Tartu Ülikool

Arvutiteaduse instituut

Informaatika õppekava

Rasmus Made

Reaalajas objekti valimine, jälgimine ja asendamine taustaga

Bakalaureusetöö (9 EAP)

Juhendaja: Ardi Tampuu, PhD

Tartu 2025

Reaalajas objekti valimine, jälgimine ja asendamine taustaga

Lühikokkuvõte:

Sügavõppel põhinevate mudelite arenguga on piltide ja videote segmentimine teinud arenguhüppe. Käesoleva bakalaureusetöö eesmärgiks oli koostada reaalaja videovoo segmentimisdemo, mis tuvastaks pildilt inimese ning asendaks ta taustapikslitega. Töö käigus katsetati erinevaid kaasaegseid sügavõppel põhinevaid segmentimismudeleid ning lõplikusse demosse valiti nendest sobivaimad. Valminud demo eesmärgiks oli Delta külastajatele kaasaegsete segmentimismudelite võimekuse tutvustamine. Kasutajate tagasiside põhjal koostatud analüüsis selgus, et demo edukamaks toimiseks on vaja suure riistvaralise võimekusega arvutit.

Võtmesõnad:

Segmentimine, video- ja pilditöötlus, närvivõrgud

CERC: P175 Informaatika, süsteemiteooria, P176 Tehisintellekt

Real-Time Object Selection, Tracking, and Replacement with Background

Abstract:

With the development of deep learning-based models, image and video segmentation has made a significant leap forward. The aim of this bachelor's thesis was to create a real-time video stream segmentation demo that detects a person in the image and replaces them with background pixels. During the work, various modern deep learning-based segmentation models were tested, and the most suitable ones were selected for the final demo. The purpose of the completed demo was to introduce the capabilities of modern segmentation models to Delta visitors. An analysis based on user feedback revealed that for the demo to function more effectively, a computer with high hardware capacity is required.

Keywords:

Segmentation, video and image processing, neural networks

CERC: P175 Informatics, systems theory, P176 Artificial Intelligence

Sisukord

Sissejuhatus

Segmentimine leiab lisaks pildi- ja videotöötlusele rakendust näiteks robootika- ja meditsiini-valdkondades. Sügavõppel põhinevate mudelite arenguga on segmentimismudelite efektiivsus ja kvaliteet palju tõusnud ning sellega ka huvi nende vastu kasvanud. Käeosoleva bakalaureusetöö eesmärgiks on koostada segmentimisdemo, mille eesmärgiks on Delta keskuse töötajatele, tudengitele ja külalistele demonstreerida kaasaegsete segmentimismudelite võimekust. Lisaks segmentimisele demonstreeritakse demos ka piksliasendust (ingl *inpainting*).

Internetis leiduvad segmentimise ja piksliasenduse demod on valdavas osas keskendunud videofailide segmentimisele ning vähesed neist kasutavad sisendina reaalaja videovoogu. Käesoleva töö käigus valminud kood asub avalikult kättesaadavas repositooriumis, mis võimaldab näiteks Tartu Ülikooli arvutiteaduse instituudi tudengitel seda oma isiklike- või kooliprojektide käigus kasutada.

Töö teoreetilises osas annan ülevaate segmentimise ning pildiosade taustaga asendamise tehnoloogiate põhiolemusest, kirjeldan nii traditsioonilisi- kui kaasaegseid lähenemisi. Seejärel annan ülevaate sarnasel teemal valminud projektidest, mida õnnestus internetist leida, ning selgitan ka segmentimise olulisust erinevates praktilistes rakenduses. Metoodikat kirjeldavas peatükis annan ülevaate töö praktilises osas kasutatud tarkvarast, sealhulgas sügavõppe mudelitest, riistvarast ning sellest, kuidas töö kulges. Tulemuste peatükis kirjeldan lõpliku demo arhitektuuri ning analüüsin demo kasutajate tagasisidet.

Töö praktilises osas kasutasin kohati tehisintellekti [1], et mudelite dokumentatsiooni paremini mõista ning enda kirjutatud koodi optimeerida. Tehisintellekti kasutati ka töö lühikokkuvõtte inglise keelde tõlkimisel.

Töö lisades on valminud demo koodi sisaldava avaliku repositooriumi link.

Terminid

Null-sammuga õpe (ingl *zero-shot learning*) - Masinõppe stsenaarium, kus tehisintellekti mudel on võimeline ära tundma ja kategoriseerima objekte, mis tema treeningandmetes ei sisaldu.[1]

Järgnevad kaks mõistet on selgitatud IBM-i kodulehe[2] põhjal:

Tehisnärvivõrk - Masinõppe mudeliarhitektuur, mis moodustab sügavõppe südamiku. Närvivõrkudel on omavahel ühendatud kihid: sisendkiht, varjatud kihid ja väljundkiht. Närvivõrkude kihid koosnevad sõlmedest ehk neuronitest ning sõlmede vahel on kaalude ning lävenditega ühendused, mis määravad selle, millised sõlmed parajasti aktiveeruvad ning milliseid andmeid nad järgmisele kihile saadavad.

Konvolutsiooniline närvivõrk - Närvivõrkude alamhulk, mida kasutatakse enamasti klassifitseerimis- ning masinnägemise ülesannetes. Kui sisendiks on pilt, siis töötavad kihid põhimõttel, et varasemad kihid tegelevad pildi üldisemate osadega nagu värvid ning servad ja iga järgnev kiht tegeleb pildi täpsemate ja abstraktsemate osadega. Konvolutsioonilised närvivõrgud on enamasti arvutuslikult nõudlikud, mistõttu on nende treenimiseks ja kasutamiseks vaja GPU-d.

Transformer - Närvivõrkude arhitektuur, mis põhineb sisemise tähelepanu mehhanismil.[3] Transformer võtab kõiki sisendandmeid korraga arvesse ning suudab nende vahel luua kontekstuaalsed seosed.[3] Võrreldes konvolutsiooniliste närvivõrkudega suudavad pildi erinevaid osi täpsemini suuremasse konteksti seada.[3] Nende treenimiseks on vaja tunduvalt rohkem andmeid ning nad on arvutuslikult nõudlikud.[3]

---

[1] https://www.ibm.com/think/topics/zero-shot-learning

[2] https://www.ibm.com/think/topics/convolutional-neural-networks

[3] https://www.ultralytics.com/glossary/transformer

1. Taust

1.1 Pildi segmentimine üldiselt

Pildi segmentimine on masinnägemise alamülesanne, kus pilt jagatakse erinevateks piksligruppideks, enamasti objektide tuvastamise eesmärgil[4]. Pildi segmentimine hõlmab endas pikslitasandil klassifitseerimist, kus iga piksel jagatakse objekti, tunnust või regiooni kujutavasse klassi[4].

IBM-i kodulehe[4] andmetel jaguneb piltide segmentimine suures plaanis kolme ossa - semantiline, objektipõhine ja panoptiline segmentimine.

- **Semantiline segmentimine** - Kõige lihtsamat tüüpi segmentimine, igale pikslile seatakse vastavusse klass, aga klassi liiki ega erinevaid objekte selles klassis ei täpsustata. Näiteks kui pildil on tänav, kus on mitu autot, siis jaotatakse neid kujutavad pikslid lihtsalt samasse klassi ilma erinevaid autosid eristamata.
- **Objektipõhine segmentimine (**ingl *instance segmentation***) -** Samasse klassi kuuluvad pikslid jaotatakse objektideks.
- **Panoptiline segmentimine** - Ühendab semantilise ja objektipõhise segmentimise, iga piksel jaotatakse semantilisse klassi ning samuti kombineeritakse objektideks.

1.2 Traditsioonilised pildi segmentimise tehnikad

Järgnev lõik tugineb Wang et al artiklile [2].

Enne sügavõppel põhinevate segmentimislahenduste kasutuselevõttu tuginesid traditsioonlilised tehnikad käsitsi loodud reeglitele ja tunnuste eraldamise tehnikatele, et piltidel mustreid tuvastada. Mediumi autori Venkatkumar artikli [3] andmetel on kõige tuntumad traditsioonilised segmenitimistehnikad lävendamine, servade tuvastamine, regioonipõhine tuvastamine, klasterdamine ja veelahkmemeetod. Nimetatud tehnikad keskenduvad pildi madalama taseme tunnustele nagu servad, kujundid ja gradient. Traditsioonilised lähenemised on arvutuslikult vähenõudlikud, kuid nende miinuseks on kehv üldistusvõime ning nende võimetus mõista ebamäärase või kompleksse sisuga stseene.

---

[4] https://www.ibm.com/topics/image-segmentation

1.3 Kaasaegsed sügavõppel põhinevad objektituvastus- ja segmentimis-tehnoloogiad

Järgnev alapeatükk tugineb peamiselt Wang et al. artiklile [2].

Kui traditsiooniliste segmentimistehnikate kontekstis saab rääkida pigem piltide madala taseme tunnuste põhjal töötlemisest, siis sügavõppel põhinevate lähenemiste puhul saab rääkida pildi mõistmisest.

