompiuteryje, kuris turėjo *Intel Core i7-8700K* pagrindinį procesorių,

NVIDIA GeForce RTX 2080 Ti grafinį procesorių ir *Windows 10* operacinę sistemą. Respondentams baigus visus ekspirementus, jų buvo paprašyta užpildyti klausimyną apie naudotą VRR sistemą.

4.2.3. Duomenų apdorojimas

Tyrimo metu pagal respondentų atsakymus į *SUS, NASA TLX* ir *IPQ* klausimynuose pateiktus teiginius, buvo suformuoti šių klausimynų įvertinimai. Įvertinimai buvo skaičiuojami pagal rezultatų interpretavimo metodikas, kurios buvo pateiktos oficialiuose klausimynų kūrėjų tinklalapiuose [93, 94, 95].

Respondetų *SUS* klausimyno atsakymai, kurie buvo intervale [1; 5], buvo pakeisti į [0; 4] intervalą. Iš klausimyno 1, 3, 5, 7 ir 9 teiginių respondetų atsakymų buvo atimama 1 reikšmė. Iš 5 reikšmės buvo atimami klausimyno 2, 4, 6, 8 ir 10 teiginių respondetų atsakymų reikšmės. Kiekvieno respondento *SUS* klausimyno teiginių atsakymai buvo sumuojami ir padauginti iš 2,5 reikšmės. *SUS* klausimyno galutinio įvertinimo gavimui buvo rastas visų atsakymų daugybos rezultatų vidurkis, kurio reikšmė buvo intervale [0; 100].

NASA TLX klausimyne pateiktiems 6 teiginiams respondetai turėjo nurodyti reikšmę esančią intervale [1; 21]. Atsakymų reikšmės buvo normalizuotos taip, kad būtų pasiskirstę intervale [0; 100]. Buvo apskaičiuotas kiekvieno klausimyno teiginio respondentų atsakymų vidurkis, kuris buvo vertinamas pagal 4.2 lentelėje pateiktas rezultatų interpretavimo reikšmes.

4.2 lentelė. *NASA TLX* klausimyno rezultatų interpretavimas

Protinis ir fizinis darbo krūvis	**Reikšmė**
Mažas	Nuo 0 iki 9
Vidutiniškas	Nuo 10 iki 29
Šiek tiek didelis	Nuo 30 iki 49
Didelis	Nuo 50 iki 79
Labai didelis	Nuo 80 iki 100

Respondetų *IPQ* klausimyno atsakymai, kurie buvo intervale [1; 5], buvo pakeisti į [0; 4] intervalą. Klausimyno 3, 7 ir 9 teiginių respondetų atsakymų reikšmės buvo invertuojamos (pvz., 0 reikšmė pakeista į 4). Iš visų kitų klausimyno teiginių atsakymų buvo atimama 1 reikšmė. Klausimyno teiginiai buvo padalinti į tris kategorijas, kurių pavadinimai buvo „erdvinis buvimas“, „tikroviškumo jausmas“ ir „įsitraukimas“. Kiekvienos kategorijos rezultatams buvo apskaičiuojamas kategorijos teiginiams parinktų atsakymų vidurkis. Gautų rezultatų reikšmės buvo normalizuotos taip, kad būtų pasiskirstę intervale [0; 100].

4.3. Tyrimo rezultatai

Šiame poskyryje yra pateikiami atlikto tyrimo rezultatai, kurie gali būti skirstomi į neuroninių tinklų modelių įvertinimo, rankų judesių sekimo technologijų taikymo ir VRR sistemos vertinimo rezultatus.

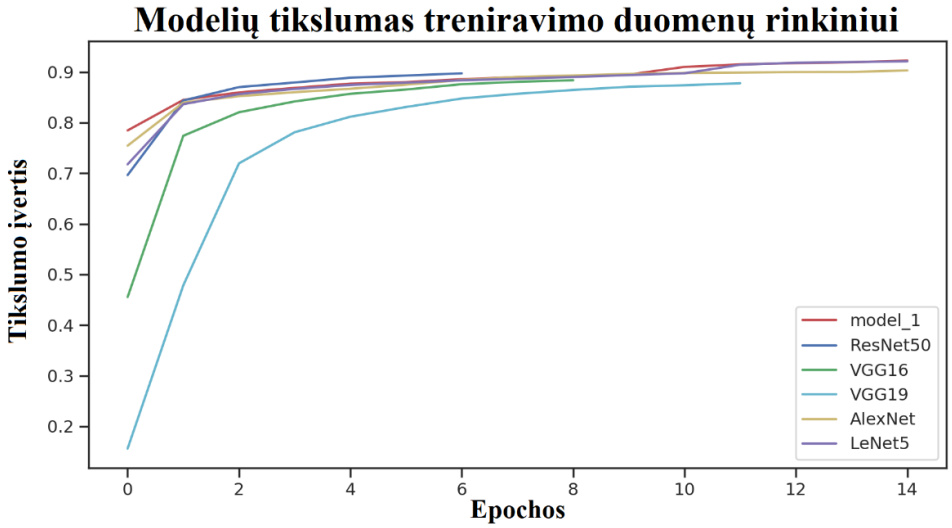
4.3.1. Neuroninių tinklų modelių įvertinimas

Atlikus *ResNet50, AlexNet, LeNet–5, VGG16, VGG19* ir „model_1“ modelių treninavimą su papildytu *EMNIST Balanced* duomenų rinkiniu buvo atliktas modelių įvertinimas su rinkinio validacijos duomenimis. 4.2 lentelėje yra pavaizduoti gauti modelių praradimo, tikslumo, preciziškumo ir atkūrimo metrikų įvertinimai.

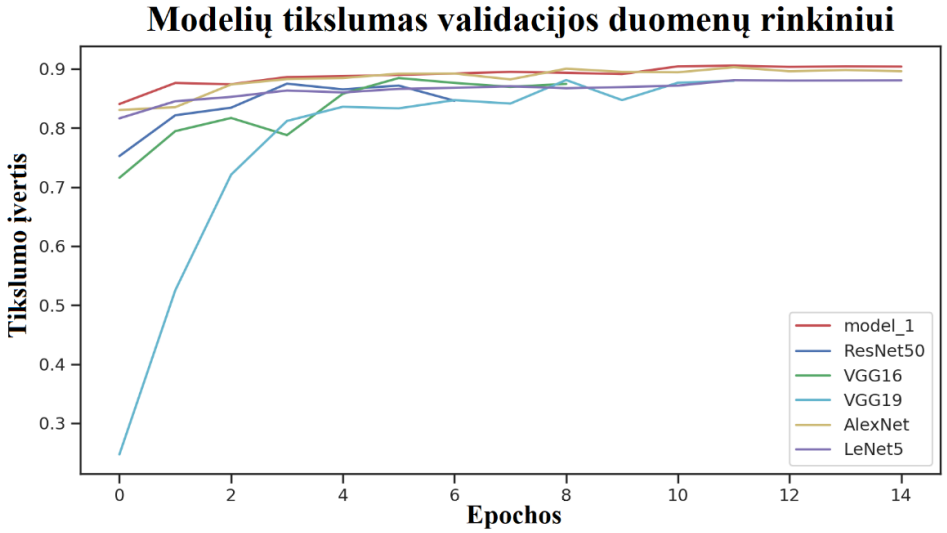
4.3 lentelė. Modelių rezultatai papildytam *EMNIST Balanced* duomenų rinkiniui

Modelio pavadinimas	Aktyvinimo funkcija	Praradimas	Tikslumas	Preciziškumas	Atkūrimas
ResNet50	*ReLU*	0,3630	0,8750	0,8924	0,8626
AlexNet	*ReLU*	0,3058	0,9029	0,9220	0,8827
LeNet–5	*Tanh*	0,3396	0,8810	0,9032	0,8629
VGG16	*ReLU*	0,3802	0,8846	0,9171	0,8488
VGG19	*ReLU*	0,3911	0,8812	0,9183	0,8424
model_1	*LeakyReLU*	0,2797	0,9056	0,9201	0,8917

Tyrimo rezultatams pavaizduoti buvo sugeneruoti modelių tikslumo reikšmių pasikeitimo kiekvienoje epochoje grafikai, kai modeliams buvo taikomi treniravimo ir validacijos duomenų rinkiniai. 4.2 ir 4.3 paveiksluose pavaizduotų grafikų generavimui buvo pasinaudota *Matplotlib* bibliotekos *Pyplot* funkcijų rinkiniu.



4.2 pav. Modelių tikslumo treniravimo duomenų rinkiniui reikšmių pasikeitimo grafikas



4.3 pav. Modelių tikslumo validacijos duomenų rinkiniui reikšmių pasikeitimo grafikas

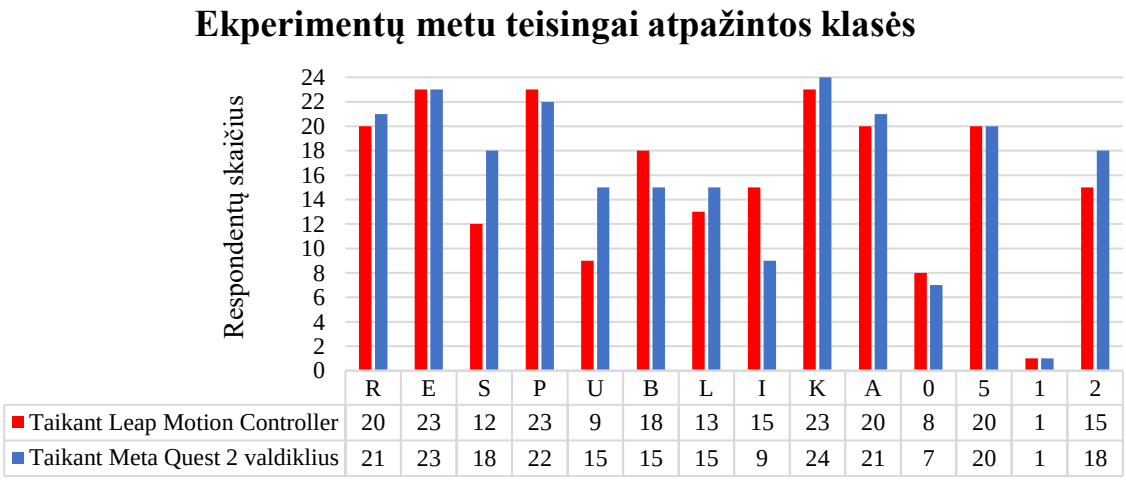
Tyrimas parodė, kad geriausi rezultatai buvo pasiekti su šiame projekte sukurtu konvoliucinio neuroninio tinklo modeliu „model_1“. Šio modelio tikslumo įvertinimas validacijos duomenims buvo 3,38 % geresnis už *ResNet50* modelio, 0,3 % geresnis už *AlexNet* modelio, 2,72 % geresnis už *LeNet–5* modelio, 2,32 % geresnis už *VGG16* modelio ir 2,69 % geresnis už *VGG19* modelio. Modelio „model_1“ praradimo ir atkūrimo įvertinimai taip pat buvo geresni už visų minėtų modelių. Pagal 4.2 lentelėje pateiktus rezultatus *AlexNet* turėjo geriausią preciziškumo įvertinimą validacijos duomenims. Šio modelio preciziškumo įvertinimas buvo 0,21 % didesnis už modelio „model_1“.

4.3.2. Rankų judesių sekimo technologijų taikymo su VRR sistema rezultatai

Taikant skirtingas rankų judėjimo sekimo technologijas kiekvienam respondento sukurtam simboliui VRR sistemoje buvo grąžinamos CNN modelio atpažintos simbolio klasės ir priklausymo šiai klasei tikimybės reikšmės. 4.4 pav. pavaizduotas grafikas, parodo kiek kartų CNN modelis teisingai atpažino atskiras simbolių klases, kai tyrimo metu respondentų rankų judesių sekimui buvo naudojami *Meta Quest 2* įrenginio valdikliai ir *Leap Motion Controller* įrenginys. Iš pateikto grafiko galima matyti, kad CNN modeliui geriausiai sekėsi atpažinti „E“, „P“ ir „K“ simbolių klases, kai eksperimentai buvo atliekami su *Leap Motion Controller* įrenginiu. Iš 24 atliktų eksperimentų su šiuo įrenginiu, modelis 23 eksperimentų metu teisingai atpažino šias klases.

Grafike taip pat yra pavaizduota, kad CNN modeliui geriausiai sekėsi atpažinti „K“ simbolių klasę, kai eksperimentai buvo atliekami su *Meta Quest 2* įrenginio valdikliais. Visuose atliktuose eksperimentuose naudojant šį įrenginį, „K“ klasę modelis atpažino teisingai.

Naudojant abi rankų judesių sekimo technologijas modelis tik vieno eksperimento metu, teisingai atpažino simbolio „1“ klasę. Tokiems prastiems CNN modelio rezultatams atpažinti „1“ klasę, galėjo turėti įtakos gautas mažas, 72,25 % tikslumas „1“ klasei, kai modelis buvo įvertintas su testavimo duomenimis iš papildyto *EMNIST Balanced* duomenų rinkinio.



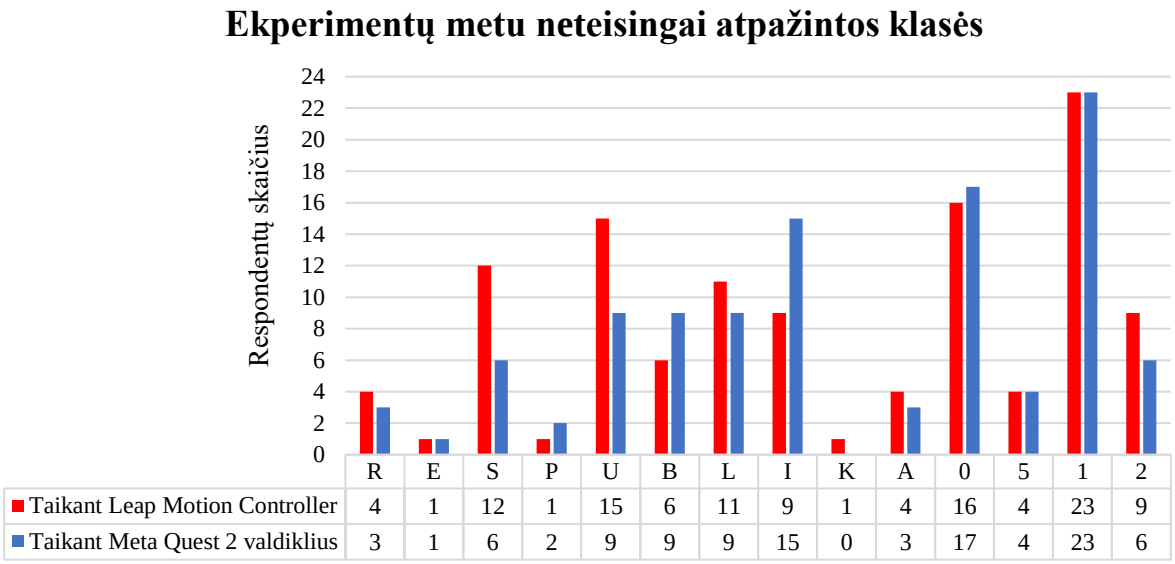
	R	E	S	P	U	B	L	I	K	A	0	5	1	2
Taikant Leap Motion Controller	20	23	12	23	9	18	13	15	23	20	8	20	1	15
Taikant Meta Quest 2 valdiklius	21	23	18	22	15	15	15	9	24	21	7	20	1	18

4.4 pav. Eksperimentų metu CNN modelio teisingai atpažintų klasių pasiskirstymo grafikas

Iš 4.5 pav. pateikto grafiko galima matyti, kad CNN modeliui prasčiausiai sekėsi atpažinti „1“, „0“ ir „U“ simbolių klases, kai eksperimentai buvo atliekami su *Leap Motion Controller* įrenginiu. Iš 24 atliktų eksperimentų, 23 eksperimentus modelis klaidingai atpažino „1“ klasę, 16 eksperimentų klaidingai atpažino „0“ klasę ir 15 eksperimentų klaidingai atpažino „U“ klasę. Tokiems rezultatams „0“ klasei atpažinti galėjo turėti įtakos tai, kad vertinant modelį su testavimo duomenimis iš papildyto

EMNIST Balanced duomenų rinkinio, modelis „0“ klasę atpažino tik 65,5 % tikslumu. Nepaisant to, kad modelio tikslumas atpažinti „U“ klasę iš papildyto *EMNIST Balanced* rinkinio duomenų buvo 92,25 %, modelis net 15 eksperimentų iš 24 klaidingai atpažino šią klasę.

4.5 pav. yra pavaizduota, kad CNN modeliui prasčiausiai sekėsi atpažinti „1“, „0“ ir „I“ simbolių klases, kai eksperimentai buvo atliekami su *Meta Quest 2* įrenginio valdikliais. Iš 24 atliktų eksperimentų, 23 eksperimentus modelis klaidingai atpažino „1“ klasę, 17 eksperimentų klaidingai atpažino „0“ klasę ir 15 eksperimentų klaidingai atpažino „I“ klasę. CNN modelio rezultatams atpažinti „I“ klasę, galėjo turėti įtakos gautas mažas, 64,5 % tikslumas klasei „I“, kai modelis buvo įvertintas su testavimo duomenimis iš papildyto *EMNIST Balanced* duomenų rinkinio.



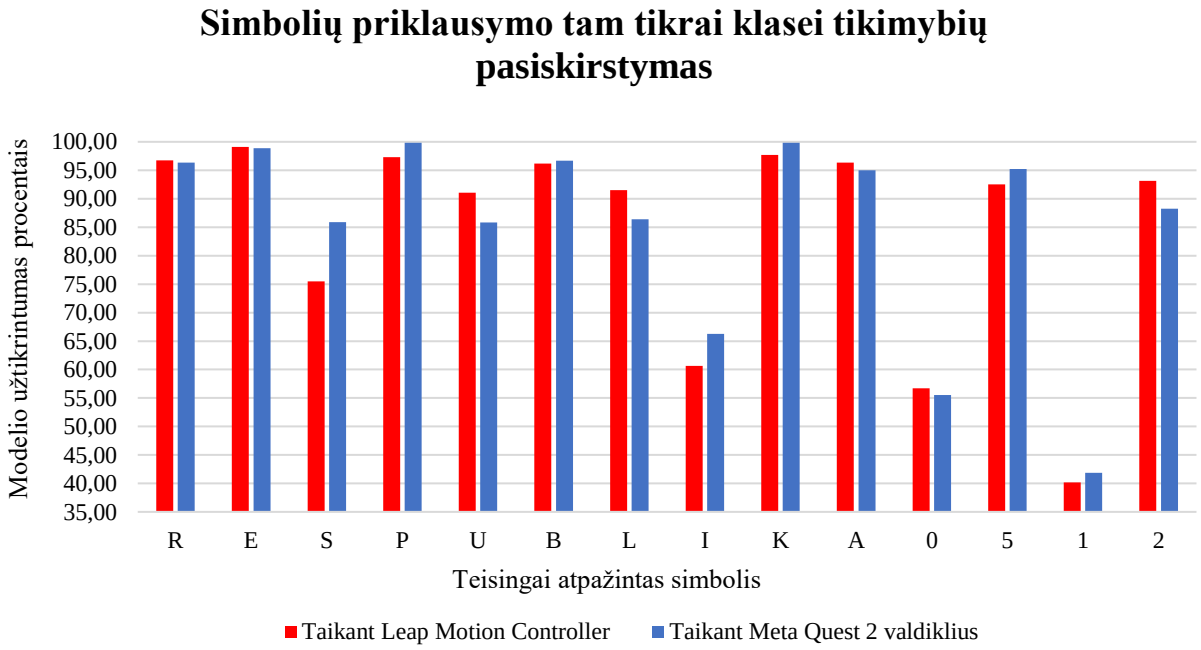
	R	E	S	P	U	B	L	I	K	A	0	5	1	2
Taikant Leap Motion Controller	4	1	12	1	15	6	11	9	1	4	16	4	23	9
Taikant Meta Quest 2 valdiklius	3	1	6	2	9	9	9	15	0	3	17	4	23	6

4.5 pav. Eksperimentų metu CNN modelio neteisingai atpažintų klasių pasiskirstymo grafikas

Peržiūrėjus bendrus eksperimentų rezultatus, buvo pastebėta kad modelis iš 672 simbolių, 223 simboliams (33,18 % simbolių) grąžino klaidingą simbolių klasę. Modelis 5,8 % simbolių klaidingai priskyrė „spiralės“ klasės reikšmę, 3,87 % simbolių klaidingai priskyrė „T“ klasės reikšmę, 3,13 % simbolių klaidingai priskyrė „L“ klasės reikšmę ir 2,68 % simbolių klaidingai priskyrė „I“ klasės reikšmę.

4.6 pav. vaizduoja CNN modelio gražintų simbolių priklausymo tam tikroms klasėms tikimybių reikšmių vidurkius. Grafike pavaizduota, kad respondentų rankų judesių sekimui taikant *Leap Motion Controler* įrenginį, modelis grąžindavo vidutiniškai 99,12 % tikimybę eksperimentų metu sukurtiems „E“ klasės simboliams priklausyti teisingai klasei ir 40,16 % tikimybę „1“ klasės simboliams priklausyti teisingai klasei.

Respondentų rankų sekimui taikant *Meta Quest 2* įrenginio valdiklius, modelis grąžindavo vidutiniškai 99,86 % tikimybę eksperimentų metu sukurtiems „K“ klasės simboliams priklausyti teisingai klasei ir 41,85 % tikimybę „1“ klasės simboliams priklausyti „1“ klasei.



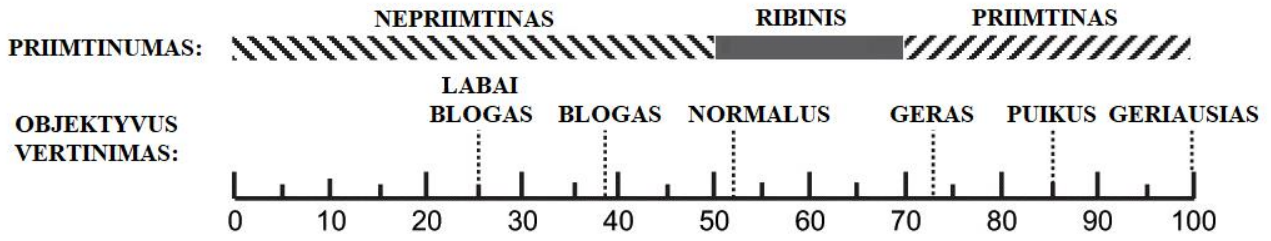
4.6 pav. Simbolių priklausymo tam tikrai klasei tikimybių pasiskirstymo grafikas

Eksperimentų metu naudojant *Meta Quest 2* įrenginio valdiklius respondentai VRR sistemoje iš viso sukūrė 336 simbolių atvaizdus. 65,48 % šių atvaizdų CNN modelis priskyrė teisingoms simbolių klasėms. Eksperimentų metu naudojant *Leap Motion Controler* įrenginį, modelis 68,15 % sukurtų simbolių teisingai priskyrė simbolių klases. CNN modeliui 2,68 % geriau sekėsi atpažinti vaizdus, kai simbolių kūrimui buvo naudoti *Meta Quest 2* įrenginio valdikliai.

Minėtiems rezultatams galėjo turėti įtakos tai, kad didelė dalis respondentų eksperimentų metu manė, kad *Meta Quest 2* įrenginio valdiklius buvo paprasčiau ir patogiau naudoti, nes rankų judesiai buvo greičiau atpažįstami ir atvaizduojami virtualioje aplinkoje, nei naudojant *Leap Motion Controler* įrenginį. Respondentai, taip pat pasisakė, kad eksperimentų metu jiems buvo sunku naudoti *Leap Motion Controler* įrenginį, nes rankoms esant toliau nei 60 cm nuo įrenginio, rankų judesiai nebebūdavo atvaizduojamaji virtualioje aplinkoje ir dažniau tekdavo pasinaudoti aplinkoje esančiu, simbolio ištrynimo mygtuku, kuris leisdavo pradėti piešimą nuo pradžių. Svarbu paminėti rezultatams galėjo turėti įtakos ir tai, kad didžioji dalis respondentų (45,8 %) klausimyne nurodė, kad VR naudojasi bent kartą į metus, todėl nebuvo linkę dažnai naudotis virtualiosios realybės įranga ir sistemomis.

4.3.3. VRR sistemos vertinimas pagal klausimyno rezultatus

VRR sistemos panaudojamumui buvo gautas 78,13 balų *SUS* klausimyno rezultatų įvertinimas, kuris buvo apskaičiuotas taikant *SUS* klausimyno vertinimo metodiką. Pagal 4.7 pav. pavaizduotą *SUS* klausimyno vertinimo skalę sukurtos sistemos vertinimas patenka į „geras“ ir „puikus“ skalės ribą, kuri yra nuo 72,5 iki 85 balų. VRR sistema pagal šią vertinimo skalę taip pat yra priimtina, nes gautas panaudojamumo įvertis yra aukštesnis nei 70 balų.

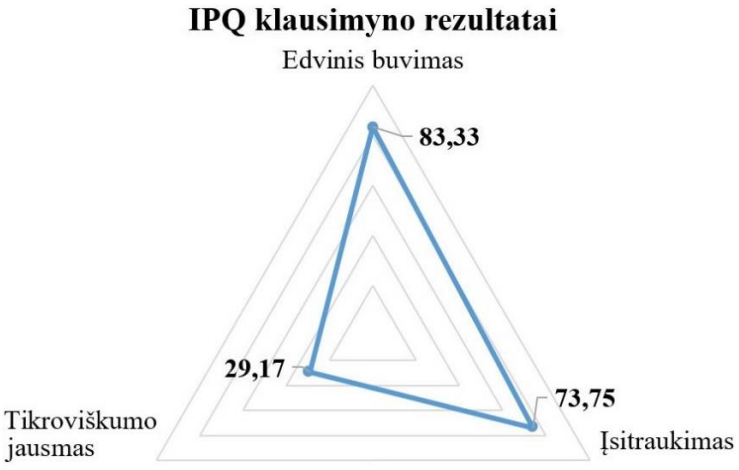


4.7 pav. *SUS* klausimyno vertinimo skalė [R18]

NASA TLX klausimyno rezultatai buvo sugrupuoti į 6 grupes, kurioms buvo gauti atskiri įvertinimai. Taikant *NASA TLX* klausimyno vertinimo metodikas buvo gauti šie įvertinimai: 12,01 balų protinis poreikis, 23,33 balų fizinis poreikis, 7,92 balų laikinasis poreikis, 35,42 balų pasitenkinimas užduoties atlikimu, 30,63 balų įdėtos pastangos ir 10,83 balų nusivylimas. Gauti rezultatai buvo interpretuojami pagal 4.2 lentelėje pateiktas *NASA TLX* klausimyno rezultatų reikšmes:

- VRR sistema respondentų reikalavo mažo lygio laikinojo poreikio, kuris nurodo, kiek respondetai jautė, kad eksperimentų metu turėjo skubėti įvykdyti užduotį;
- VRR sistema respondentų reikalavo vidutinio lygio protinio ir fizinio poreikio, eksperimentų atlikimo metu mąstyti, spręsti ir būti fiziškai aktyviems;
- Respondentai savo pasisekimą atlikti užduotis su VRR sistema vertino prastai;
- Respondentai jautė, kad užduočių atlikimas su VRR sistema reikalavo nuo vidutinių iki didelių pastangų;
- VRR sistemos naudojimas respondentams sukėlė vidutinio lygio susierzinimą, nusivylimą savimi ir stresą.

Pagal *NASA TLX* klausimyno rezultatų vertinimo metodikas buvo paskaičiuotas bendras 20,03 balų vertės protinio ir fizinio darbo krūvio įvertinimas VRR sistemai. Šis įvertinimas nurodo, kad naudojimasis VRR sistema reikalauja vidutinio lygio darbo krūvio.