1.3.1 Sügavõppel põhinevate lahenduste areng semantilises segmentimises

Long et al. kasutasid 2015. aastal semantlise segmentimise ülesande lahendamiseks täielikku konvolutsioonilist närvivõrku ehk FCN-i (ingl *Fully Convolutional Network*)[4]. Tavalisel konvolutsioonilisel närvivõrgul asendati täissidusad kihid konvolutsiooniliste kihtidega, et võimaldada piksli täpsusega ennustamine terve pildi ulatuses. Samal aastal avaldatud U-Net arhitektuur [5] parandas FCN lahedust sellega, et lisas närvivõrgu erinevate kihtide vahele lisaühendusi, mis võimaldasid madalama taseme kihtides leitud peened detailid hilisematele kihtidele säilitada. Google'i 2017 aastal loodud DeepLab mudelid [6] parandasid segmentimiskvaliteeti sellega, et kasutasid laiendatud konvolutsiooni (ingl *atrous convolution*), mis võimaldas objekti laiemat konteksti pildil paremini mõista. 2021. Aastal loodud SegFormer mudel on üks esimesi semantiliseks segmentimiseks mõeldud transformeritel põhinev kergekaaluline närvivõrk [7].

1.3.2 Sügavõppel põhinevate lahenduste areng objektipõhises segmentimises

Sügavõppel põhinevad objektipõhised segmentijad jaotuvad ühe- ja kahesammulisteks.[4] Kahesammulised segmentijad tegelevad esimesel sammul objektituvastusega ning teisel sammul pikslite täpsema klassifitseerimisega.[4] Ühesammulised objektipõhised segmentijad teevad seda järjest ning on kahesammulistest segmenitjatest kiiremad, aga ebatäpsemad.[4]

Varased objektipõhise segmentimisega tegelevad mudelid, nagu SDS (2014) [8], DeepMask (2015) [9], SharpMask (2016) [10] ja Mask R-CNN (2017) [11] on kahesammulised segmentijad ja nende tuumaks on regioonipõhine konvolutsiooniline närvivõrk ehk R-CNN (ingl *Region Based Convolutional Neural Network*). R-CNN-ides on võtmerollis huvipakkuvaid alasid tuvastav (ingl *region proposal)* osa, kus määratakse pikslipiirkonnad, mida hilisemad kihid analüüsivad.

Ühesammulised objektipõhised segmentijad, nagu FCOS (2019) [12], YOLACT (2019) [13], SOLO (2020) [14] jätavad eksplitsiitse huvipakkuvate alate tuvastamise vahele ning määravad iga objekti maskid ühe sammuna piksli- või ruumipõhist klassifitseerimist kasutades. Guo et al. 2021 aasta SOTR mudel [15] kombineertib transformerid ja konvolutsioonilise närvivõrgu kaheharulisse süsteemi: üks haru ennustab transformeri abil objektide kategooriad ning teine haru genereerib CNN-i abil segmentatsioonimaskid.

1.3.4 Demos kasutatud mudelid

Mediapipe SelfieSegmenter[5] (2021) - Semantiline segmentija, mis klassifitseerib sisendiks saadava kaadri pikslid kaheks - inimene ja taust.[5] Selle arhitektuuri pole selle loojad avalikult arutanud, aga kogukonna aruteludest selgub, et tegu on U-Neti arhitektuuril põhineva mudeliga.[6] Sisendina võtab mudel kas pildi, video või reaalaja videovoo.[5]

SAM 2.1[7] (2024) - Transformerite arhitektuuril põhinev *zero-shot* objektipõhine segmentija, mis segmendib objekti kas kasutaja sisestatud ruumilise viipe (ingl *prompt*) põhjal või jaotab sisendi kõikideks objektideks, mis ta pildilt leiab.[7] Ruumiliseks viipeks sobib punkt koordinaatidena või suurem huvipiirkond koordinaatidena.[7] Sisendiks sobib kas pilt või video.[7] SAM 2.1 mudeli treenimisel on kasutatud selleks eraldi loodud SA-V andmestikku.[8]

1.4 Segmentimise rakendused

Semanitline pildi segmentimine (ingl *semantic image segmentation*) on pildi üldise sisu ja konteksti mõistmise üks põhilisemaid elemente [16]. See annab võimaluse mõista pildil olevate objektide olemust ja seda, kuidas nad üksteise suhtes paiknevad [16].

Matthew-Mcmulleni Mediumis ilmuinud artikli [17] andmetel leiab segmentimine rakendust näiteks järgnevates valdkondades:

- **Isejuhtivad sõidukid** - Isejuhtivad sõidukid peavad end ümbritseva keskkonnaga mõistlikuks reageerimiseks klassifitseerima end ümbritsevaid objekte, näiteks jalakäijaid, valgusfoore, liiklusmärke, tänavasilte, ehitisi ja teisi sõidukeid.
- **Meditsiin** - Meditsiinis on laias kasutuses tomograafia, mille tulemuseks saadavate piltide segmentimine aitab arste patsientide diagnoosimisel.

[5] https://ai.google.dev/edge/mediapipe/solutions/vision/image_segmenter
[6] https://community.dfrobot.com/makelog-314259.html?
[7] https://ai.meta.com/sam2/
[8] https://docs.ultralytics.com/models/sam-2/#core-components

- **Põllumajandus** - Põldudest tehtud aerofotode segmentimine aitab farmeritel tuvastada oma põldudel märke levivatest viljahaigustest, segmentides näiteks ebanormaalse välimusega viljapead normaalse välimusega viljapeadest.
- **Moetööstus** - Veebipoodides on kasutusel virtuaalsed proovikabiinid, kus kasutaja saab ostetavaid riideid, meiki või muid akessuaare endast tehtud pildil proovida. Inimene segmenditakse taustast, et tema kehale valitud tooted virtuaalselt projitseerida.

Rohit Kundu kirjutab oma videosegmentimise teemalises artiklis [18], et videosegmentimine leiab kasutust:

- **Videotöötluses** - Monteerijate töö lihtsustamine, segmentimine võimaldab neil olemasolevat materjali kiiremini uuteks videoteks kokku lõigata.
- **Turvasüsteemides** - Turvakaamerate videovoo segmentimine võimaldab turvasüsteemil automaatselt inimesi või objekte tuvastada.
- **Spordis** - Videotes mängijate segmentimine aitab treeneritel oma tööd paremini teha ja võistkonna strateegiaid paremini analüüsida.

Veel leiab segmentimine kasutust videokõnedes, kus kasutajad saavad kõne ajal oma tausta vahetada, samas kui inimest kujutavad pikslid jäävad samaks. Samuti kasutatakse segmentimist erinevates sotsiaalmeedia rakendustes, kus kasutajad endale videopildis filtreid seavad.

3. Varasemad tööd

Tao Yu, Runseng Feng jt 2023. aastal avaldatud töö [19] kombineerib omavahel piltide segmentimise ning täitmise. Nende projekt sisaldab endas pildilt, videolt või 3D pildilt objekti eemaldamist ning tausta või tehisintellekti poolt genereeritud objektiga asendamist. Käesoleva töö teemaga on kõige enam seotud nende *video-inpainting* osa. Kasutaja sisestab video, valitud punkti koordinaadid ja video kaadrisageduse. Mudelitest kasutavad nad Meta SAM mudelit[9] objektide segmentimiseks igal kaadril, objekti jälgimiseks läbi kaadrite kasutavad nad OSTrack mudelit[10] ning inpaintimiseks STTN mudelit[11]. Nende töö videoga seotud sektsiooni eristab käesolevast tööst asjaolu, et nemad ei teostanud segmentimist ning kaadrist eemaldatud alade taastäitmist reaalajas.

Software Mansioni 2024. aasta projektis [20] võtsid autorid eesmärgiks koostada töövoo, mis suudaks sooritada pildi täitmist reaalajas ilma hilinemiseta ning 20 kaadrisageduse suuruse läbilaskevõimega. Nende lahenduse jaoks olid kaasaegsed difusioonil põhinevad pildi täitmise mudelid liiga arvutusnõudlikud. Selleks disainisid nad enda mudeli, mis põhineb Rui Liu et al. loodud kergekaalulisel *spatial-temporal decouple* mehhanismil [21]. Arvutuste vähendamiseks kasutasid nad SWIN transformereid [22] ning lisasid tingimusliku positsioonilise kodeeringu, mis võimaldas mudeli sisendiks oleva video resolutsiooni dünaamiliseks muuta. Treeningandmeteks kasutasid nad Youtube-VOS[12] andmestikku. Reaalajas toimimiseks rekonstrueerib nende mudel kaadreid partiide kaupa ning kasutab protsessi kiirendamiseks torch.compile teeki[13], mis kiirendab protsessi 20% võrra. Kuna nende lahenduse eesmärk oli reaalajas pildi täitmist sooritada läbi veebi, siis pidid nad töödeldava video resolutsiooni langetama 480 x 288 piksli peale. Erinevus käeolevast tööst on lisaks madalale resolutsioonile ja madalale kaadrisagedusele (20 kaadrit sekundis) see, et nende lahenduses kasutatud segmentimismudel on demo lihtsuse eesmärgil treenitud

[9] https://segment-anything.com/

[10] https://github.com/botaoye/OSTrack

[11] https://github.com/researchmm/STTN

[12] https://youtube-vos.org/

[13] https://pytorch.org/tutorials/intermediate/torch_compile_tutorial.html

segmentima vaid mudeleid ning teiste objektide segmentimiseks tuleks see ümber treenida. Mudelid, mida projektis kasutatakse pole avalikult kättesaadavad.