4.8 pav. *IPQ* klausimyno rezultatai

4.8 pav. yra pavaizduoti *IPQ* klausimyno respondentų atsakymams gauti trijų kategorijų įvertinimai. Šių įvertinų reikšmės yra [0; 100] intervale. Geriausiai buvo įverinta „erdvinio buvimo“ kategorija, kuriai buvo paskaičiuotas 83,33 balų įvertinimas. Prasčiausiai buvo įvertinta „tikroviškumo jausmo“ kategorija, kuriai buvo paskaičiuotas 29,17 balų įvertinimas. „Įsitraukimo“ kategorijai buvo gautas

73,75 balų įvertinimas. Pagal šiuos įvertinimus galima daryti išvadą, kad respondentai eksperimetų metu gerai jautė juos supantį virtualų pasaulį ir mažai dėmesio kreipė į tikrąją aplinką, tačiau VRR sistemos aplinka nebuvo pakankamai realistiška ir panaši tikrą pasaulį.

4.4. Tyrimo apibendrinimas

Tyrimui buvo paruoštas rankos judėjimo ir kognityvinių įgūdžių lavinimo pratimas, kuris buvo atliekamas sukurtos VRR sistemos virtualiosios realybės aplinkoje. Taikant VRR sistemą respondentai turėjo atlikti simbolių sekos piešimo užduotį, kurioje yra taikomas grįžtamojo ryšio motorinio mokymosi ir kontrolės principas bei atliekamas kognityvinių įgūdžių, tokių kaip dėmėsio, atminties ir kalbos, lavinimas.

Reabilitacijos pratimo atlikimui buvo pasirinktos dvi skirtingos rankos judesių sekimo technolgijos: *Meta Quest 2* įrenginio valdikliai, kuriuos respondentai turėjo laikyti rankose ir *Leap Motion Controller* įrenginys, kuris buvo tvirtinamas ant *Meta Quest 2* virtualiosios realybės akinių.

Tyrimo metu taikant rankų judesių sekimo technologijas ir registruojant respondentų rankų judesius trimatėje erdvėje buvo sukuriami judesius reprezentuojantys 2D vaizdai, kurie buvo pateikiami CNN modeliui. CNN modelis gautus vaizdus įvertindavo ir VRR sistemai grąžindavo atpažintas duomenų klases, bei priklausymo aptiktoms klasėms tikimybes.

Projekte pasiūlytas konvoliucinio neuroninio tiklo modelis „model_1“ su papildyto *EMNIST Balanced* rinkinio validacijos duomenimis pasiekė tikslumą, kuris buvo 3,38 % geresnis už *ResNet50* modelio, 0,3 % geresnis už *AlexNet* modelio, 2,72 % geresnis už *LeNet–5* modelio, 2,32 % geresnis už *VGG16* modelio ir 2,69 % geresnis už *VGG19* modelio. Modelio „model_1“ tikslumas atpažinti papildyto *EMNIST Balanced* rinkinio validacijos duomenimis buvo įvertintas 0,9056 verte, kuri nurodo kad 90,56 % duomenų modelis atpažino teisingai.

Įvertinus eksperimentų metu surinktų duomenų atpažinimo rezultatus, buvo nustatyta, kad modelis teisingai atpažino 66,82 % duomenų. Eksperimentų metu VRR sistemą naudojant su *Leap Motion Controler* įrenginiu, 65,48 % duomenų modelis priskyrė tinkamoms klasėms. CNN modeliui 2,68 % geriau sekėsi atpažinti vaizdus, kai VRR sistemoje duomenų kūrimui buvo naudoti *Meta Quest 2* įrenginio valdikliai. Šiems rezultatams galėjo turėti įtakos tai, kad didelė dalis respondentų eksperimentų metu manė, kad *Meta Quest 2* įrenginio valdiklius buvo papraščiau ir patogiau naudoti, nes rankų judesiai buvo greičiau atpažįstami ir atvaizduojami virtualioje aplinkoje, nei naudojant *Leap Motion Controler* įrenginį. Taip pat svarbu paminėti, kad didžioji dalis respondentų nebuvo linkę dažnai naudotis virtualiosios realybės įranga ir sistemomis.

Pagal *SUS* klausimyno vertinimo skalę sukurtos sistemos panaudojamumas yra nuo „geras“ iki „puikus“ *SUS* skalės ribose. Pagal *NASA TLX* klausimyno rezultatų interpretavimo metodikas buvo nustatyta, kad VRR sistema reikalauja vidutinio lygio darbo krūvio ja naudotis. *IPQ* klausimyno rezultatai parodė, kad respondentai eksperimetų metu gerai jautė juos supantį virtualų pasaulį ir mažai dėmesio kreipė į tikrąją aplinką, tačiau VRR sistemos aplinka nebuvo pakankamai realistiška ir panaši tikrą pasaulį.

Išvados

1. Išanalizavus realybėje taikomus judesių ir kognityvinių įgūdžių reabilitacijos metodus buvo pastebėta, kad lyginant su tradiciniais reabilitacijos metodais, VR naudojimas reabilitacijos procese gali suteikti aukštesnio pasikartojimo ir intensyvumo užduotis, objektyvų grįžtamąjį ryšį ir padidinti paciento įsitraukimą bei motyvaciją.
2. Išanalizavus dirbtinio intelekto algoritmus, kurie gali būti pritaikomi žmogaus judesiams atpažinti, buvo pastebėta, kad pasikartojantys neuroniniai tinklai yra labiau tinkami atpažinti trumpą veiklą, kuri turi natūralią tvarką, o konvoliuciniai neuroniniai tinklai labiau yra tinkami atpažinti ilgalaikę pasikartojančią veiklą. Todėl kuriamai VRR sistemai nuspręsta taikyti konvoliucinius neuroninius tinklus.
3. Išanalizavus žmogaus judėjimo sekimo technologijas, kurios gali būti taikomos virtualios realybės sistemoms, buvo pastebėta, kad *LiDAR* ir optinio sekimo technologijos gali būti naudojamos kartu, tam kad būtų užtikrintas ypač aukšto tikslumo rankų judesių sekimas, todėl nuspręsta šias technologijas panaudoti kuriamoje VRR sistemoje.
4. *Unreal Engine* žaidimų variklyje sukurta projekcijos transformacijos apskaičiavimo funkcija leido sistemos naudotojo rankų vietos trimatėje erdvėje informaciją projektuoti į dvimatės plokštumos tekstūros paveikslo erdvę ir taip įgyvendinti piešimo pratimą VRR sistemoje.
5. *Unreal Engine* žaidimų variklio *Blueprint* klasės komponentui parinkus IP adreso ir tinklo prievado numerio parametrus, buvo galima VRR sistemoje surinktus duomenis siųsti TCP protokolu į asmeniniame kompiuteryje veikiančią programą, kuri gautus duomenis pateikdavo konvoliucinio neuroninio tinklo modeliui ir grąžindavo duomenų klasifikavimo rezultatus. Šio duomenų siuntimo metodo taikymas sukurtoje VRR sistemoje veikė sklandžiai ir nesukėlė trikdžių.
6. Ištyrus sukurto CNN modelio ir klasikinių klasifikavimo modelių tikslumą atpažinti papildytą *EMNIST Balanced* duomenų rinkinį buvo nustatyta, kad sukurtas CNN modelis pasiekė apytiksliai 0,91 tikslumo įvertinimą, kuris buvo vidutiniškai 2,28 % geresnis už kitus tyrime naudotus modelius.
7. Ištyrus judesių sekimo technologijų įtaką sukurto modelio tikslumui buvo nustatyta, kad taikant *Leap Motion Controler* įrenginio ir *Meta Quest 2* įrenginio valdiklius rankų judesių registravimui, *Meta Quest 2* įrenginio valdiklių naudojimas lėmė tik 2,68 % aukštesnį modelio tikslumą. Tokie rezultatai parodė, kad modelio tikslumas išliko panašus naudojant abi technologijas.
8. Ištyrus 24 respondentų atsakymus į pateiktus *SUS* klausimyno teiginius, buvo nustatytas 78,13 balų sukurtos sistemos panaudojamumo įvertinimas, kuris nurodo, kad pagal *SUS* klausimyno vertinimo skalę, sistemos panaudojamumas yra nuo „geras“ iki „puikus“ *SUS* skalės ribose, tačiau sistemos funkcionalumą dar galima patobulinti atsižvelgiant į respondetų atsiliepimus.
9. Kartu su bendraautoriais virtualiosios realybės ir dirbtinio intelekto metodų panaudojimo temomis buvo pristatyti 2 straipsniai:
   - „A System for Providing Educational Content in Virtual Reality“ pristatytas ALTA’21 konferencijoje;
   - „Influence of Aerial Image Resolution on Vehicle Detection Accuracy“ pristatytas IVUS 2023 konferencijoje.

Literatūros sąrašas

1. MAZURYK T. and GERVAUTZ M. *Virtual Reality – History, Applications, Technology and Future*. 1999 [žiūrėta 2021-12-01]. Prieiga per: Research Gate.
2. VIRTUAL REALITY SOCIETY. *Advantages of Virtual Reality in Medicine*. 2015 [žiūrėta 2021-12-01]. Prieiga per: https://www.vrs.org.uk/virtual–reality–healthcare/advantages.html.
3. HOWARD C. *A meta–analysis and systematic literature review of virtual reality rehabilitation programs*. Computers in Human Behavior. 2017, 70 (1), 317–327 [žiūrėta 2021-12-01]. ISSN 0747-5632.
4. LAVER K. E., LANGE B. and GEORGE S. *Virtual reality for stroke rehabilitation*. Cochrane Database of Systematic Reviews. 2017 [žiūrėta 2021-12-03]. ISSN 0747-5632. doi: 10.1002/14651858.CD008349.pub4.
5. YUAN Y., HUANG J. and YAN K. *Virtual Reality Therapy and Machine Learning Techniques in Drug Addiction Treatment*. 2019, 241–245 [žiūrėta 2021-12-03]. doi: 10.1109/ITME.2019.00062.
6. MURATORI L., LAMBERG E., QUINN L. and DUFF S. *Applying principles of motor learning and control to upper extremity rehabilitation*. J Hand Ther. 2013, 26 (2), 94–103 [žiūrėta 2021-12-04]. doi:10.1016/j.jht.2012.12.007.
7. MAGILL, R. A. *Motor learning and control: Concepts and Applications*. 9–oji laida. McGraw Hill, 2011[žiūrėta 2021-12-06]. Prieiga per: Scientific Research SCRIP.
8. GENTILE A. M. *A working model of skill acquisition with application to teaching*. Quest. 1972, 17 (1), 3–23 [žiūrėta 2021-12-06]. doi: 10.1080/00336297.1972.10519717.
9. WINSTEIN C. J. and WOLF S. L. *Task-oriented training to promote upper extremity recovery*. JAMA. 2016, 14 (6), 33–44 [žiūrėta 2021-12-06]. doi: 10.1001/jama.2016.0276.
10. WADDELL K. J., BIRKENMEIER R. L., MOORE J. L., HORNBY T. G. and LANG C. E. *Feasibility of High-Repetition, Task-Specific Training for Individuals With Upper-Extremity Paresis*. American Journal of Physical Medicine & Rehabilitation. 2014, 93 (10), 857-866 [žiūrėta 2021-12-06]. doi: 10.5014/ajot.2014.011619.
11. Magill R. A., Hall K. G. *A review of the contextual interference effect in motor skill acquisition*. Human Movement Science. 1990, 9 (3–5), 241–289 [žiūrėta 2021-12-10]. ISSN 0167-9457. doi: 10.1016/0167-9457(90)90005-X.
12. PLESSIS S. D. *16 Cognitive Skills that Matter, How to Improve Them*. Cognitive Skills and Study Methods, Edublox Research. 2022 [žiūrėta 2021-12-10]. Prieiga per: https://www.edubloxtutor.com/cognitive-skills/.
13. KATZ D. I., ASHLEY M. J., O'SHANICK G. J. and CONNORS S. H. *Cognitive Rehabilitation: The Evidence, Funding and Case for Advocacy for Brain Injury*. McLean: Brain Injury Association of America. 2006 [žiūrėta 2021-12-10].
14. CERNICH A. N., KURTZ S. M., MORDECAI K. L. and RYAN P. B. *Cognitive rehabilitation in traumatic brain injury*. Curr Treat Options Neurol. 2010, 12 (5), 412–23 [žiūrėta 2021-12-10]. doi: 10.1007/s11940-010-0085-6.
15. MAHER C. 15 *Helpful Cognitive Rehabilitation Exercises to Sharpen Your Mind*. 2020 [žiūrėta 2021-12-10]. Prieiga per: https://www.flintrehab.com/cognitive-exercises-tbi/.
16. YEUNG A. W. K., TOSEVSKA A., KLAGER E. and EIBENSTEINER F. *Virtual and Augmented Reality Applications in Medicine: Analysis of the Scientific Literature*. JMIR Publications. 2021, 23 (2) [žiūrėta 2021-12-10]. Prieiga per: https://www.jmir.org/2021/2/e25499/.