Githubi kasutajanimega KaiKaic1998 avatud lähtekoodiga repositooriumis[14] kasutab ta sihtmärgi jälitamiseks Siammaski mudelit [23], segmentimiseks Meta SAM mudelit eeskujul valminud hq-sam[15] mudelit ning pikslite asendamiseks Deep Video Inpainting mudelit[16]. Autor märgib oma töös, et kaasaegsete mudelitega on keeruline saavutada täpset ning kiiret segmentatsiooni korraga. Segementimise kvaliteet langeb, kui sihtmärgiks valitud objekt on osades kaadrites täielikult varjatud ning ilmub seejärel uuesti vaatevälja. Kvaliteet langeb ka siis kui sihtmärgi suurus on pidevalt muutuv ning kui sihtmärki suumitakse videos sisse- ning väljapoole. Reaalajas pildi täitmise teeb keeruliseks see, et paljud võimsad *video-inpainting* mudelid kasutavad töötamiseks ka kaadreid, mis järgnevad praegusele kaadrile. Käesoleva tööga võrreldes erineb selle koodibaasi ülesandepüstitus selle poolest, et seal puudub reaalaja videovoo implementatsioon ning see on mõeldud salvestatud videotel objektide segmentimiseks ning taustaga asendamiseks.

[14] https://github.com/kaikaic1998/Real_Time_Video_Inpainting_PoC?tab=readme-ov-file

[15] https://huggingface.co/lkeab/hq-sam

[16] https://github.com/mcahny/Deep-Video-Inpainting

4. Metoodika

4.1 Kasutatud tehnoloogiad

Python

Python on masinõppes enimkasutatud programmeerimiskeel ning töö praktilises osas kasutatud mudelite dokumentatsioonid põhinesid peamiselt Pythonil.

Flask

Flask on Pythonis kirjutatud mikro-veebiraamistik (ingl *micro web framework*)[17]. Sellel puuduvad veebiraamistikel tavapäraselt eksisteerivad andmebaasi- jt funktsionaalsused, kuid neid saab Flaskile laiendustena lisada[17]. Flask on kavandatud olema oma olemuselt lihtne ja skaleeritav ning sellega töötamisega ei kaasne suurt õppimiskõverat[17]. Valisin Flaski kasutamise ka sellel põhjusel, et mul on sellega kõikidest Pythoni veebiraamistikest kõige enam kogemust.

Docker

Docker on tarkvara, mille abil saab ehitada, jagada ning jooksutada konteineriseeritud rakendusi ja mikroteenuseid. Docker toetab paljusid laialt levinud tööriistu ning programmeerimiskeeli ning kasutajal on ligipääs paljudele usaldusväärsetele baaspiltidele ning mallidele, mille abil saab kasutaja koostada dockeripilte ja dockeri kontenereid.[18] Dockeri kasutamine võimaldas valminud demol olla arvuti ülejäänud protsessidest stabiilses keskkonnas eraldatud.

WSL

WSL ehk Windows Subsystem for Linux on Windowsi funktsionaalsus, mille abil saab Windowsi arvutil käivitada Linuxi keskkonna.[19] Selles töös kasutatakse WSL 2 versiooni, mis loob sisuliselt Windowsi operatsioonisüsteemi Linuxi virtuaalmasina. Virtuaalmasin käivitub isoleeritud konteinerina, kuid võimaldab lihtsat ligipääsu *host*-masina failisüsteemile ja

---

[17] https://pythonbasics.org/what-is-flask-python/

[18] https://docs.docker.com/desktop/

riistvaralistele ressurssidele.[19] Algselt tekkis praktilise osa käigus WSL-i Ubuntuga kasutamise vajadus, sest demos kasutatud SAM2.1 dokumentatsioonis oli selle kasutamine SAM mudeli abipakettide sujuvaks installeerimiseks ja kasutamiseks rangelt soovitatud.

OpenCV

OpenCV on avatud lähtekoodiga tarkvarateek, mis on arendatud reaal-ajas toimivate masinnägemise ja masinõppe lahenduste jaoks.[20] Teeki kuulub üle 2500 optimeeritud algoritmi, mis täidavad näotuvastuse, objektijälitamise, 3D rekonstruktusiooni, pilditöötluse, liitreaalsuse jpm ülesandeid.[20] OpenCV toetab C++, Python, Java ja MATLAB programmeerimiskeeli.[20] Edaspidi viidatakse käesolevas töös OpenCVle selle Pythoni mooduli cv2 nimega.

PyTorch

PyTorch on avatud lähtekoodiga paindlik ja kasutajasõbralik sügavõpperaamistik.[21] Selle abil saab ehitada ja treenida närvivõrke ning see toetab GPU-kiirendust.[21] Käesolevas töös kasutatakse PyTorchi SAM 2.1 mudeli laadimiseks, optimeerimiseks ning GPU-l käivitamiseks.

4.2 Eesmärgid ja kasutuses olev riistvara

Töö eesmärgiks oli segmentimise ja eemaldatud pikslite taastäitmise demo, mis suudaks võtta sisendiks 1280x720 formaadis kaadri ning suudaks kaadreid töödelda vähemalt kakskümmend tükki sekundis. Sellise demo arendamiseks oli plaanis katsetada selleks sobivaid kaasaegsemaid mudeleid ning valida lõpptulemuseks nendest sobivaim. Arenduskeelena kasutasin Pythonit, sest see on kaasaegse masinnägemise arenduse standard ning mul on sellega eelnevalt kõige enam kogemust.

Praktilise osa käigus kasutatud arvutite tehnilised näitajad on välja toodud tabelis 1.

[19] https://learn.microsoft.com/en-us/windows/wsl/about

[20] https://opencv.org/about/
[21] https://www.nvidia.com/en-eu/glossary/pytorch/

Tabel 1. Demo arendamise ja eksponeerimise käigus kasutatud arvutite riistvaraline ülevaade.

Arvuti	1.01.25 - 31.03.25	31.03.25 - 15.05.25
Operatsioonisüsteem	Windows 11 Education	Windows 11 Education
CPU	Intel(R) Core(TM) i7-9700K 3.60 GHz	AMD Ryzen 9 3950X 3.49 GHz
GPU	NVIDA GeForce RTX 2080, 8GB VRAM	NVIDIA GeForce RTX 3080, 10GB VRAM
RAM	24.0 GB	64.0 GB

Selle riistvara peal oli vaja lahendada kolm alamülesannet - pildi segmentimine, teatud objekti järgimine läbi kaadrite ning segmenditud ala asendamine taustaga (ingl *inpainting*). Järgnevalt loetlen kõiki tehnoloogiaid, mida nende ülesannete lahendamiseks katsetati ning toon välja, millised lõplikku rakendamisse valiti ning miks.

4.3 Eksperimendid segmentimismudelitega

SAM 2.1

Esimene segmentimismudel, millega praktilise osa katsetusi alustasin oli Meta SAM 2 (Segment Anything Model 2)[7]. Mudel on kõikvõimalike objektide segmentimises väga täpne ning segmenteeritava objekti valimiseks pildil võib kasutaja anda kas ühe või mitu punkti pildil, mis sellele objektile kuuluvad, või pildiala, mis objektile kuulub. Võimalik on ka kogu pildil kõik objektid korraga leida. SAM 2 on mõeldud piltidelt või salvestatud videotelt objektide segmentimiseks ning selle ametlikus versioonis puudub reaalajas kasutamise võime. Mudelit puudutav dokumentatsioon on pealiskaudne ning mudeli kasutamisel oli enim abi kogukonnapõhistest aruteludest. Kuna eesmärk oli reaalajas demo, siis katsetasin esimese lahendusena reaalaja kaadrite SAM 2 pildisegmentijaga töötlemist ning väjundina saadud maskide värvilisena ükshaaval ekraanile kuvamist. SAM2 objektide üle aja jälgimise võimekust saab kasutada vaid salvestatud videote puhul.

SAM 2 kõige uuem väljalase on SAM 2.1 ning sellel on neli võimalikku mudelisuurust: *tiny*, *small*, *base* ja *large*. Siinkohal kirjeldatud katsetuste ajal oli kasutuses esimene arvuti ning SAM2.1-*tiny* võimaldas 640x480 pikslit kaadrisuuruse korral kaadrihaaval pildisegmentijat kasutades ligikaudu 10 kaadrit sekundis, mille juures oli GPU hõivatus ligi 100%. Videovoo kaadrihaaval SAM 2.1 pildisegmentijaga töötlemise suurim puudujääk oli selle võimetus valitud objekti läbi mitme kaadri jälgida. SAM 2.1 ei sea sisendpildi resolutsioonile piiranguid, aga otsustasin katsetamise tulemusena kasutada sisendsuurust 512x512 pikslit, sest see saavutas segmentimiskvaliteedi ja -kiiruse vahel parima kompromissi.