17. KHOKALE R., MATHEW G. S., AHMED S. and MAHEEN S. *Virtual and Augmented Reality in Post-stroke Rehabilitation: A Narrative Review*. 2023, 15 (4) [žiūrėta 2023-02-05]. doi: 10.7759/cureus.37559.
18. HEBERT D., LINDSAY M. P. and TEASELL R. *Canadian stroke best practice recommendations: Stroke rehabilitation practice guidelines*. International Journal of Stroke. 2015, 11 (4) [žiūrėta 2023-02-05]. doi: 10.1177/1747493016643553.
19. KO L., STEVENSON C., CHANG W. and YU K. *Integrated Gait Triggered Mixed Reality and Neurophysiological Monitoring as a Framework for Next-Generation Ambulatory Stroke Rehabilitation*. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2021, 29, 2435–2444 [žiūrėta 2023-02-05]. doi: 10.1109/TNSRE.2021.3125946.
20. MENG J., YAN Z., GU F. and XUE T. *Transcranial direct current stimulation with virtual reality versus virtual reality alone for upper extremity rehabilitation in stroke: A meta-analysis*. Heliyon. 2023, 9 (1) [žiūrėta 2023-02-05]. doi: 10.1016/j.heliyon.2022.e12695.
21. LEONG, S. C., TANG, Y. M., TOH, F. M. *Examining the effectiveness of virtual, augmented, and mixed reality (VAMR) therapy for upper limb recovery and activities of daily living in stroke patients: a systematic review and meta-analysis*. J NeuroEngineering Rehabil. 2022, 19 (93) [žiūrėta 2023-02-05]. doi: 10.1186/s12984-022-01071-x.
22. LUCA R. D., LEONARDI S., MARESCA G. and MARILENA F. C. *Virtual reality as a new tool for the rehabilitation of post-stroke patients with chronic aphasia: an exploratory study*. Aphasiology. 2023, 37 (2), 249–259 [žiūrėta 2023-04-10]. doi: 10.1080/02687038.2021.1998882.
23. AMFT C. S., END K., SUICA Z. and THALER I. *Effect of a four-week virtual reality-based training versus conventional therapy on upper limb motor function after stroke: A multicenter parallel group randomized trial*. 2018 [žiūrėta 2023-04-10]. doi: 10.1371/journal.pone.0204455.
24. ROSS R., HART E., WILLIAMS E. R. and GREGORY C. *Combined Aerobic Exercise and Virtual Reality-Based Upper Extremity Rehabilitation Intervention for Chronic Stroke: Feasibility and Preliminary Effects on Physical Function and Quality of Life*. ACRM. 2023, 5 (1) [žiūrėta 2023-04-15]. doi: 10.1371/journal.pone.0204455.
25. BROWNLEE, J. *Deep Learning Models for Human Activity Recognition*. 2018 [žiūrėta 2021-12-08]. Prieiga per: https://machinelearningmastery.com/deep-learning-models-for-human-activity-recognition/.
26. BROWNLEE, J. *1D Convolutional Neural Network Models for Human Activity Recognition*. 2018 [žiūrėta 2021-12-08]. Prieiga per: https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/.
27. ANGUITA D., GHIO A. and ONET L. *A public domain dataset for human activity recognition using smartphones*. 2013 [žiūrėta 2021-12-09]. Priega per: https://upcommons.upc.edu/handle/2117/20897.
28. ADALOGLOU N. *Understanding the receptive field of deep convolutional networks*. 2020 [žiūrėta 2021-12-09]. Priega per: https://theaisummer.com/receptive-field/.
29. MISHRA M. *Convolutional Neural Networks, Explained. Towards Data Science*. 2020 [žiūrėta 2021-12-09]. Priega per: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.
30. CHORARIA A. *Class Activation Mapping in Deep Learning*. Login Radius. 2023 [žiūrėta 2023-02-02]. Prieiga per: https://www.loginradius.com/blog/engineering/class-activation-mapping/.

31. BROWNLEE, J. *How to Visualize Filters and Feature Maps in Convolutional Neural Networks*. 2019 [žiūrėta 2021-12-08]. Prieiga per: https://machinelearningmastery.com/how-to-visualize-filters-and-feature-maps-in-convolutional-neural-networks/.
32. SHARMA S. *Activation Functions in Neural Networks*. Towards Data Science. 2017 [žiūrėta 2022-11-14]. Prieiga per: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.
33. BRODTMAN Z. *The Importance and Reasoning behind Activation Functions*. Towards Data Science. 2021 [žiūrėta 2022-11-15]. Prieiga per: https://towardsdatascience.com/the-importance-and-reasoning-behind-activation-functions-4dc00e74db41#:~:text=Why%20do%20we%20need%20them,a*x%2Bb.
34. VISHWAKARMA S. *Why is Sigmoid Function Important in Artificial Neural Networks?* 2023 [žiūrėta 2023-03-10]. Prieiga per: https://www.analyticsvidhya.com/blog/2023/01/why-is-sigmoid-function-important-in-artificial-neural-networks/.
35. HE K., ZHANG X., REN S. and SUN J. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. IEEE International Conference on Computer Vision (ICCV). 2015, 1026-1034, doi: 10.1109/ICCV.2015.123.
36. PRAHARSHA V. *ReLU (Rectified Linear Unit) Activation Function*. 2023 [žiūrėta 2023-03-10]. Prieiga per: https://iq.opengenus.org/relu-activation/.
37. KANKAM N. K. *Exponential Linear Unit (ELU)*. 2023 [žiūrėta 2023-03-10]. Prieiga per: https://iq.opengenus.org/exponential-linear-unit/.
38. SHAW S. *Activation Functions Compared With Experiments*. 2022 [žiūrėta 2023-03-11]. Prieiga per: https://wandb.ai/shweta/Activation%20Functions/reports/Activation-Functions-Compared-With-Experiments--VmlldzoxMDQwOTQ#experimenting-with-the-activation-functions.
39. HIMANSHU S. *Activation Functions : Sigmoid, tanh, ReLU, Leaky ReLU, PReLU, ELU, Threshold ReLU and Softmax basics for Neural Networks and Deep Learning*. 2019 [žiūrėta 2023-03-11]. Prieiga per: https://himanshuxd.medium.com/activation-functions-sigmoid-relu-leaky-relu-and-softmax-basics-for-neural-networks-and-deep-8d9c70eed91e.
40. ANTONIADIS P. *Activation Functions: Sigmoid vs Tanh*. 2023 [žiūrėta 2023-03-18]. Prieiga per: https://www.baeldung.com/cs/sigmoid-vs-tanh-functions.
41. PRIYA C. B. *Softmax Activation Function: Everything You Need to Know*. 2023 [žiūrėta 2023-03-18]. Prieiga per: https://www.pinecone.io/learn/softmax-activation/.
42. BROWNLEE J. *A Gentle Introduction to Pooling Layers for Convolutional Neural Networks*. 2019 [žiūrėta 2021-12-08]. Prieiga per: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/.
43. VERMA Y. *A Complete Understanding of Dense Layers in Neural Networks*. Mystery Vault. 2021 [žiūrėta 2023-03-25]. Prieiga per: https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/.
44. POKHREL S. *Beginners Guide to Convolutional Neural Networks*. 2019 [žiūrėta 2021-12-09]. Prieiga per: https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d.
45. BROWNLEE J. *Crash Course in Convolutional Neural Networks for Machine Learning*. 2016 [žiūrėta 2021-12-10]. Prieiga per: https://machinelearningmastery.com/crash-course-convolutional-neural-networks/.

46. MURAD A. and PYUN J. Y. *Deep Recurrent Neural Networks for Human Activity Recognition*. 2017 [žiūrėta 2022-01-10]. Prieiga per: https://www.mdpi.com/1424-8220/17/11/2556.
47. HOCHREITER S. and SCHMIDHUBER J. *Long Short-Term Memory*. Neural Comput. 1997, 9 (8), 1735–1780 [žiūrėta 2022-01-10]. doi: 10.1162/neco.1997.9.8.1735.
48. SINGHAL G. *Introduction to LSTM Units in RNN*. 2020 [žiūrėta 2022-01-10]. Prieiga per: https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn.
49. THAKUR D. *LSTM and its equations*. 2018 [žiūrėta 2022-01-10]. Prieiga per: https://medium.com/@divyanshu132/lstm-and-its-equations-5ee9246d04af.
50. MURAD A. and PYUN J. Y. *Deep Recurrent Neural Networks for Human Activity Recognition*. Sensors. 2017, 17, 2556 [žiūrėta 2022-01-10]. doi: 10.3390/s17112556.
51. BROWNLEE J. *Time Series Forecasting as Supervised Learning*. 2016 [žiūrėta 2021-12-10]. Prieiga per: https://machinelearningmastery.com/time-series-forecasting-supervised-learning/.
52. BANOS O., GALVEZ M. and DAMAS M. *Window Size Impact in Human Activity Recognition*. 2014 [žiūrėta 2022-01-12]. Prieiga per: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029702/
53. LAGUNA J. O., OLAYA A. G. And BORRAJO D. *A Dynamic Sliding Window Approach for Activity Recognition*. 2011 [žiūrėta 2022-01-17]. Prieiga per: https://doi.org/10.1007/978-3-642-22362-4_19.
54. DAWSON H. L., DUBRULE O. and JOHN C. M. *Impact of dataset size and convolutional neural network architecture on transfer learning for carbonate rock classification*. Computers & Geosciences. 2023, 171 [žiūrėta 2023-04-27]. ISSN 0098-3004. doi: 10.1016/j.cageo.2022.105284.
55. PANDIT B. *Four most popular data normalization techniques every data scientist should know*. 2023 [žiūrėta 2023-04-27]. Prieiga per: https://dataaspirant.com/data-normalization-techniques/.
56. CUEMATH. *Standard Deviation*. 2023 [žiūrėta 2023-04-27]. Prieiga per: https://www.cuemath.com/data/standard-deviation/.
57. UIL VR SOLUTIONS BV. *What is virtual reality hand tracking?* 2022 [žiūrėta 2023-04-27]. Prieiga per: https://vr-expert.com/what-is-virtual-reality-hand-tracking/.
58. BUCKINGHAM G. *Hand Tracking for Immersive Virtual Reality: Opportunities and Challenges.* Front. Virtual Real. 2021 [žiūrėta 2023-04-27]. doi: 10.3389/frvir.2021.728461.
59. ULTRALEAP. *Pimax Integrates Ultraleap Hand Tracking in Next Gen VR Headsets*. 2020 [žiūrėta 2023-04-28]. Prieiga per: https://www.ultraleap.com/company/news/press-release/pimax-adds-ultraleap-hand-tracking/.
60. ULTRALEAP. *Leap Motion Controller*. UH-003206-TC Issue 6 Leap Motion Controller Data Sheet. 2013 [žiūrėta 2023-04-29]. Prieiga per: https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf.
61. BURGESS C. *How to Use Hand Tracking in VR and XR Applications – Guest Post by Ultraleap*. 2022 [žiūrėta 2023-01-03]. Prieiga per: https://varjo.com/vr-lab/how-to-use-hand-tracking-in-vr-and-xr-applications/.
62. VARJO DEVELOPER. *Hand Tracking*. 2023 [žiūrėta 2023-01-07]. Prieiga per: https://developer.varjo.com/docs/v3.6.1/get-started/hand-tracking.
63. IVANKOV A. *Types and Methods of Virtual Reality Systems*. 2019 [žiūrėta 2023-01-09]. Prieiga per: https://www.profolus.com/topics/types-and-methods-of-virtual-reality-systems/.
64. NGUYEN W. *Inside Out VS Outside In Tracking*. Vrheaven. 2022 [žiūrėta 2023-01-11]. Prieiga per: https://vrheaven.io/inside-out-vs-outside-in-tracking/.

65. VIVE. *Base Station*. 2023 [žiūrėta 2023-01-18]. Prieiga per: https://www.vive.com/eu/accessory/base-station/.
66. VIVE. *Base Station 2*. 2023 [žiūrėta 2023-01-18]. Prieiga per: https://www.vive.com/eu/accessory/base-station2/.
67. AGARWAL R. *The 5 Classification Evaluation metrics every Data Scientist must know*. Towards Data Science. 2019 [žiūrėta 2023-03-23]. Prieiga per: https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226.
68. CREATELY. *UML Diagram Types Guide: Learn About All Types of UML Diagrams with Examples*. 2021 [žiūrėta 2022-03-20]. Prieiga per: https://creately.com/blog/diagrams/uml-diagram-types-examples/.
69. BLENDER DOCUMENTATION TEAM. *About* [interaktyvus]. 2023 [žiūrėta 2022-12-10]. Prieiga per: https://docs.blender.org/manual/en/latest/getting_started/about/introduction.html.
70. BLENDER DOCUMENTATION TEAM. *Object* [interaktyvus]. 2023 [žiūrėta 2022-12-10]. Prieiga per: https://docs.blender.org/manual/en/latest/scene_layout/object/types.html.
71. BLENDER DOCUMENTATION TEAM. *Modifiers* [interaktyvus]. 2023 [žiūrėta 2022-12-10]. Prieiga per: https://docs.blender.org/manual/en/latest/modeling/modifiers/introduction.html.
72. RUDÉN E. and LOHIKOSKI L. *Optimization of 3D Game Models: A qualitative research study in Unreal Development Kit*. Technologijos mokslai. 2013 [žiūrėta 2022-12-20]. Prieiga per: http://sh.diva-portal.org/smash/record.jsf?pid=diva2%3A708048&dswid=-9769.
73. ALLENE C., PONS J. P. and KERIVEN R. *Seamless image-based texture atlases using multi-band blending*. Technologijos mokslai. 2008. doi: 10.1109/ICPR.2008.4761913.
74. BLENDER DOCUMENTATION TEAM. *Unwrapping* [interaktyvus]. 2023 [žiūrėta 2022-12-20]. Prieiga per: https://docs.blender.org/manual/en/2.79/editors/uv_image/uv/editing/unwrapping/introduction.html.
75. UNREAL ENGINE DOCUMENTATION TEAM. *Setting Up Collisions With Static Meshes* [interaktyvus]. 2023 [žiūrėta 2023-01-05]. Prieiga per: https://docs.unrealengine.com/4.27/en-US/WorkingWithContent/Types/StaticMeshes/HowTo/SettingCollision/.
76. UNREAL ENGINE DOCUMENTATION TEAM. *Developing for Oculus* [interaktyvus]. 2023 [žiūrėta 2023-01-05]. Prieiga per: https://docs.unrealengine.com/5.0/en-US/developing-for-oculus-in-unreal-engine/.
77. ULTRALEAP DOCUMENTATION TEAM. Getting Started [interaktyvus]. 2023 [žiūrėta 2023-01-07]. Prieiga per: https://docs.ultraleap.com/unreal-api/unreal-guide/getting-started.html.
78. UNREAL ENGINE DOCUMENTATION TEAM. *Pawn* [interaktyvus]. 2023 [žiūrėta 2023-01-07]. Prieiga per: https://docs.unrealengine.com/5.1/en-US/pawn-in-unreal-engine/.
79. UNREAL ENGINE DOCUMENTATION TEAM. *Camera* [interaktyvus]. 2023 [žiūrėta 2023-01-08]. Prieiga per: https://docs.unrealengine.com/5.1/en-US/cameras-in-unreal-engine/.
80. COHEN G., AFSHAR S., TAPSON J. and VAN SCHAIK A. *EMNIST: Extending MNIST to handwritten letters*. International Joint Conference on Neural Networks (IJCNN). Anchorage, AK, USA. 2017, 2921-2926 [žiūrėta 2022-11-10]. doi: 10.1109/IJCNN.2017.7966217.
81. GROTHER P. J. *NIST Special Database 19*. 2010 [žiūrėta 2022-11-10]. doi: 10.18434/T4H01C.
82. LECUN Y. and CORTER C. *The MNIST database of handwritten digits*. Computer Science. 1999 [žiūrėta 2022-11-01]. Prieiga per: http://www.pymvpa.org/datadb/mnist.html.