Teisena kasutatud arvuti GPU jõudlus võimaldab SAM 2.1 oluliselt edukamat kasutamist: GPU üle 90% hõivatusega 1280x720 pikslit resolutisooniga videosisend saavutas kiiruse 15 kaadrit sekundis, mis on küll vähe, kuid autori hinnangul piisav esialgse demo loomiseks.

YOLO-v11-seg

Teine segmentimismudel, millega katsetasin, oli Ultralytics YOLO-v11-seg[22]. See mudel on objektipõhine segmentija ning selle vaikimisi versioon on treenitud COCO andmestikuga, mis sisaldab endas 80 klassi, sh “inimene”. Mudelil on väga põhjalik ja kvaliteetne dokumentatsioon, mis tegi selle kasutamise selgeks ja mugavaks. YOLOv-11 puhul kasutasin samuti kaadrihaaval pildisegmentijat ning selle mudeliga tuli objekti jälgimise võimekus kaasa. See mudel võimaldab sisendiks esitada vaid terve kaadri ning püüab igalt kaadrilt leida kõik objektiklassid, mille põhjal see treenitud on. Mudel osutus samuti pisut liiga ressursinõudlikuks, sest 1280 x 720 pikslit resolutsiooniga sisendiga saavutasin jõudluse alla 10 kaadri sekundis. Selle tulemusena saadud segmentimismaskid olid väga varieeruva kvaliteediga, näiteks kui kaamera ette seisis kirju särgiga inimene, ei olnud kogu inimene enam ühe “inimene” tüüpi objektina segmenditud, vaid erinevad särgi alad liigitati mingitesse muudesse klassidesse.

Mediapipe SelfieSegmenter

Kolmanda mudelina testisin Google’i arendatava Mediapipe raamistiku Selfie Segmentation lahenduse SelfieSegmenter mudelit[23]. Mediapipe on Google’i arendatud avatud lähtekoodiga

[22] https://docs.ultralytics.com/models/yolo11/

[23] https://ai.google.dev/edge/mediapipe/solutions/vision/image_segmenter

raamistik, mis on mõeldud erinevate masinõppe töövoogude koostamiseks.[24] Selle raamistiku Selfie Segmentation lahedus ongi mõeldud reaalajas inimest segmentima, kuid pakub ka piltidelt ja videotelt inimese segmentimise võimalust.

Selfie Segmentation lahenduse SelfieSegmenter mudelil on kaks versiooni: "SelfieSegmenter (square)" sisendsuurusega 256x256 pikslit ja "SelfieSegmenter (landscape)" sisendsuurusega 144x256 pikslit. Mõlemad mudelid põhinevad MovileNetV3 arhitektuuril, mis on oma olemuselt optimeeritud mobiiltelefonide protsessoriarhitektuurile. SelfieSegmenter mudeleid eristab ainult nende sisendi resolutsioon, mina valisin töös kasutamiseks ruudukujulise 256x256 pikslit sisendresolutsiooniga mudeli, sest testimise käigus ei näinud ma mõlema mudeli töökiiruses olulist vahet ning valitud resolutsiooniga mudel võtab sisendina suurema arvu piksleid.

Mediapipe Selfie Segmentation lahenduse dokumentatsioonis on jõudluse võrdlusalus (ingl *benchmark*) (joonis 1) koostatud Google Pixel 9 CPU ja TPU põhjal. Kasutasin mudeli töövoo koostamiseks MediaPipe'i Ppythoni API-t. Mudelile saab anda sisendiks nii pildi, video kui reaalajas videovoo. Mudeli dokumentatsioon oli puudulik ja pisut vananenud, näiteks avastasin, et selle reaalaja funktsionaalsuse dokumentatsioonis on selle videofailide töötluse meetodi kood. Saavutasin mudeliga soovitud resolutsioonil ja soovitud kvaliteediga segmentimise, kuid kaadrisagedus jäi alla viieteistkümne kaadri sekundis, samas kui võrdlusalus lubas Pixel 9 CPU peal 30 kaadrit sekundis. Tol hetkel kasutasin Inteli protsessoriga lauaarvutit, mille protsessori arhitektuur erineb põhimõtteliselt mobiiltelefonide omast. MediaPipe Python API ei võimalda GPU kasutamist ning juhendajaga arutelu tulemusena proovisin suunata MediaPipe sees kasutatavaid TensorFlow mudeleid GPU-le. See katse ei olnud edukaks ning võtsime juhendajaga vastu otsuse kasutada teist arvutit.

Teisel, AMD protsessoriga arvutil saavutasin kiiresti ligi 30 kaadrit sekundis segmentimise jõudluse, mis on demo eesmärgil piisav

Mediapipe Selfie Segmentation lahenduse teised mudelid

Lisaks SelfieSegmenter mudelile pakub Mediapipe Selfie Segmentation lahendus Pythoni API-ga kasutuseks veel juuste segmentimiseks mõeldud HairSegmenteri ja mitmeklassilise väljundiga SelifeMulticlass mudelit, mis segmendib eraldi viit kategooriat: taust, juuksed,

[24] https://ai.google.dev/edge/mediapipe/solutions/guide

kehaosade nahk, näonahk, riided ja other (muu). Samuti võimaldavad nad DeepLab-v3 mudeli kasutamist, mis on samuti mitmeklassilise väljundiga segmentija. Kuna Google demonstreerib neid mudeleid veebis[25], siis testisin HairSegmenteri ja SelfieMulticlass mudelit, aga minu arvuti CPU jõudluse juures jäi nende töövoogudega saavutatav resolutsioon ja kaadrisagedus alla poole minu seatud standardi. DeepLab-V3 ma ei testinud, sest Googlei pakutud jõudluse võrdlusalusest ja teiste mudelite testimisest selgus, et suure tõenöosusega jääks mu arvutil ka DeepLab-V3 mudeli kasutamiseks CPU jõudlust puudu.

Here's the task benchmarks for the whole pipeline based on the above pre-trained models. The latency result is the average latency on Pixel 6 using CPU / GPU.

Model Name	CPU Latency	GPU Latency
SelfieSegmenter (square)	33.46ms	35.15ms
SelfieSegmenter (landscape)	34.19ms	33.55ms
HairSegmenter	57.90ms	52.14ms
SelfieMulticlass (256 x 256)	217.76ms	71.24ms
DeepLab-V3	123.93ms	103.30ms

Joonis 1. Google Selfie Segmentation lahenduse mudelite jõudluse võrdlusalus.[23]

Olgugi, et Google koostas ülesande jõudluse võrdlusaluse Google Pixel 6 nutitelefoni peal ja minu kasutuses oleva lauaarvuti protsessor on tunduvalt võimsam, siis Mediapipe'i TFLite mudel on sedavõrd mobiiltelefonide protsessoritele optimeeritud, et praktilises töös kasutatud lauaarvuti CPU latentsus inimese segmentimise ülesande täitmisel (ingl *latency*) oli keskmiselt 50ms, mis on üle 50% suurem, kui Google'i poolt raporteeritud latentsus Pixel 6 protsessoril.

[25] https://mediapipe-studio.webapps.google.com/demo/image_segmenter

Katsetuste kokkuvõte

Katsetuste tulemusena selgus, et Mediapipe SelfieSegmener oli töö eesmärkide ja seatud kvaliteedistandarditega kõige täpsemini kooskõlas. See võtab sisendiks reaalaja videovoo, sellel on olemas objekti jälgimise võimekus ning lisaks on see riistvaraliselt vähenõudlik. SAM 2.1 puuduvad nii reaalaja videovoo sisendiks võtmise kui ka objekti jälitamise võimekus, kuid kuna tegu on võimeka segmentimismudeliga, mis lubab kasutajasisendi põhjal segmentida ükskõik millist objekti, siis tundus see mudel, mille töötamist oleks publikul huvitav näha. Saavutatud 15 kaadrit sekundis jõudlus hinnati "talutavaks" ning otsustati ikkagi ka seda tehnoloogiat kasutajatele näidata.

4.4 Objekti jälgimine ja puuduva ala täitmine

SAM2 tracking lahendus

Kogukonnapõhistes lahendustes leidub mitu koodirepositooriumi, mis on ehitanud SAM 2.1 videosegmentija ümber reaalajas segmentijaks. Praktilises osas katsetati üht nendest, kasutaja Gy920 loodud lahendust[26]. See lahendus ei olnud töö eesmärkidega täpselt kooskõlas, sest tollel hetkel kasutuses olev arvuti võimaldas selle kasutamist vaid 480x480 pikslit resolutsiooniga ning tulemuseks saadav videovoo kaadrisagedus oli alla kaheksa kaadri sekundis.

Kuna olime juhendajaga otsutanud SAM 2.1 lõplikus demos kasutamise, siis mõtlesime objekti jälitamiseks välja järgneva lähenemise: videovoog segmenditakse kaadrihaaval ning eelmise kaadri segmentimise tulemusel saadud segmentimismaski keskpunkt antakse järgmise kaadri segmentimisel sisendpunktiks. Kasutajale jääb vabadus hiire abil igal hetkel uus sisendpunkt ehk uus jälgitav ja eemaldatav objekt valida. Gy920 lahendust ei olnud võimalik eeskujuna kasutada, sest tema lahendus ei kasutanud SAM 2.1 pildi- või videosegmentija mooduleid, vaid selles oli loodud täiesti eraldiseisev *camera_predictor* moodul. Samuti oli tema lahenduse miinuseks see, et uue objekti valimisel initsialiseeritakse mudel iga kord uuesti, mis põhjustab aja- ja ressursikulu.