83. KERAS. *EarlyStopping* [interaktyvus]. 2023 [žiūrėta 2023-01-05]. Prieiga per: https://keras.io/api/callbacks/early_stopping/.
84. BROWNLEE J. *Understand the Impact of Learning Rate on Neural Network Performance*. 2019 [žiūrėta 2021-12-10]. Prieiga per: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/.
85. KERAS. *ReduceLROnPlateau* [interaktyvus]. 2023 [žiūrėta 2023-01-05]. Prieiga per: https://keras.io/api/callbacks/reduce_lr_on_plateau/.
86. KERAS. *ModelCheckpoint* [interaktyvus]. 2023 [žiūrėta 2023-01-05]. Prieiga per: https://keras.io/api/callbacks/model_checkpoint/.
87. DOSHI S. *Various Optimization Algorithms For Training Neural Network*. Towards Data Science. 2019 [žiūrėta 2022-06-05]. Prieiga per: https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6.
88. SENARATNE R. *How to Augmentate Data Using Keras*. Towards Data Science. 2020 [žiūrėta 2022-06-05]. Prieiga per: https://towardsdatascience.com/how-to-augmentate-data-using-keras-38d84bd1c80c.
89. HE K., ZHANG X., REN S. and SUN J. *Deep Residual Learning for Image Recognition*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016, 770-778 [žiūrėta 2023-01-02]. doi: 10.1109/CVPR.2016.90.
90. SIMONIAN K. and ZISSERMAN A. *Very deep convolutional networks for large-scale image recognition*. ICLR. 2015 [žiūrėta 2023-01-02]. Prieiga per: https://arxiv.org/pdf/1409.1556.pdf.
91. KRIZHEVSKY A., SUTSKEVER I. and HINTON G. E. *ImageNet Classification with Deep Convolutional Neural Networks*. Communications of the ACM. 2017, 60 (6), 84-90 [žiūrėta 2023-04-20]. doi: 10.1145/3065386.
92. BANGAR S. *LeNet 5 Architecture Explained*. Computer Vision. 2022 [žiūrėta 2023-01-19]. Prieiga per: https://medium.com/@siddheshb008/lenet-5-architecture-explained-3b559cb2d52b.
93. BROOKE J. *SUS: A quick and dirty usability scale*. 1995 [žiūrėta 2023-02-05]. Prieiga per: https://www.researchgate.net/publication/228593520_SUS_A_quick_and_dirty_usability_scale.
94. NATIONAL AERONAUTICS AND SPACE ADMINISTRATION. *NASA TLX task load index.* 2022 [žiūrėta 2023-02-05]. Prieiga per: https://humansystems.arc.nasa.gov/groups/TLX/tlxpaperpencil.php.
95. IGROUP. *Igroup presence questionnaire (IPQ) Database*. 1995 [žiūrėta 2023-02-05]. Prieiga per: http://www.igroup.org/pq/ipq/data.php.

Informacijos šaltinių sąrašas

R1. Shiva Verma. Time series data from an accelorometer. *Towards data science*. 2019 [žiūrėta 2022-01-10]. Preiga per: https://towardsdatascience.com/understanding–1d–and–3d–convolution–neural–network–keras–9d8f76e29610.

R2. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. *Towards data science*. 2019 [žiūrėta 2022-01-12]. Preiga per: https://towardsdatascience.com/convolutional–neural–networks–explained–9cc5188c4939.

R3. Himanshu S. Activation Function for Neural Networks. *Medium*. 2019 [žiūrėta 2022-01-20]. Preiga per: https://himanshuxd.medium.com/activation–functions–sigmoid–relu–leaky–relu–and–softmax–basics–for–neural–networks–and–deep–8d9c70eed91e.

R4. Indoml. Pooling Layer. *Indoml*. 2018 [žiūrėta 2022-01-10]. Prieiga per: https://indoml.com/2018/03/07/student–notes–convolutional–neural–networks–cnn–introduction/.

R5. Gaurav Singhal. Introduction to LSTM Units in RNN. *Plural sight*. [žiūrėta 2022-09-24]. Prieiga per: https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn.

R6. Abdulmajid Murad and Jae–Young Pyun. The proposed HAR architecture. *MDPI*. 2017 [žiūrėta 2022-01-16]. Prieiga per: https://doi.org/10.3390/s17112556

R7. Pradip Samuel. Cross–Validation in Time Series Model. *Medium*. 2020 [žiūrėta 2022-01-15]. Prieiga per: https://medium.com/@pradip.samuel/cross–validation–in–time–series–model–b07fbba65db7

R8. NextGenVR. My Review of the Pimax Hand Tracking Module! *Youtube*. 2021 [žiūrėta 2022-03-20]. Prieiga per: https://www.youtube.com/watch?v=prFuSZr6B5g.

R9. Ultraleap. Leap Motion Controller. *Ultraleap*. 2023 [žiūrėta 2023-01-10]. Prieiga per: https://www.ultraleap.com/product/leap-motion-controller/.

R10. Varjo. Hand tracking offset for XR-3 and VR-3. *Varjo*. 2023 [žiūrėta 2023-01-13]. Prieiga per: https://developer.varjo.com/docs/v3.6.1/get-started/hand-tracking.

R11. Ben Lang. Oculus ‘Designing for Hands’ Document Introduces Best Practices for Quest Hand-tracking. *Road To Vr*. 2020 [žiūrėta 2022-11-04]. Prieiga per: https://www.roadtovr.com/oculus-designing-for-hands-quest-hand-tracking-best-practices/.

R12. Niarax. That Pool | 例のプール. *Sketchfab*. 2020 [žiūrėta 2022-12-11]. Prieiga per: https://sketchfab.com/3d–models/that–pool–c563bd34fda046f6943c7069caf664d9

R13. Pyromma–GS. Monobloc – Plastic Garden Chair. *Sketchfab*. 2020 [žiūrėta 2022-12-11]. Prieiga per: https://sketchfab.com/3d–models/monobloc–plastic–garden–chair–261d81ec28d845eba3bb7373768c60c5

R14. Andreas Mischok. Lago d'Isola. *Poly Haven*. 2020 [žiūrėta 2022-12-11]. Prieiga per: https://polyhaven.com/a/lago_disola

R15. Chris Crawford. EMNIST (Extended MNIST). *Kaggle*. 2020 [žiūrėta 2022-04-05]. Prieiga per: https://www.kaggle.com/datasets/crawford/emnist.

R16. Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre van Schaik. Visual breakdown of the EMNIST datasets. *Arxiv*. 2017 [žiūrėta 2022-04-07]. Prieiga per: https://arxiv.org/pdf/1702.05373.pdf.

R17. Albers Uzila. ResNet-50 architecture. *Towards Data Science*. 2022 [žiūrėta 2023-02-28]. Prieiga per: https://towardsdatascience.com/5-most-well-known-cnn-architectures-visualized-af76f1f0065e.

R18. John Brooke. SUS: A quick and dirty usability scale. *Research Gate*. 1995 [žiūrėta 2023-03-01]. Prieiga per: https://www.researchgate.net/publication/228593520_SUS_A_quick_and_dirty_usability_scale.

R19. Stanton N, Salmon P and Walker G. Mental workload assessment method. *Agency for Healthcare research and Quality*. 2010 [žiūrėta 2023-03-03]. Prieiga per: https://digital.ahrq.gov/health-it-tools-and-resources/evaluation-resources/workflow-assessment-health-it-toolkit/all-workflow-tools/nasa-task-load-index.

R20. Igroup. igroup presence questionnaire (IPQ). *Igroup.* 1995 [žiūrėta 2023-03-05]. Prieiga per: http://www.igroup.org/pq/ipq/index.php.

Priedai

1 priedas. „Influence of Aerial Image Resolution on Vehicle Detection Accuracy“

Influence of Aerial Image Resolution on Vehicle Detection Accuracy

Donata Gliaubičiūtė [1], Rokas Janavičius [1], Aušra Gadeikytė [1] and Lukas Paulauskas [1]

[1] *Kaunas University of Technology, Studentų g. 50, Kaunas, 51368, Lithuania*

Abstract

Nowadays, the engineering application of vehicle detection from aerial images is a challenging task due to the particularity of perspective, the small size of the objects, and the complex background. This research aim is to investigate low-resolution aerial images of vehicles that can be utilized for vehicle detection using machine learning models. The research work was conducted using one-stage deep learning-based object detection algorithms YOLOv5, YOLOv7, and YOLOv8 on the two datasets (COWC and VEDAI) that addressed the task of small vehicle detection. For the training of the models, available pre-trained weights were used as a starting point, and then each model was trained by utilizing transfer learning. The obtained results of the study demonstrated that by reducing the image pixel ratio every 5 cm per pixel from 12.5x12.5 to 27.5x27.5 cm per pixel, the accuracy of the object detection models decreases by an average of 3.51%. When the pixel ratio varies from 30x30 to 32.5x32.5 cm per pixel, the accuracy of the models drops by an average of 2.33% on the COWC dataset and 42.4% on the VEDAI dataset.

Keywords

Vehicle detection, aerial images, convolutional neural networks, pixel ratio, YOLOv5, YOLOv7, YOLOv8

1. Introduction

Object detection involves finding the numerous objects in the images and identifying their locations. Recently object detection has been considered one of the most challenging tasks in computer vision due to the appearance of objects varying greatly depending on various circumstances, such as image capture technologies. One such technology is UAVs (Unmanned Aerial Vehicles) - a critical enabler for a wide range of applications including automated driving, crowd flow counting, topographic exploration, environmental pollution monitoring, etc.

Over the years, a lot of effort has been put into identifying vehicles and other small targets in the images that UAVs collect. According to B. Wang and B. Xu in 2021 [1] the most common difficulties of object detection in aerial images are:

- The particularity of perspective. Since aerial images are typically taken from above, the objects have less texture features [1]. As a result, the targets can be easily mistaken with other objects [2].
- The size of the objects. In aerial images, the objects are quite small (composed of only 15 to 30 pixels). Moreover, Convolutional Neural Networks (CNNs) down-sampling layers minimize the amount of information that each object has. For instance, after four down-sampling layers, a 24x24 pixels object maintains only around one pixel in feature maps, making it challenging to identify small objects from the background [1].
- The complexity of the background. Usually, aerial images might cover an area of several square kilometers. The presence of different backgrounds in this receptive field, such as the countryside, mountains, urban areas, etc., interfere with the object detection process [1].

Proceedings Name, Month XX–XX, YYYY, City, Country
EMAIL: email1@mail.com (A. 1); email2@mail.com (A. 2); email3@mail.com (A. 3)
ORCID: XXXX-XXXX-XXXX-XXXX (A. 1); XXXX-XXXX-XXXX-XXXX (A. 2); XXXX-XXXX-XXXX-XXXX (A. 3)


Deep learning algorithms have enabled vehicle identification technologies to attain very high performance. The deep convolutional neural network may use the dataset to train and enhance its model independently. Deep learning-based object detection algorithms that are frequently utilized may be split into two categories: one-stage and two-stage detectors [3].

Two-stage object detection algorithms Faster R-CNN (Region-based Convolutional Neural Networks) divide the target detection into two stages, that is, first use the Region Proposal Network (RPN) to extract candidate target information, and then use the detection network to complete the location and category of candidate targets [3]. One-stage object detection algorithms such as YOLO (You Only Look Once) do not require to use RPN, and directly generate the location and category information of the target through the network, which is an end-to-end target detection algorithm. Therefore, the single-step target detection algorithms have a faster detection speed [3].

One of the first methods to use convolutional neural networks for object detection and to show off their impressive capabilities is region-based CNN (R-CNN) [4]. In R-CNN, a selective search algorithm selects image regions that could contain target objects, and then the CNN is used to map the target objects in the suggested region. Fast R-CNN used an SPP (Spatial Pyramid Pooling) layer and a RoI pooling layer to increase accuracy and runtime over R-CNN [4]. Unlike the R-CNN, which classifies each region proposal independently, Fast R-CNN computes a feature map from a full image only once and then categorizes region proposals by projecting each one onto that feature map. Moreover, the Fast R-CNN algorithm uses a time-consuming selective search method to look for region suggestions in a target image. In Faster R-CNN [4], the selective search is replaced with RPN, which calculates region proposals from an input image. Faster R-CNN is 900% faster than Fast R-CNN and is made up completely of deep learning networks. Directly connecting the RPNs and the classifier network would help Faster R-CNN to further advance [5]. In 2017, T. Tang et al. designed an improved Faster-RCNN to solve the difficulties of locating the positions of small vehicles and classifying the vehicle from the background [6].

In one pass, YOLO predicts and categorizes bounding boxes of objects. An image is initially divided into non-overlapping grids through YOLO. For each cell in the grids, YOLO fore-casts the likelihood that an object will be present, the coordinates of the anticipated box, and the object's class. Each cell's bounding boxes and their confidence scores are predicted by the network. The network then determines the classes' probabilities for each cell [7]. The first version of YOLO, coined YOLOv1, reportedly achieves a faster inference time, but lower accuracy compared to a single-shot detector [8]. In order to increase the speed and accuracy of detection, YOLOv2 was suggested. Anchor boxes are used in YOLOv2 together with convolutional layers that are not fully connected [8]. The accuracy of the network is further increased by YOLOv2 using batch normalization (BN) and a high-resolution classifier. YOLOv3 [9] uses three detection levels and predicts three box anchors for each cell. To extract feature maps, YOLOv3 adds a deeper backbone network (Darknet-53) to the system. Due to the addition of more layers, the prediction is slower than with YOLOv2. Many technical improvements were made in YOLOv4 while maintaining its computational efficiency. The improvements slightly affected the inference time but significantly increased accuracy [10].

According to A. Ammar et al. in 2021 [2], vehicle detection is possible for different data sets with an accuracy from 85.3% to 98%. However, vehicle detection is still challenging when aerial images are small in size and contain a large number of objects. It might cause information loss when convolution operations are performed [2].

There are different aerial image data sets such as OIRDS [11], PUCPR [13], COWC[14], and VEDAI [15] that might be used for the investigation of vehicle detection. The overhead imagery research data set (OIRDS) project produced a data set with almost 1,000 labeled images suitable for developing automated vehicle detection algorithms [11]. “Overhead imagery research data set” contains approximately 1,800 labeled targets. For each target, there are over 30 annotations and over 60 statistics, that describe the target within the context of the image. Images sizes range from 256×256 pixels to 512×512 pixels. The dataset contains five classes of vehicles (“truck”, “pickup”, “car”, “van” and “unknown”). Annotations give information such as color and distance to the ground [11]. On the other hand, this database is hard to apply to benchmark target detection algorithms because there is no defined evaluation protocol, the dataset is obtained by aggregating multiple sources of images (20 different sources), and does not have sufficient statistical regularity. These issues make the results difficult to reproduce, preventing other researchers from making any comparisons with this dataset [11, 12]. It was

tried to split this database (easy, medium, and hard). However, the precise set of images in each split was not defined, preventing the reproduction of results [12].

Approximately 17,000 photos in the Pontifical Catholic University of Parana Dataset (PUCPR) are devoted to car counting in settings of various parking lots. The dataset includes details about 16,456 vehicles. The aerial images in the collection were taken from a drone view at a height of about 40 meters. The image set is annotated by a bounding box per car. All labeled bounding boxes have been well recorded with the top-left and bottom-right points [13].

Nearly 90,000 automobiles were collected using a drone from 4 distinct parking lots for the Car Parking Lot Dataset (CARPK). This is a large dataset with an emphasis on automobile counting in various parking lots. The bounding box for each car is annotated in the image set. Top-left and bottom-right points have been accurately recorded for each labeled bounding box. It is supporting object counting, object localizing, and further investigations with the annotation format in bounding boxes [13].

The purpose of this study is to investigate the change in the accuracy of object detection models for detecting vehicles in aerial images when the resolution of the images fed to the models is reduced. The findings of this study might be useful when certain situations require quick and real-time decision-making regarding the distribution of vehicles in a geographic space if collecting aerial photographs of this space is available. When it is known what minimum resolution and object detection models are sufficient to obtain acceptable results from aerial images, it is possible to save time for flying UAVs, processing information, and presenting results. The models selected for this study are one of the best-performing one-stage detectors (YOLOv5, YOLOv7, YOLOv8) that reach high accuracy and speed when applied to object detection tasks.

2. Methods

2.1. Data Preprocessing

The research was conducted using object detection algorithms on the COWC and VEDAI datasets that address the task of small vehicle detection. The cars overhead with context (COWC) dataset contains many unique cars (32,716) from six different image sets, each covering a different geographical location and produced by different images [14]. The images cover regions from Toronto (Canada), Selwyn (New Zealand), Potsdam and Vaihingen (Germany), Columbus, and Utah (The United States). The COWC dataset provides data from overhead at 15 cm (about 5.91 in) per pixel resolution at ground (all data is EO) and is designed to be challenging for detection models. Furthermore, it contains 58,247 usable negative targets, many of which have been hand-picked objects similar to cars such as boats, trailers, bushes, and A/C units. To compensate for the additional difficulty, the context was included around targets. Context can help identify something that may not be a car or confirm it is a car. In general, the idea is to allow a deep learner to decide the weight between context and appearance such that something that looks very much like a car is detected even if it is in an unusual place.

The vehicle detection in aerial imagery (VEDAI) database includes various back-grounds such as woods, cities, roads, parking lots, construction sites, or fields. In addition, the vehicles to be detected have different orientations and can be altered by specular spots, occluded, or masked. Each image is available in several spectral bands and resolutions. VEDAI set has 2950 cars in 512x512 and 1024x1024 images. The dataset with 1024x1024 resolution images has a resolution of 12.5cm×12.5cm per pixel. Likewise, 512x512 resolution images have a resolution of 25cm×25cm per pixel. The images were taken during the spring of 2012. Raw images have 4 uncompressed color channels. The dataset has nine different classes of vehicles: “plane”, “boat”, “camping car”, “car”, “pick-up”, “tractor”, “truck”, “van”, and “other”. Two meta-classes are also defined and considered in the experiments. The “small land vehicles” class has the “car”, “pick-up”, “tractor”, and “van” classes included, and the “large land vehicles” class contains the “truck” and “camping car” classes [15].

When preparing the VEDAI dataset for the experiments, these classes have been dropped: “plane”, “boat”, “camping car”, “tractor”, “truck”, and “other”. These changes were done in order to have comparable visual data about the vehicles. Only one class "Car" was left in both datasets. In the COWC

dataset, a class of negative samples was removed. Figure 1. depicts a histogram of different numbers of cars in the VEDAI dataset.

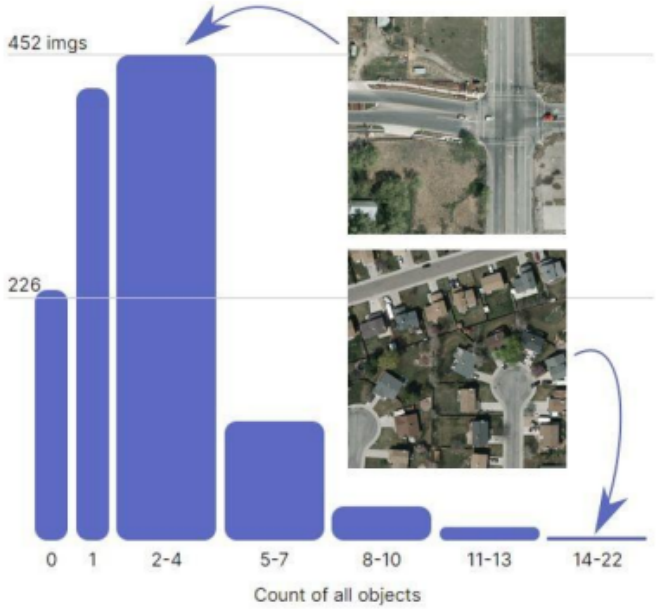


Figure 1: Histogram of "Car" class count by the image in the VEDAI dataset

The Roboflow [16] platform was used to manage the data sets. For each evaluation sample, the dataset images were proportionally reduced in size to achieve the desired centimeters per pixel ratio. With the use of the Roboflow platform, an analysis of the data was performed, resulting in the following findings:

- COWC dataset contains 32810 annotations. The average number of bounding boxes per image is 18.8. There are 800 images in the dataset with null examples.
- VEDAI dataset contains 2807 annotations. The average number of bounding boxes per image is 2.2. There are 233 images in the dataset with null examples.
- Figure 1. demonstrates that the VEDAI dataset has mostly from 2 to 4 vehicles per image and the COWC dataset has from 2 to 49 vehicles per image.

2.2. Object Detection Algorithms

The YOLOv5 adopted the concept of anchor boxes to speed up the R-CNN algorithm and abandoned the use of manually chosen anchor boxes was released in 2020. To get a better prior value, K-means clustering was done on the bounding box dimensions [17].

Introduced in 2022 YOLOv7 surpassed all known object detectors created before in both speed and accuracy in the range from 5 FPS to 160 FPS and had the highest accuracy 56.8% AP among in that time all known real-time object detectors with 30 FPS or higher on GPU V100. YOLOv7 was trained on the MS COCO dataset from scratch without using any other datasets or pre-trained weights. The YOLOv7 model preprocessing method is integrated with YOLOv5, and the use of Mosaic data augmentation is suitable for small object detection [17].

The most recent group of YOLO-based object detection models is called YOLOv8. For detection, segmentation, and classification, there are five models (Nano, Small, Medium, Large, and Xtra Large) in each category of the YOLOv8. The fastest and smallest is YOLOv8 Nano, and the slowest and most accurate is YOLOv8 Extra Large (YOLOv8x). All of the YOLOv8 models had improved throughput when compared to other YOLO models trained at 640 image resolution while using around the same amount of parameters [17]. The effectiveness of object detection on 640 image size between YOLOv8 and YOLOv5 is summarized in Table 1.

Table 1
Object detection performance comparison between YOLOv8 and YOLOv5

Model Size	YOLOv5	YOLOv8	Difference
Nano	28	37.3	+33.21%
Small	37.4	44.9	+20.05%
Medium	45.4	50.2	+10.57%
Large	49	52.9	+7.96%
Xtra Large	50.7	53.9	+6.31%

3. Evaluation metrics

For the evaluation of the performance of the models, the following evaluation metrics [18] were used in this study:

- Total (pre-process + inference + NMS) detection speed in milliseconds;
- Precision;
- Recall;
- Mean average precision (mAP) calculated at Intersection over Union (IoU) [19] threshold 0.5 (mAP@0.5);
- mAP over different IoU thresholds from 0.5 to 0.95 with step 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95);
- F1 score;
- Confusion Matrix.

The precision metric Precision stands for the proportion of positive samples in the samples with positive prediction results (see calculation formula (1)).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1}$$

where TP denotes true positive samples, FP – false positive samples.

The recall represents the prediction result as the proportion of the actual positive samples in the positive samples to the positive samples in the whole sample. The calculation formula (2), where FN stands for false negative samples can be defined as

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

The F1 score is the weighted average of precision and recall, calculated (3) as follows:

$$F1 = \left(\frac{2}{Recall^{-1} + Precision^{-1}}\right) = 2 \cdot \frac{Precision \cdot Recall}{\mathrm{Precision} + \mathrm{Recall}} \tag{3}$$

Precision reflects the model's ability to distinguish negative samples. The higher the precision, the stronger the model's ability to distinguish negative samples. Recall reflects the model's ability to identify positive samples. The higher the recall, the stronger the model's ability to detect positive samples. The F1 score is a combination of the two. The higher the F1 score, the more robust the model.

For object detection tasks, the most common way to determine if a single object proposal is correct is by using the Intersection over Union (IoU) metric [19]. It takes the set A of proposed object pixels and set of true object pixels B and calculates the intersection area. The calculation formula (4) is as follows:

$$IoU(A, B) = \frac{A \cap B}{\mathrm{A} \cup \mathrm{B}} \tag{4}$$

In most cases, an IoU of over 0.5 means that the object was detected, otherwise it was a failure .

$$mAP = \frac{1}{\mathrm{n}} \sum_{k=1}^{k=n} AP_k \; Recall = \frac{TP}{\mathrm{TP} + \mathrm{FN}} \tag{5}$$

The mean of average precision (AP) values are calculated over recall values from 0 to 1. Average precision is calculated as the weighted mean of precisions at each threshold and the weight is the increase in recall from the prior threshold. The mean of average precision is the average AP of each class. The mAP is evaluated (5) by finding each class's average precision (AP) and then averaging over all specified classes.

4. Experiments and results

During the experiments, the processes indicated in Fiure. 2. were carried out, which consisted of obtaining images with their original pixel ratio, resizing images, dividing the data sets into training, validation, and testing sets, running training sets to models, custom training models using pre-trained weights, evaluating models with testing sets.

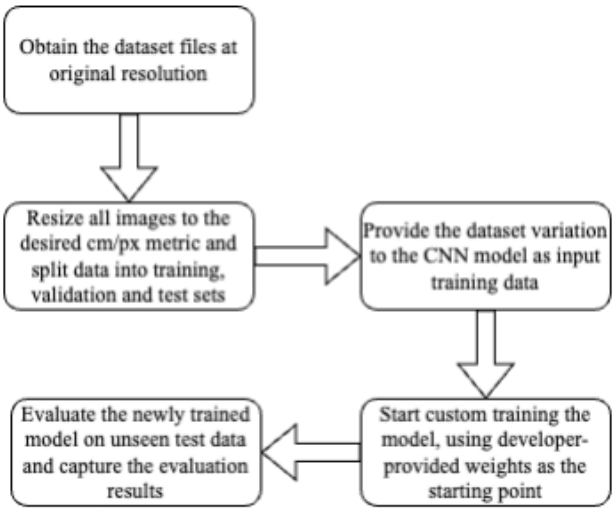


Figure 2: Process flow diagram of experiments

For the training of the models, available pre-trained weights were used as a starting point for each variation of the datasets („yolov5s.pt" for the YOLOv5, „yolov7.pt" for the YOLOv7, and „yolov8s.pt" for YOLOv8). Afterward, each model was trained by utilizing transfer learning for 100 epochs. After training, testing was performed, by providing models with unseen images and capturing their inference time, as well as available precision metrics. Regarding the dataset splits, both datasets featured the same data split: 70% training, 20% validation, and 10% testing. The hardware used for training and testing each variation is provided in the Table 2.

Table 2
Hardware used for the experiments

Hardware	Specification
CPU model	Intel(R) Xeon(R)
CPU frequency	2.30GHz
GPU model	Nvidia T4
GPU VRAM	16GB
GPU Memory Clock	1.59GHz

The following issues were observed during models training:

- YOLOv8 training on COWC at 15x15 cm per pixel variation would display uncommon behavior in several repeated runs, where model training performance drops in the middle of training, and the precision drops instantly.

- YOLOv7 training on VEDAI at 20x20 cm per pixel variation also exhibits inconsistent behavior, as after the first 20 training epochs the training metrics fluctuate and drop, resetting the training progress.

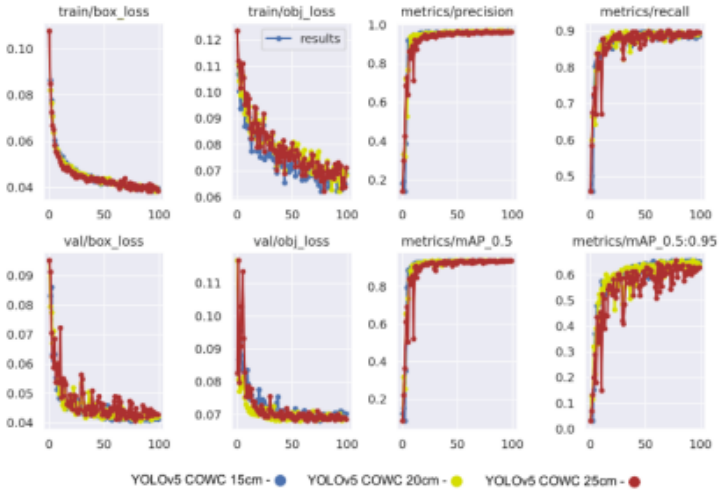


Figure 3: Training YOLOv5 on COWC results

The results of training YOLOv5 on the COWC dataset are provided in Figure 3. The testing results (see Table 3.) demonstrate the performance of YOLOv5, YOLOv7, and YOLOv8 models when trained and tested at various image pixel ratios. Specifically, the models were evaluated at 15, 20, 25, 27.5, 30, and 32.5 cm per image pixel resolution. However, it should be noted that the original pixel ratio of the VEDAI dataset is 12.5 cm per pixel, while the initial pixel ratio of the COWC dataset is 15 cm per pixel, meaning that the original resolution of the datasets did not match in the starting evaluation phase. As a result, the findings of the VEDAI and COWC datasets were not directly comparable during the experiments when the pixel ratio was 12.5 cm per pixel.