Kirjeldatud maski keskpunkti põhine jälitamine leiab lõplikus demos rakendust, kuid 15 fps (ingl *frames per second*) töötluse juures on piiratud tulemuslikkusega. Kiiremate liikumiste

[26] https://github.com/Gy920/segment-anything-2-real-time

ning väiksemate objektide puhul võib objekt töödeldud kaadrite vahel nii palju liikuda, et eelmise kaadri objekti keskpunkti asukoht ei kuulu enam sellele objektile. Siiski, sujuval liigutamisel suudab lahendus jälgida ka väiksemaid objekte nagu telefon või kohvitass.

Pildi täitmise lahendus

Kaasaegsed sügavõppel põhinevad pildi täitmise tehnikad on oma olemuselt reaalajas kasutamiseks liiga arvutusnõudlikud [20], mistõttu otsustasime juhendajaga kasutada piksliasendamist: pikslid, mis ei asu segmentijaga maskitud piirkonnas salvestatakse. Iga piksli kohta tekib justkui ajalugu - milline see piksel oli enne, kui objekt selle kattis. Töövoo sisselülitamise hetkel ei toimu segmenditud alal piksliasendust, sest sinna kuuluvate pikslite ajalugu sisaldab seda sama objekti hetk enne seda kui see eemaldamiseks välja valiti. Kui segmenditud objekt videovoos liikumist jätkab, siis on selle ala pikslite kohta, kuhu ta liikus, juba salvestatud tausta välimus ning need pikslid asendatakse maskitud alale. Kuna segmentimismudelitelt tulevad maskid pole ideaalselt täpsed, siis selleks, et kasutajale kuvatavas kaadris poleks inimese piirjooni, näiteks juuksesalke, kasutatakse maski morfoloogilist laiendamist cv2 *dialate()* funktsiooni abil. Lõplikusse demosse kuuluval Mediapipe SelfieSegmenter abil inimese segmentimise töövoos laiendatakse maski igas suunas ligikaudu 12 pikslit, SAM 2.1 töövoos ligikaudu kaks pikslit. Kuna Mediapipe SelfieSegmenter töövoo ainus eesmärk oli inimese täielikult pildilt kaotamine, aga SAM 2.1 töövoo ülesanne oli pildilt kaotada väiksemaid ja tihtipeale selgemate piirjoontega objekte, siis kujunes maski laiendamise ulatus katsetuste käigus erinevaks. Samuti kasutatakse SelfieSegmenter maski laiendamiseks ellipsikujulist laiendust, sest siis näeb tulemuseks saadud mask loomulikum välja.

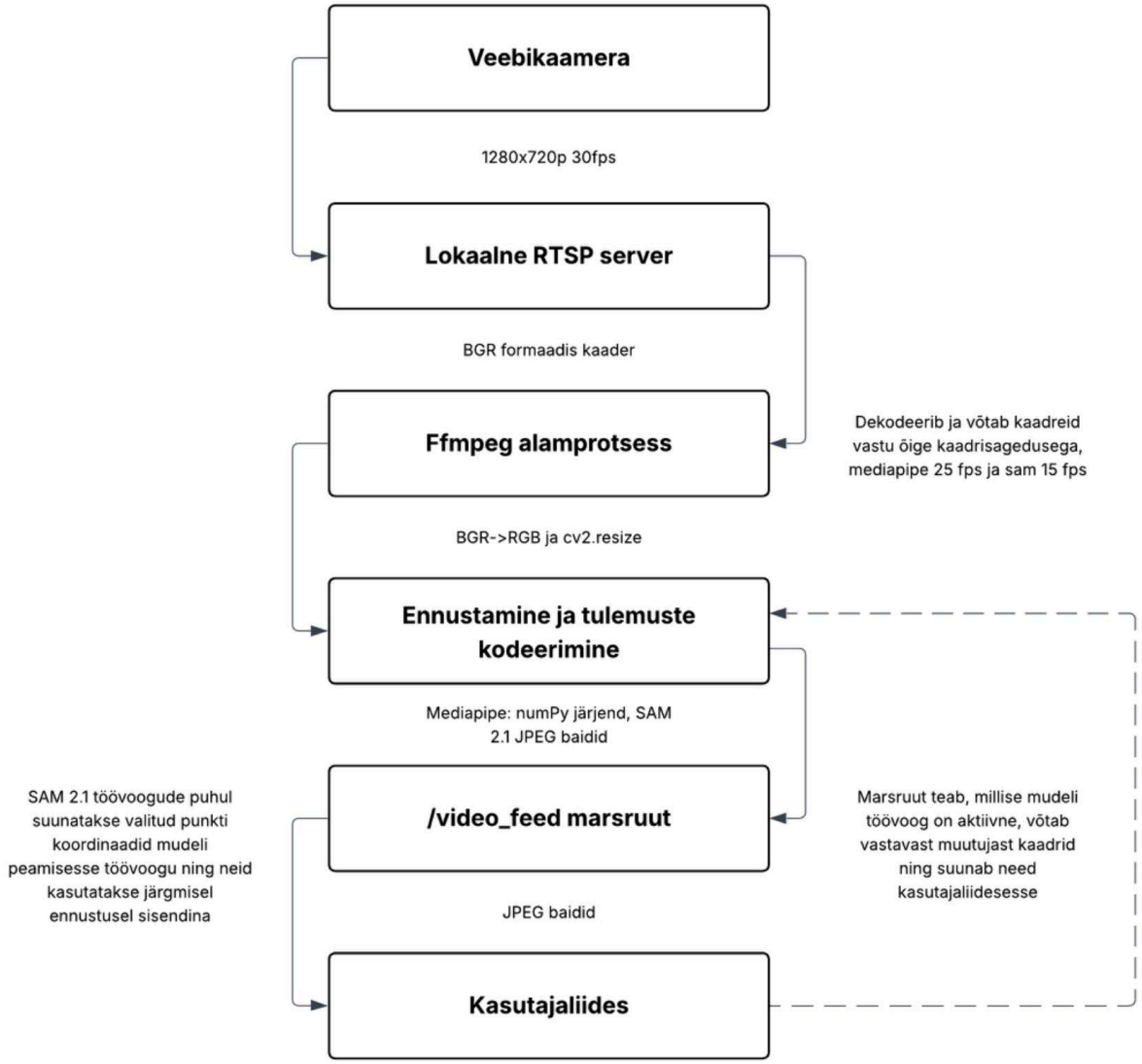
Järjendit, kust tulevad pikslid, millega segmenditud ala asendada, uuendatakse igal sissetuleval kaadril 3%. Eriti kui inimene liigub, siis tekib peaaegu igas kaadris inimese maskimisel ebatäpsusi, näiteks jääb osa õlast segmentimata. Kui taustapikslite ajalugu hoiustavat järjendit uuendada täielikult igal kaadril, oleks asendatav ala kaetud segmentimise ebatäpsusest tulenevate inimeseosadega. Kui järjendit uuendada vähehaaval, võetakse arvesse mitme kaadri segmentimistulemusi ning üksikutest kaadritest tulenevad vead ei jää videovoos nähtavale. Määratud 3% uuenduskiirus saavutas katsetuste käigus kõige loomulikuma välimusega tulemuse.

4.5 Lõpliku demo koostamine

Demo lõplikuks pakendamiseks koostasin selle Pythoni Flask veebiraamistikku kasutades lihtsa kujundusega kasutajaliidese (vt alapeatükk 6.1). Selles osas pidin mudelite töövoogudes suured muudatused tegema, sest sellele eelnevas tööprotsessis olin töövoogusid ainult ükshaaval käivitanud. Lõplikus demos on abimeetoditega saavutatud funktsionaalsus, mis teeb kindlaks, et kui kasutajaliideses kolme töövoo vahel vahetada, ei jääks eelmine protsess arvutilt ressursse, näiteks GPU VRAMi, võtma.

Kasutan oma demos REST API *endpointe*, et erinevaid töövoogusid käivitada ja välja lülitada. Samuti kasutan neid selleks, et SAM 2.1 demo jaoks vajalik kasutajasisend (hiireklikk, mis määrab millist objekti segmenditakse) mudelini jõuaks. Mudelitega töödeldud kaadrid jõuavad brauserisse JPEG baidivoo formaadis. Tööle kasutajaliidese koostamine tõi ka vajaduse Pythoni threading mooduli kasutamiseks. Töövoog peab ühel ja samal ajal lugema videovoost kaadreid, saatma need läbi mudeli ning Flask veebiserver peab otspunktides (ingl *endpoint*) tulemusi ootama. Flask veebiserver jookseb peamises lõimes ning mudeli töövoog taustalõimes. Mõnede koodiosade juures, näiteks cv2, NumPy ja videvoo sisendi ootamise ajal saab Python suunata ressursi tagasi Flaski peamise lõime juurde. Kaalusin lahendusena ka Pythoni Multiprocessing mooduli kasutamist, sest mul oli sellega eelnevalt kogemus ning selle abil saab töövoo erinevateks sünkroonselt jooksvateks protsessideks jagada. Multiprocessing mooduli kasutamise miinuseks on, et igal protsessil on erinev mäluruum, mistõttu on protsessidevahelise infovahetuse arendamine mitmetasandilisem ja piiratum kui mitmelõimelisuse puhul. Kuna sain demo vastavalt seatud standardile tööle, siis jäin Pythoni threading mooduli kasutamise juurde.