It was found that on the VEDAI dataset P, R, and mAP indicators values are smaller than on the COWC dataset (see Table 3). However, the testing speed was higher. When the datasets had the lowest pixel ratio, which is 32.5 cm per pixel, the YOLOv7 model obtained better test results on the COWC dataset than the YOLOv5. However, YOLOv5 showed significantly higher values on the VEDAI dataset than YOLOv7. For the COWC dataset, YOLOv7 achieved mAP score of 0.93, while YOLOv5 achieved 0.889 under the same conditions. For the VEDAI dataset, YOLOv7 achieved mAP score of 0.029, while YOLOv5 achieved 0.248. YOLOv7 had a faster total detection speed across all datasets and their variations. Figure 4. display the change of accuracy metric of all the models changed with the COWC or VEDAI datasets when the image pixel ratio was reduced by 5 or 2.5 cm, while Figure 5. graphs the detection speed results for the same evaluation task.

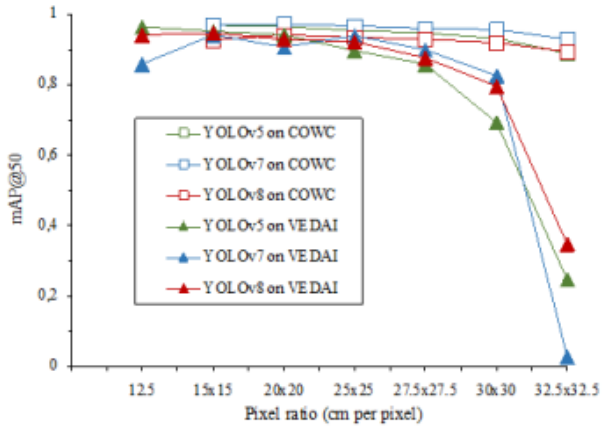


Figure 4: Graph of dependence of mAP@50 indicator and image pixel ratio

Table 3
Testing YOLOv5 YOLOv7 and YOLOv8 models on COWC and VEDAI datasets results

Model	Dataset	Centimeters per pixel	Detection Speed (ms)	P	R	mAP@ 50	mAP@ 50-95
v5	COWC	32.5 x 32.5 cm	10.9	0.87	0.79	0.88	0.45
	VEDAI		9.5	0.41	0.31	0.24	0.07
v7	COWC		9.7	0.93	0.84	0.93	0.56
	VEDAI		5.3	0.11	0.11	0.03	0.01
v8	COWC		6.6	0.89	0.83	0.89	0.53
	VEDAI		5.1	0.42	0.41	0.34	0.11
v5	COWC	30 x 30 cm	14.8	0.91	0.87	0.93	0.55
	VEDAI		9.8	0.71	0.64	0.69	0.31
v7	COWC		12.8	0.94	0.91	0.95	0.61
	VEDAI		8.8	0.83	0.74	0.82	0.35
v8	COWC		7.6	0.93	0.85	0.91	0.60
	VEDAI		5.7	0.72	0.73	0.79	0.39
v5	COWC	27.5 x 27.5 cm	18.2	0.95	0.88	0.94	0.58
	VEDAI		13.9	0.83	0.77	0.86	0.42
v7	COWC		14	0.95	0.90	0.96	0.62
	VEDAI		9.8	0.8	0.86	0.89	0.48
v8	COWC		10.4	0.94	0.87	0.93	0.62
	VEDAI		6	0.87	0.78	0.87	0.49
v5	COWC	25 x 25 cm	23.6	0.96	0.9	0.95	0.61
	VEDAI		15.3	0.88	0.78	0.89	0.44
v7	COWC		18	0.95	0.93	0.96	0.64
	VEDAI		11.7	0.90	0.88	0.94	0.5
v8	COWC		12.5	0.94	0.88	0.93	0.63
	VEDAI		9.2	0.88	0.86	0.92	0.55
v5	COWC	20 x 20 cm	30.2	0.96	0.92	0.96	0.65
	VEDAI		20.9	0.87	0.87	0.94	0.54
v7	COWC		30.5	0.96	0.94	0.97	0.67
	VEDAI		20.6	0.88	0.82	0.90	0.50
v8	COWC		18.1	0.95	0.89	0.94	0.66
	VEDAI		14.4	0.94	0.85	0.93	0.58
v5	COWC	15 x 15 cm	46.3	0.96	0.93	0.97	0.67
	VEDAI		36.8	0.96	0.87	0.95	0.55
v7	COWC		40.5	0.96	0.94	0.97	0.65
	VEDAI		32.6	0.95	0.84	0.94	0.55
v8	COWC		27.3	0.93	0.88	0.92	0.60
	VEDAI		25.9	0.91	0.90	0.94	0.61
v5	VEDAI	12.5 x 12.5 cm	41.9	0.95	0.89	0.96	0.57
v7			38.5	0.81	0.84	0.86	0.47
v8			30.1	0.91	0.90	0.94	0.63

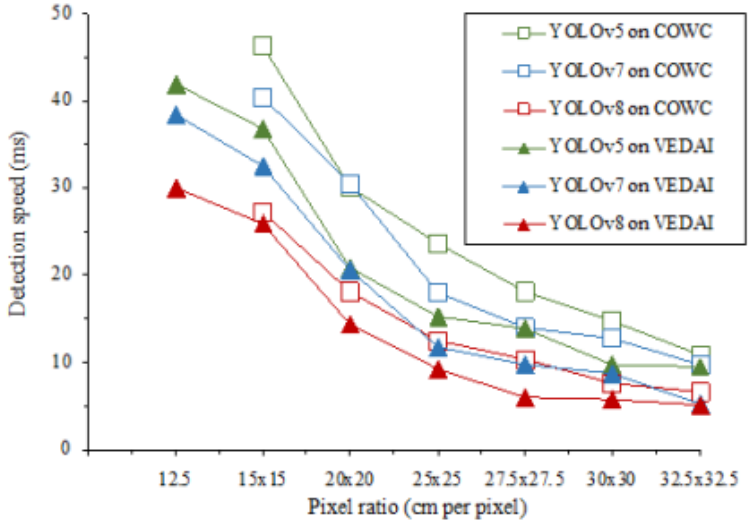


Figure 6: Graph of dependence of detection speed and image pixel ratio

5. Conclusion

In this paper, the change in the accuracy of vehicle detection using reduced pixel ratio aerial images was investigated. When models object detection on aerial images mAP indicator results from 0.86 to 0.97 needed to be reached, the usage of a pixel ratio of 12.5x12.5 to 27.5x27.5 cm per pixel, and YOLOv5, YOLOv7, and YOLOv8 object detection algorithms were proposed. When investigating the dependence of aerial image resolution on the performance of object detection models, it was observed that one-stage object detection algorithms such as YOLOv5, YOLOv7, and YOLOv8 achieve an average of 3.51% lower mAP scores when the image pixel ratio is reduced every 5 cm per pixel from 12.5x12.5 to 27.5x27.5 cm per pixel.

The YOLOv8 model had the most stable results among other models, decreasing by an average of 0.24% when tested on the COWC dataset from 12.5x12.5 to 27.5x27.5 cm per pixel image pixel ratio. The YOLOv5 model achieved an average mAP reduction of 9.6% with images from 12.5x12.5 to 27.5x27.5 cm per pixel image pixel ratio from the COWC dataset, significantly lagging behind the other tested models. All models performed better on the COWC and not on the VEDAI dataset during testing. The VEDAI dataset only had 2807 annotated vehicles, while the COWC dataset contained 32810 annotated vehicles, which may have influenced the testing results. Additionally, vehicles in the VEDAI dataset were labeled, even if some of them were partially hidden by other objects or just partially visible at the edges of the image. When models were tested with the COWC dataset, results dropped by an average of 0.68% and by an average of 6.35% with the VEDAI dataset when using image pixel ratios between 12.5x12.5 and 27.5x27.5 cm per pixel. More pronounced changes in the mean average accuracy of the models are noticeable when the pixel ratio varies from 30x30 to 32.5x32.5 cm per pixel. When the pixel ratio was 30x30 cm per pixel then the accuracy dropped by an average of 1.42% on the COWC dataset and 11.96% on the VEDAI dataset. When the pixel ratio was 32.5x32.5 cm per pixel then the accuracy dropped by an average of 3.23% on the COWC dataset and 72.84% on the VEDAI dataset.

Future research should include other one-stage and two-stage deep learning-based object detection algorithms and experiments with more image-pixel ratio options in order to collect more data on the change in accuracy of the object detection models.

6. Acknowledgements

The analysis was carried out as a group work of the subject P170M113 Applied Research Project in Kaunas University of Technology Masters' programme Artificial Intelligence in Computer Science. We also thank dr. Andrius Kriščiūnas for support, useful suggestions, and mentoring process.

7. References

[1] B. Wang and B. Xu, "A feature fusion deep-projection convolution neural network for vehicle detection in aerial images," PLoS One, vol. 16, no. 5, p. e0250782, May 2021, doi: 10.1371/journal.pone.0250782.

[2] A. Ammar, A. Koubaa, M. Ahmed, A. Saad, and B. Benjdira, "Vehicle Detection from Aerial Images Using Deep Learning: A Comparative Study," Electronics (Basel), vol. 10, no. 7, p. 820, Mar. 2021, doi: 10.3390/electronics10070820.

[3] P. Soviany and R. T. Ionescu, "Optimizing the Trade-Off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction," in 2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Sep. 2018, pp. 209–214. doi: 10.1109/SYNASC.2018.00041.

[4] R. Girshick, "Fast R-CNN," in 2015 IEEE International Conference on Computer Vision (ICCV), Dec. 2015, pp. 1440–1448. doi: 10.1109/ICCV.2015.169.

[5] Y. Koga, H. Miyazaki, and R. Shibasaki, "A CNN-Based Method of Vehicle Detection from Aerial Images Using Hard Example Mining," Remote Sens (Basel), vol. 10, no. 1, p. 124, Jan. 2018, doi: 10.3390/rs10010124.

[6] T. Tang, S. Zhou, Z. Deng, H. Zou, and L. Lei, "Vehicle Detection in Aerial Images Based on Region Convolutional Neural Networks and Hard Negative Example Mining," Sensors, vol. 17, no. 2, p. 336, Feb. 2017, doi: 10.3390/s17020336.

[7] H. V. Koay, J. H. Chuah, C.-O. Chow, Y.-L. Chang, and K. K. Yong, "YOLO-RTUAV: Towards Real-Time Vehicle Detection through Aerial Images with Low-Cost Edge Devices," Remote Sens (Basel), vol. 13, no. 21, p. 4196, Oct. 2021, doi: 10.3390/rs13214196.

[8] W. Liu et al., "SSD: Single Shot MultiBox Detector," 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

[9] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018.

[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020.

[11] F. Tanner et al., "Overhead imagery research data set — an annotated data library & tools to aid in the development of computer vision algorithms," in 2009 IEEE Applied Imagery Pattern Recognition Workshop (AIPR 2009), Oct. 2009, pp. 1–8. doi: 10.1109/AIPR.2009.5466304.

[12] A. Kembhavi, D. Harwood, and L. S. Davis, "Vehicle Detection Using Partial Least Squares," IEEE Trans Pattern Anal Mach Intell, vol. 33, no. 6, pp. 1250–1265, Jun. 2011, doi: 10.1109/TPAMI.2010.182.

[13] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, "Drone-Based Object Counting by Spatially Regularized Regional Proposal Network," in 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 4165–4173. doi: 10.1109/ICCV.2017.446.

[14] T. N. Mundhenk, G. Konjevod, W. A. Sakla, and K. Boakye, "A Large Contextual Dataset for Classification, Detection and Counting of Cars with Deep Learning," 2016, pp. 785–800. doi: 10.1007/978-3-319-46487-9_48.

[15] S. Razakarivony and F. Jurie, "Vehicle detection in aerial imagery : A small target detection benchmark," J Vis Commun Image Represent, vol. 34, pp. 187–203, Jan. 2016, doi: 10.1016/j.jvcir.2015.11.002.

[16] B., N. J. (2022), S. J., et. al. Dwyer, "Roboflow (Version 1.0) [Software]." https://roboflow.com. computer vision., 2022.

[17] S. Rath, "YOLOv8 Ultralytics: State-of-the-Art YOLO Models," https://learnopencv.com/ultralytics-yolov8/, Jan. 10, 2023.

[18] K. Jiang et al., "An Attention Mechanism-Improved YOLOv7 Object Detection Algorithm for Hemp Duck Count Estimation," Agriculture, vol. 12, no. 10, p. 1659, Oct. 2022, doi: 10.3390/agriculture12101659.

[19] H. Rezatofighi, N. Tsoi, J. Y. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," Apr. 2019, arXiv:1902.09630.

2 priedas. „A System for Providing Educational Content in Virtual Reality“

A SYSTEM FOR PROVIDING EDUCATIONAL CONTENT IN VIRTUAL REALITY

Aurimas Gecas, Tomas Blazauskas, Lukas Paulauskas, Andrius Paulauskas, Donata Gliaubiciute

Kaunas University of Technology, Lithuania

Abstract. Virtual Reality (VR) technology allows people to use new ways of learning various subjects. Numerous studies show that VR technology helps students better absorb information and makes the learning process more interesting. But most of the existing applications are not adopted for teachers who have no programming skills to provide educational content in virtual reality. We have developed a system for delivering educational content in virtual reality, which does not require any programming or other special information technology skills. It allows educators to simply prepare virtual environments with learning material and show it to students in virtual reality. In this article, we present that system and its application possibilities.
Keywords: Virtual reality, education, learning.

1. INTRODUCTION

Education is one of the most important areas of a prosperous society. The transfer of knowledge and acquiring practical skills is a priority for learning institutions. Most technologies for learning are designed to make information more accessible. However, faster and more efficient ways of transferring knowledge are increasingly being discussed and sought to provide more interesting and understandable information. In this era of digital devices, there are many methods to improve learning through technology. And one of the steps forward in the development of education is virtual reality (VR), which makes it possible to present educational content much more attractive. This technology is a new high-tech in recent years, which simulates a three-dimensional space through computer and VR devices (Liu et al., 2019).