4.6 Töövoog



Joonis 2. DEMO töövoog skeemina.

Joonisel 2 kujutatud töövoo kulg:

1) Arvutisse ühendatud veebikaamera sisend suunatakse lokaalsesse rtsp (*real time streaming protocol*) serverisse, mis tekitab 1280x720 pikslit resolutsiooniga 25 kaadrit sekundis videovoo.
2) Kõigil kolmel demosse kuuluval vaatel on oma töövoog. Iga segmentija algatab kõigepealt ffmpeg protsessi, mis kasutab TCP protokolli, et võtta vastu rtsp voo saadetud kaadreid. Kuna ffmpeg protsessi väljundiks on toru (ingl *pipe*), saab Pythoni töövoog kaadreid ükshaaval ning dekodeerituna. Lugemise alamprotsess on vastava parameetriga seatud lugema kaadreid:
    - **Mediapipe** - 25 kaadrit sekundis, kaadreid loetakse otse ffmpeg alamprotsessi torust.
    - **SAM 2.1** - 15 kaadrit sekundis, kaadrid suunatakse ffmpeg alamprotsessist üheelemendilisse Pythoni andmetüüpi queue.Queue, mis saavutab kindluse, et mudelisse siseneb alati kõige hiljutisem kaader.
3) Peale seda, kui segmentija töövoo peatsükkel BGR-formaadis kaadri kätte saab, konverteeritakse see cv2 abil RGB-formaati ning selle suurust muudetakse.

Mediapipe

Sisendformaat on 256x256 pikslit. Kaader suunatakse mudeli asünkroonsesse *segment_async* funktsiooni. Vahetult enne seda salvestatakse *timestamp* muutujasse ajahetk mikrosekundites. See on vajalik *callback* funktsiooni jaoks, miks aktiveerub kohe, kui mudel on ennustamise lõpetanud.

Mediaipipei *callback* funktsioon kõigepealt laiendab maski, et pidile maski ebatäpsusest tulenevaid inimese piirjooni ei jääks. Seejärel salvestatakse backgroundBuffer muutujasse järk-järgult kõigi pikslite väärtused, mis klassifitseeruvad mudeli hinnangul taustapikslite hulka. BackgroundBufferist tulevate pikslivääртusega asendatakse inimene. Taustapiksleid hoiustava järjendi vähehaaval uuendamine aitab mudeli ennustuste ebatäpsuse mõju vähendada, sest siis ei jää iga valesti klassifitseeritud piksel kohe videovoos näha.

SAM 2.1

Sisendformaat on 512x512 pikslit. SAM 2.1 töövood saadavad mudelisse lisaks rgb formaadis kaadri NumPy järjendina koordinaatides kasutaja valitud punkti. SAM 2.1 töövoog võimaldab PyTorchi abil mudelil GPU kasutamise.

SAM 2.1 mask töövoos viiakse mudelilt saadud mask originaalkaadriga samasse formaati ning cv2 abil lisatakse mask punaselt ja läbipaistvana kaadrile.

SAM 2.1 piksliasendusega töövoos viiakse mask samuti kõigepealt õigesse formaati, kuid värvimise asemel asendatakse mask backgroundBufferi muutujast tulevate pikslivääртustega. Järjendi uuendamine toimub samuti vähehaaval. SAM 2.1 inpaint töövoos suurendatakse mudelilt saadud maski igas suunas ligikaudu viie piksli võrra, Mediapipe töövoos 25 piksli võrra.

4) Kõigis töövoogudes lisatakase kaadrile FPS-lugeri väärtus ning cv2.imencode abil kodeeritud kaader suuunatakse jagatud muutujasse.
    - **MediaPipe** töövoog salvestab mutuujasse NumPy massiivi
    - **SAM 2.1** töövood salvestavad muutujasse kaadri JPEG-baitidena.
5) Flask serveris asuv /video_feed marsruut kontrollib, milline töövoog on parajasti aktiivne ning võtab vastava muutuja alt saabuvad kaadrid. Kohe kui /video_feed on uue kaadri edasi suunanud, kuvab index.html failis asuv <img> element seda videopildis.

SAM 2.1 töövoogude puhul saab JavaScript aru, kui kasutaja valib videopildil uue punkti. Punkt konverteeritakse SAM 2.1 sisendi 512x512 resolutisooni ning kirjutatakse input_point massiivi.

6. Tulemused

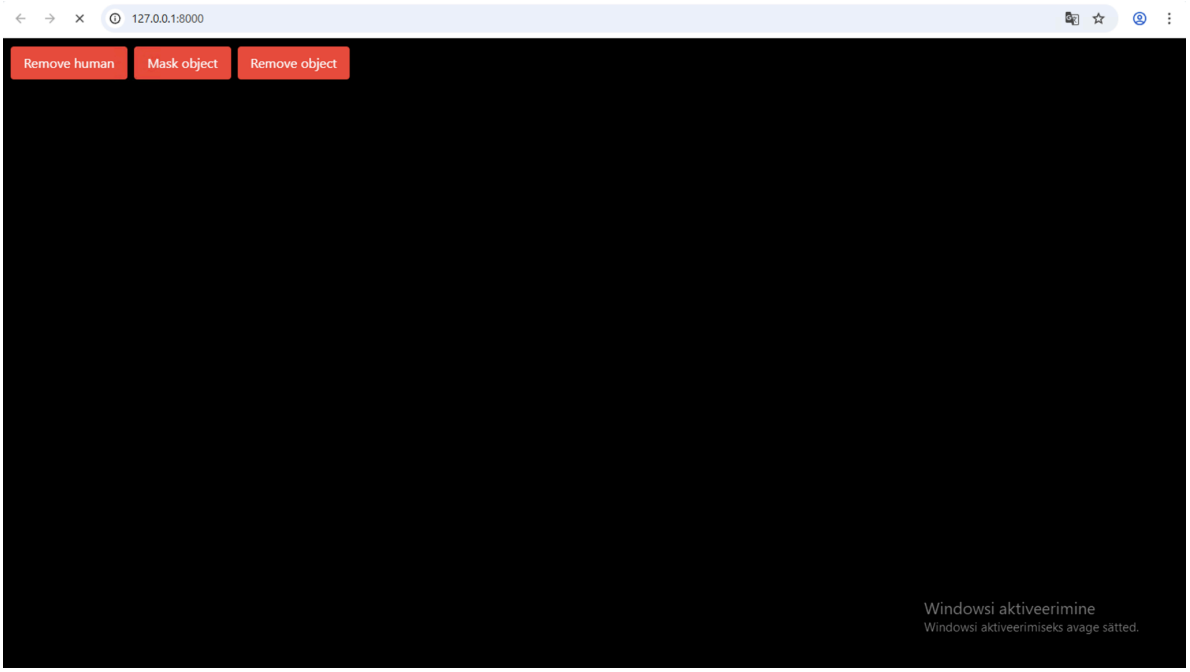
6.1 Lõpliku demo kooslus

Lõplik demo koosneb kolmest osast:

1. Inimese segmentimine ja taustaga asendamine Mediapipe SelfieSegmenter mudeliga. Inimene seisab kaamera ette, ta tuvastatakse automaatselt ja tema terve keha segmenditakse ning asendatakse taustaga. Kaadrisagedus on keskmiselt 25 kaadrit sekundis ning resolutsioon on 1280x720p.

2. Valitava objekti segmentimine Meta Segment Anything 2.1 mudeliga. Kasutaja valib hiire abil punkti, mida segmentida, ning objekt, millele see punkt kuulub, segmenditakse reaalajas ning kaetakse läbipaistva punast värvi maskiga. Kaadrisagedus on keskmiselt 15 kaadrit sekundis ja resolutsioon on 1280x720 pikslit. Valitavaks objektiks sobib nii inimene, inimese kehaosa, inimese käes olev objekt või mõni taustale kuuluv element.

3. Valitava objekti segmentimine ja taustaga asendamine Meta Segment Anything 2.1 mudeliga. Kasutaja valib hiire abil objekti, objekt segmenditakse, asendatakse taustaga ja seda jälitatakse reaalaja videovoos. Valitavad objektid, kaadrisagedus ja sisendresolutsioon on samad, mis eelmises alapunktis kirjeldatud SAM 2.1 maskimise demos.



Joonis 3. Kasutajaliidese välimus, kui ükski töövoog pole aktiivne.

Kasutajaliideses (joonis 3) on kolm nuppu, mis igaüks käivitab erineva töövoo. Aktiivne saab olla ainult üks töövoog korraga ning aktiivset töövoogu tähistab sellele vastava nupu roheline värv.

6.1.1 Inimese eemaldamine

Joonis 4. Autor segmendituna reaalajas Mediapipe SelfieSegmenter mudeliga.