Virtual reality can be used to improve student learning and engagement in the analyzed field. Such learning is based on creating a virtual environment according to the real or imagined examples of places (Elmqaddem, 2019). These environments allow not only to see the places but also to interact with them. That helps the user to become even more immersed in the virtual space. Immersiveness in the research area motivates the user to go deeper and fully understand the information.

Also, one of the most important advantages of virtual reality is the simulation of various scenarios and complex situations (Lorenz et al., 2016). It is

possible to recreate medical procedures, needed environments, or other situations, which can be dangerous or cost too much in reality. This feature allows users to perform tasks repeatedly and avoid risks while improving their skills.

This work aims to present a system for providing educational content in virtual reality. The system allows users to prepare virtual environments by adding educational content. These environments and learning material can be viewed in virtual reality.

2. RELATED WORKS

The main use of virtual reality technology has been envisaged in the gaming and entertainment business. However, today this technology is not limited to the entertainment industry. In the last ten years, virtual reality technologies have greatly improved, and at the same time, the scope of this technology has expanded. Recently, virtual reality has become particularly popular in education and is applied in various training in order to acquire both theoretical and physical skills (Concannon et al., 2019).

The realization of virtual reality can vary greatly in format and complexity, from a technological point of view, depending on the situation. In some cases, the software is adapted to existing virtual reality hardware, and sometimes standard external devices can be used with the program. When trying to adapt virtual reality to a very specific area, you may need to create new or modify existing, application-specific hardware (Wood et al., 2020).

During the Covid-19 global pandemic, hospitals faced a shortage of experienced medical personnel. To solve this problem, the start-up Virti has created a virtual reality program to prepare medical staff to work with patients with Covid-19 infection. This technology allows doctors and nurses to learn to recognize the symptoms of the disease and treat patients correctly without any risk. It helps to improve the decision-making process for practitioners in emergency situations. This VR software includes various scenarios, from how to put on protective gear efficiently and safely to more intensive cases with sick patients (Kelly, 2020).

Kaminska et al. (2017) explored an interactive VR training strategy for mechanical and electrical engineering education. The prototype of the program allows the user to analyze and study a 3D object by manipulating it in virtual reality. Results of initial studies indicate that the use of VR can be useful for improving the efficiency and quality of higher education. Also, it can enrich competencies, qualifications, and the skills of graduates.

In their research, Horvat et al. (2022) analyzed the potential of immersive virtual reality technology for design education. Forty students of different

expertise levels participated in an experiment and studied a design representation using immersive virtual reality or non-immersive virtual reality visualization technologies. The results show that visualization technology did not have an overall effect on understanding, but immersive virtual reality helped students with lower knowledge to understand specific aspects of a design better.

Chan et al. (2021) explored the use of immersive virtual reality for teaching architectural history. The Pantheon in Rome was chosen as a representative test case for assessing the effectiveness of virtual reality for remote teaching. During the research, two experiments with 57 and 68 students were separately performed, and learning about architecture, history, structural realism, sense of presence in VR were compared to in-class learning. Studies revealed that educational content about the architectural building provided in immersive virtual reality (IVR) positively affected the learning experience and information absorption. Also, the use of IVR allows students to more accurately gauge, recognize, and assess the three-dimensional aspects, size, and proportion of architectural buildings.

3. SITUATION OF STEM EDUCATION UNDER INVESTIGATION

We developed a system for providing educational content in virtual reality using the Unity game engine. The platform consists of a tool for setting up virtual environments and a tool for viewing the content in virtual reality. We use Google Firebase services for authentication, data, and file storage.

The system configuration is presented in figure 1.

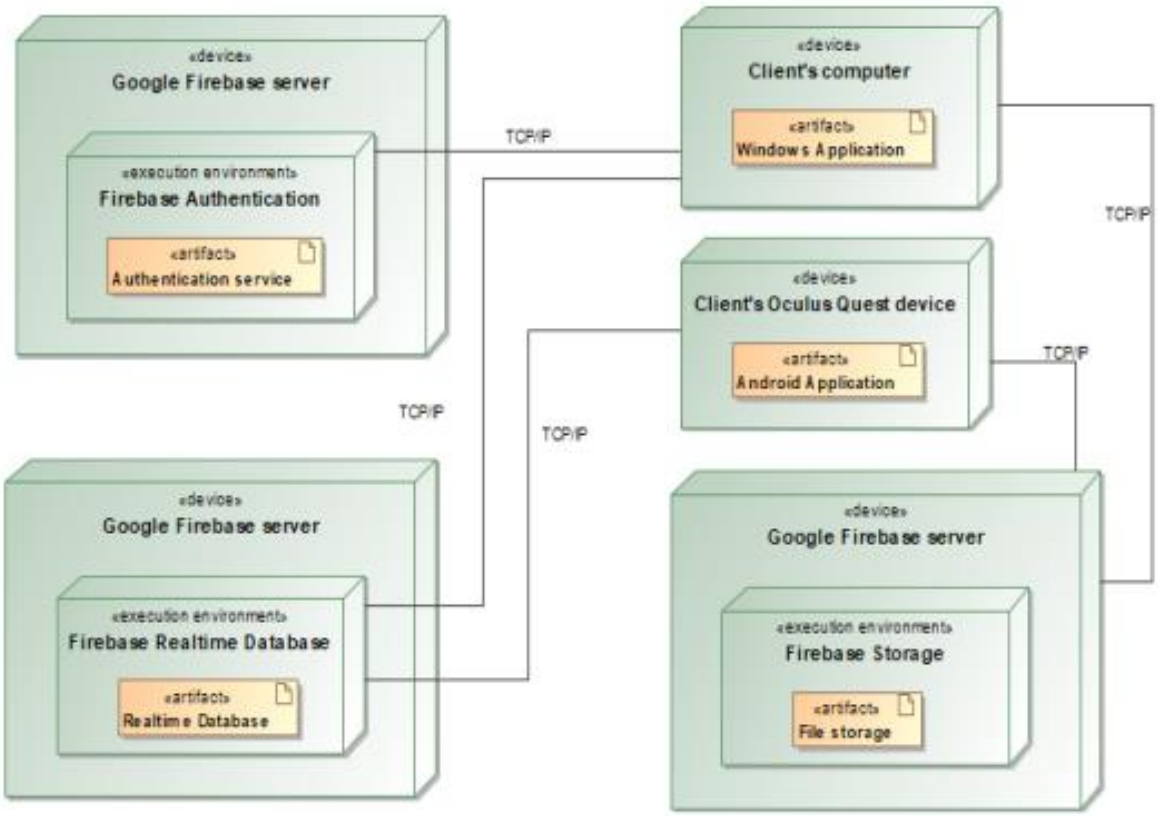


Figure 1. System for presenting educational content in virtual reality

The teacher uses a computer with Windows operating system (see Client's computer in the top right corner of figure 1) to set up virtual environments for the learning process. After successful authentication, which is performed by using Google Firebase Authentication services (see Firebase Authentication in the top left corner of figure 1), the teacher accesses the VR project management interface. It allows the lecturer to create new or edit his own existing projects. In the VR environment editor, the teacher can create a 3D model presentation by importing three-dimensional objects from the computer to the project and setting the order in which the models will be displayed. Each model has transformation settings that allow the user to change object position, rotation, and scale in a 3D space. A slide show can be created in the same way. The teacher has two options to create a slide: upload a picture or a video from a computer. Figure 2 shows the VR project environment editor interface.

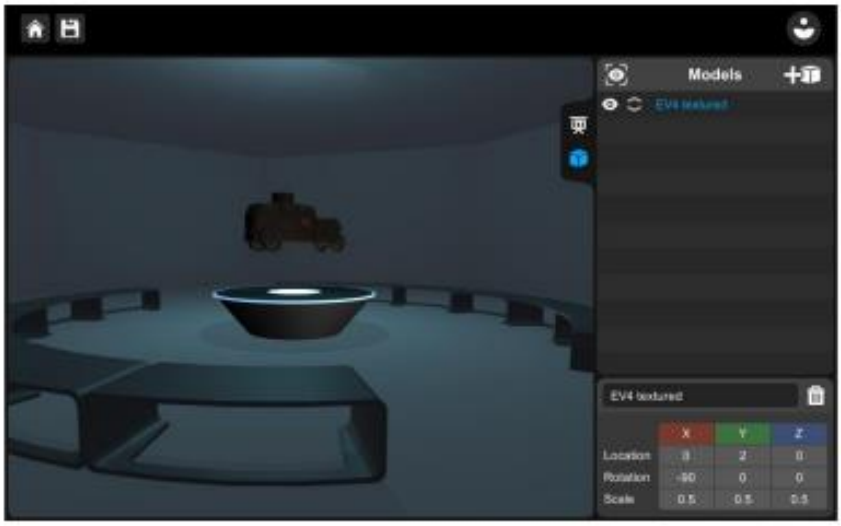


Figure 2. VR environment setup interface

When the VR environment setup is finished, the teacher must save his project. All the data is stored in Google Firebase Realtime Database, and files are held in Firebase Storage.

After the learner opens a project using content viewing in VR tool, all project data is downloaded to the student device for displaying the prepared virtual environment. The educational content viewer is designed for Oculus VR devices that allow 6 DOF (Degrees of Freedom). It means that these devices track movement and rotation in three-dimensional space on all axes when the learner explores the VR environment and educational content. Figure 3 shows the view of the user inspecting the environment in virtual reality.

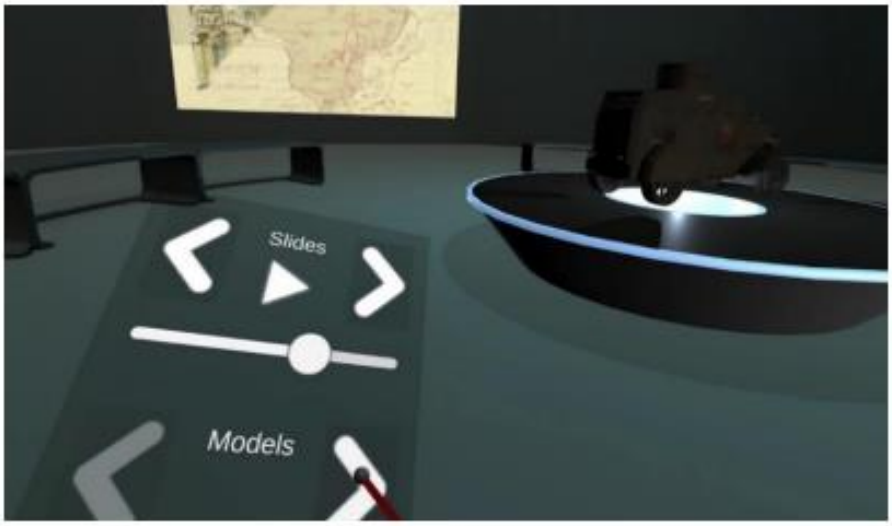


Figure 3. The user explores learning material in VR

A student is placed into a predefined virtual space. In a virtual environment, the user can move physically as far as his physical environment allows or by using the teleportation function, which immediately moves the student to another location.

Within the VR environment, a learner uses a presentation management menu which is mapped to the left virtual hand. During the experience in VR, the student switches between slides and 3D models by clicking white arrows in this menu (see figure 3).

4. CONCLUSION

Virtual reality is being increasingly used across various professional and educational fields. Numerous studies show that VR technology positively impacts education compared to traditional learning methodologies. Despite the growing popularity of VR, most applications are not adopted for providing educational content in virtual reality and are not suitable for people who do not specialize in the information technology field. Some of these systems allow users to predefine virtual environments, but in most cases, they require programming skills or have numerous limitations, especially for 3D model incorporation. Our developed VR educational system allows educators to prepare virtual environments by adding pictures, videos, and 3D models to them and providing this content in virtual reality.

REFERENCES

1. Liu, Y., Fan, X., Zhou, X., Liu, M., Wang, J., & Liu, T. (2019). Application of virtual reality technology in distance higher education. Proceedings of the 2019 4th International Conference on Distance Education and Learning. https://dl.acm.org/doi/10.1145/3338147.3338174.
2. Elmqaddem, N. (2019). View of augmented reality and virtual reality in education. Myth or reality? Online-Journals.Org. Retrieved February 11, 2022, from https://online-journals.org/index.php/i-jet/article/view/9289/5456.
3. Lorenz, D., Armbruster, W., Vogelgesang, C., Hoffmann, H., Pattar, A., Schmidt, D., Volk, T., & Kubulus, D. (2016). Eine neue Ära der MANV-Ausbildung?: InSitu - Realitätsnahes Üben in virtuellen Umgebungen. Der Anaesthesist, 65(9), 703–709. https://doi.org/10.1007/s00101-016-0196-x.

4. Concannon, B. J., Esmail, S., & Roberts, M. R. (2019). Head-mounted display virtual reality in post-secondary education and skill training. Frontiers in Education, 4.
5. Wood, G., Wright, D. J., Harris, D., Pal, A., Franklin, Z. C., & Vine, S. J. (2020). Testing the construct validity of a soccer-specific virtual reality simulator using novice, academy, and professional soccer players, Virtual Reality.
6. Kelly, S. M. (2020). Doctors and nurses are using VR to learn skills to treat coronavirus patients.Cnn.Com. Retrieved February 11, 2022, from https://edition.cnn.com/2020/04/21/tech/vr-training-coronavirus/index.html.
7. Kamińska, D., Sapiński, T., Aitken, N., Rocca, A. D., Barańska, M., & Wietsma, R. (2017). Virtual reality as a new trend in mechanical and electrical engineering education. Open Physics, 15(1), 936–941. https://doi.org/10.1515/phys-2017-0114.
8. Horvat, N., Martinec, T., Lukačević, F., Perišić, M. M., & Škec, S. (2022). The potential of immersive virtual reality for representations in design education. Virtual Reality. https://doi.org/10.1007/s10055-022-00630-w.
9. Chan, C.-S., Bogdanovic, J., & Kalivarapu, V. (2021). Applying immersive virtual reality for remote teaching architectural history. Education and Information Technologies. https://doi.org/10.1007/s10639-021-10786-8.