Töövoog käivitub nupuga “Remove human”. Kuvatõmmis töövoo toimimisest on joonisel 4.

Link demovideole: https://tinyurl.com/tatsvm7s

Mediapipe SelfieSegmenter töövoog asendab taustaga edukalt kas ühe või mitu inimest ning selle juures on CPU ressursikasutus ligikaudu 15%. Töövoog on tundlik valgusoludele, kui sisendkaadris on inimene hämaras valguses, langeb segmentimiskvaliteet märgatavalt.

Töövoog reageerib igasugusele liikumisele sujuvalt, ainus olukord, kus inimene jääb peaaegu alati videovoos näha, on see, kui kasutaja seisab näoga otse kaamera poole ja tõstab siis järsult sirged käed enda kõrvale üles. Selline olukord on näha ka üleelmises lõigus viidatud demovideos. Töövoog on võimeline segmentima ka mitut inimest, siiski toimib ta täpsemini, kui kaamera ees seisab üks inimene korraga.

6.1.2 SAM 2.1 objekti maskimine

Joonis 5. Autori juuksed segmenditуna SAM 2.1 objekti maskimise töövoos.

Töövoog käivitub nupuga “Mask object”. Kuvatõmmis töövoo toimimisest on joonisel 5.

Link demovideole: https://tinyurl.com/kcydj27j

SAM 2.1 objekti maskimise töövoo peamine funktsioon on anda kasutajale arusaam, kuidas mudel vastavalt sisendile objekte segmendib. Sellel puudub jälitamisvõimekus ning kasutaja valitud punkt jääb kuni järgmise punkti vajutamiseni staatiliseks.

6.1.3 SAM 2.1 objekti eemaldamine

Töövoog käivitub nupuga “Remove object”. Kuvatõmmis töövoo toimimisest on joonisel 6.

Link demovideole: https://tinyurl.com/c3c33v5y

Joonis 6. Autori käes oleva lauatenniseretketi videovoolt eraldamine. Roheline punkt on eemaldatud objekti keskpunkt, mida üle aja uuendatakse ning mille kaudu on objekti jälgimine üle aja implementeeritud.

SAM 2.1 objekti eemaldamise töövoog võimaldab kasutajale demonstreerida nii SAM 2.1 segmentimisvõimekust, autori ja juhendaja loodud maski keskpunkti põhist jälitamisfunktsionaalsust ning piksliasendust. See, kui hästi objekt videovoolt eemaldatakse sõltub suuresti objektivalikust. Nupule “Remove object” vajutades käivitub töövoog nullist ning sellega kaasneb ka taustapiksleid sisaldava tühja järjendi initsieerimine. Järjend täidetakse esimese voogu läbiva kaadri sisuga, ehk kui valida objekt, mis pole töövoo käivitamise ajast liikunud, ei eemaldata objekti kaadrist.

Valitavaks objektiks sobib kõige paremini selgete piirjoontega objekt, näiteks punast värvi lauatenniserekt. Selle punane värv loob tumedates toonides taustaga selge kontrastierinevuse, mistõttu on selle segmentimismaskid kaadrist kaadrisse võrdlemisi sama suuruse ja kujuga. Sellist objekti on töövool kõige lihtsam jälitada, sest kui objekti liigutada, liigub selle maski

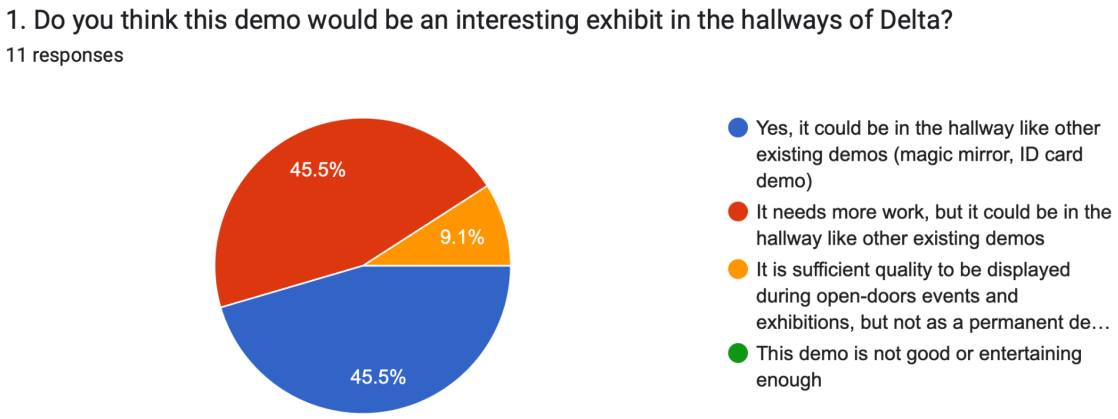
keskpunkt sellega sujuvalt kaasa. Terve inimese segmentimine ja jälitamine ei tööta selles töövoos kuigi hästi, sest SAM 2.1 segmendib vahepeal inimest täielikult ning vahepeal segmendib tema täpsemaid osi, näiteks riidemustreid, juukseid, ainult nägu. See muudab maski keskpunkti hüplikuks ning objekti jälitamine ei realiseeru. Siiski on heade valgusolude korral võimalik selles režiimis saavutada näiteks ainult juuste või näo kaadrist eemaldamise pildilt, tekitades põnevaid ja kummastavaid stseene, mida kasutajad ka telefoniga pildistada soovivad.

6.2 Kasutajate tagasiside

Eksponeerisin valminud demo vahemikus 8.05.2025 - 13.05.2025 Delta keskuse töötajate puhkealal (joonis 7).

Joonis 7. Demo eksponeerimine Deltas töötajate puhkealal (13.05.2025). Ekraanil on näha, et pildi tegemiseks kasutatud mobiiltelefon on segmenteeritud ning taustaga asendatud.

Joonisel 7 on näha arvuti peale asetatud juhendit ning sellel olevat QR-koodi mis viis kasutaja seitsmest küsimusest koosneva Google Forms tagasisidevormini. Et instituudis on ka eesti keelt mitte valdavad töötajad ja külastajad, oli küsimustik inglise keeles. Vastanuid oli demo eksponeerimise lõpuks 11.



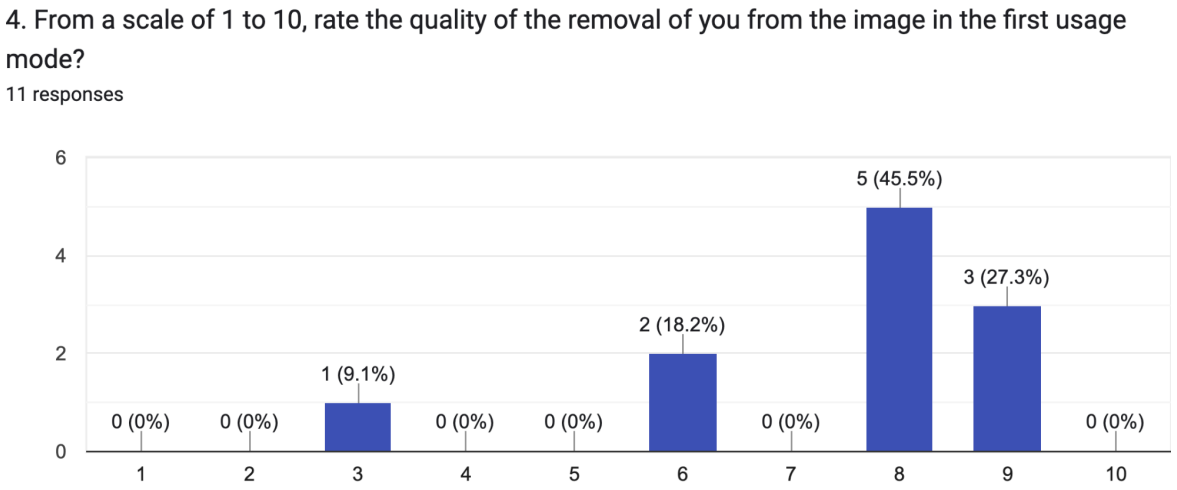
Joonis 8. Kasutajate tagasiside vormi esimesele küimusele.

Esimene küsimus (joonis 8) puudutas demo üldist kvaliteeti ning huvitavust. Kümme kasutajat arvas, et demo oleks Delta koridoris eksponeerimiseks sobiv, küll aga arvasid kümnest kasutajast pooled, et demo peaks veel täiustama. Üks kasutaja arvas, et demo on küll piisav avatud ustega üritustele, aga alaliseks eksponeerimiseks ei sobi. Arvatavasti piisaks lahenduse täiustamiseks võimsamast arvutist ja graafikakaardist, mis lubaks kõrgemat kaadrisagedust ning resolutsiooni, mis aitaks muuhulgas ka objektide jälgimise kvaliteeti tõsta.

Järgneva kahe küsimuse juures sai valida korraga mitu vastust.

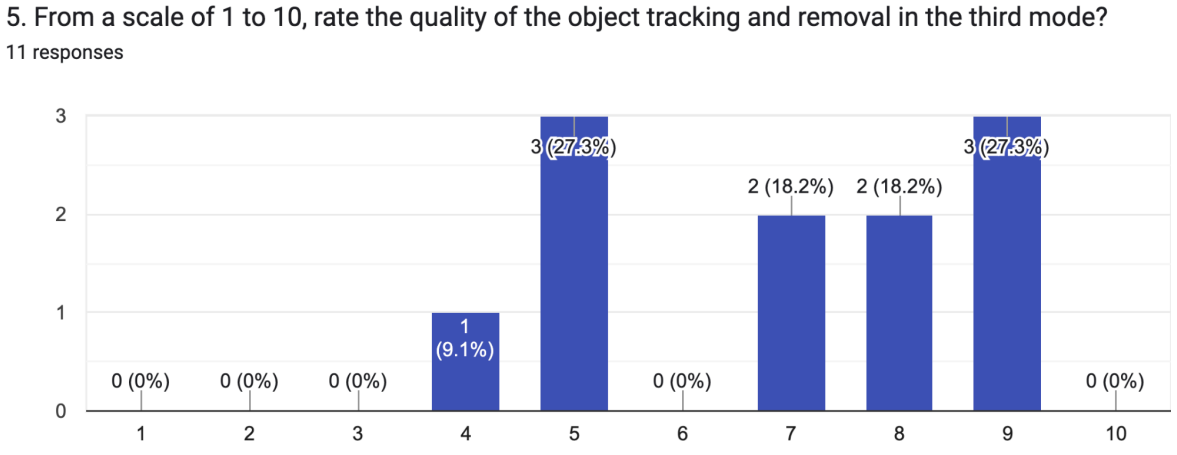
Vormi teine küsimus oli, et millise Delta külastajagrupi jaoks oleks demo põnev või meelelahutuslik. Kõik küsimustikule vastanud (11/11) arvasid, et demo oleks põnev tudengitele ja keskkooliõpilastele. Delta töötajatele (8/11), giidituuridel osalejatele (9/11) ja VIP külastajatele (5/11) peeti demo vähem põnevaks. See tulemus võib viidata, et demo sisu polnud hetkel piisavalt uudne või tehniliselt keeruline, et tekitada põnevust suurema kogemusega külastajates.

Kolmanda küsimuse sisuks oli, et milliseid eesmärke täidaks demo Deltas eksponeerimine. Hindajad nõustusid, et demo harib kasutajaid pilditöötluse tehnoloogiate vallas (10/11 vastajatest), õpetab neile midagi uut ning potentsiaalselt suurendab õpilaste huvi sellise tehnoloogia vastu. Üheksa vastanut arvas, et demo võib Delta koridorikeskkonda põnevamaks muuta ning demonstreerib bakalaureusetudengitele, et sellise töö loomine on nende kompetentsitaseme juures võimalik.



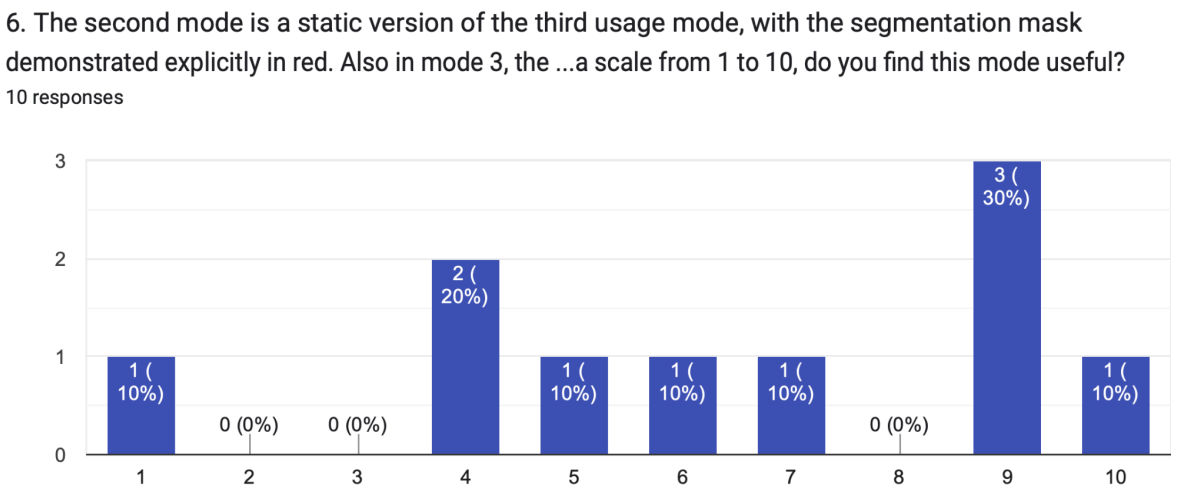
Joonis 9. Tagasisidevormi neljas küsimus, mis palus kasutajatel hinnata Mediapipe SelfieSegmenter mudeli töövoo toimist kümnepallliskaalal.

Kaheksa kasutajat hindas esimese töövoo kvaliteeti vähemalt väärtusega kaheksa (joonis 9) ja kaks kasutajat hindas väärtusega kuus. Vaid üks kasutaja andis töövoo toimisele madala hinde ehk kolm. Kuna demo oli eksponeeritud ruumis, kus olid aknad, siis olid sellel ajaperioodil sisendvideo valgustingimused pidevalt muutuvad, mis muutis ka töövoo segmentimiskvaliteeti. Sellest järeldan, et kui demo alaliselt eksponeerida, tuleb selle asukohaks valida võimalikult heade ja stabiilsete valgusoludega keskkond. Üldiselt võib aga järeldada, et esimene töövoog toimis hästi ning kasutajad olid sellega rahul.



Joonis 10. Tagasisidevormi viies küsimus, mis palus kasutajatel hinnata SAM 2.1 objekti jälitamis- ning asendusfunktsionaalsusega töövoo toimimist kümnepalliskaalal.

Üle poole kasutajatest (7/11) hindas SAM 2.1 objektieemaldamise töövoo kvaliteeti vähemalt hindele 7/10 (joonis 10). Ülejäänud kasutajad (4/11) arvas, et töövoo toimiskvaliteet väärib hindeks viis punkti kümnest või vähem. Sellest järeldub, et ilmselt proovisid kasutajad segmentida ebamääraste piirjoontega objekte, või liigutasid valitud objekti liiga järsult. Üks lahendus antud probleemile oleks ilmselt suurema riistvaralise jõudlusega arvuti kasutamine, sest oleks võimalik suurendada sisendvoo kaadrisagedust ning töövoog poleks valitud objektide järsult liigutamisele niivõrd tundlik.



Joonis 11. Tagasisidevormi kuues küsimus, mis palus kasutajatel hinnata SAM 2.1 objekti segmentimist demonstreeriva töövoo toimimist kümnepalliskaalal. Sellele küsimusele oli vastanuid 10.

Vaid 40 protsenti kasutajatest hindas segmentimist demonstreerivat töövoo vajalikkust vähemalt hindele 9/10 (joonis 11). Teine 40 protsenti kasutajatest andis hindeks 5/10 või vähem. Ülejäänud 20% kasutajatest hindas töövoo vajalikkust vähemalt hindele 6/10. See on eeldatav tulemus, sest kolmandas töövoos toimuv objekti jälitamine ning taustaga asendamine on kasutajatele põnevam, kui lihtsalt valitud objekti visuaalselt taustast eraldamine. Siiski koguksin võimalusel rohkem tagasisidet, enne kui eemaldaksin SAM 2.1 segmentimist demonstreeriva töövoo täielikult lõplikust demost.

Seitsmes küsimus palus kasutajatel hinnata demo üldist toimimist ning demo kõrval asetsenud kasutamisõpetust. Sellele küsimusele vastas viis inimest ning nende soovitused olid, et kasutajajuhiste kirjasuurus võiks olla suurem ning juhend võiks olla värvilisema kujundusega. Üks kasutaja arvas, et juhend võiks olla lakoonilisemalt sõnastatud, näiteks võiks inimese eemaldamist puudutava vaate juhendis olla lihtsalt kiri "vajuta nupule, et eemaldada end videovoost". Samuti arvas üks kasutaja, et demol võiks puuduta kaameralatentsus. Järeldan antud küsimuse vastustest, et pean demo juhendi täielikult ümber kujundama. Kaamera latentsus on tingitud sellest, et demo oli vormistatud käivituma dockeri konteineris ning Windowsi operatsioonisüsteem ei võimalda veebikaamerat dockeri konteinerisse sisendiks andma. Parim lahendus antud probleemile oleks demo käivitamine Linux operatsioonisüsteemiga arvutil.

Kaheksas ja viimane küsimus palus kasutajatel hinnata seda, et kuidas demo edasi arendada. Ka sellel küsimusel oli viis vastajat ning nende soovitusteks oli kasutajaliidese värvilisemaks muutmine, ning demo üldise efektiivsuse parandamine. Samuti soovitas üks kasutaja demo kunstiinstallatsioonina esitlemist. Demo üldise efektiivsuse parandamisele aitaks kindlasti kaasa parem riistvara, mis võimaldaks suuremat kaadrisagedust.

6.3 Töö võimalikud edasiarendused

Enamik kasutajatelt madala hinde saanud aspekte tulenes SAM 2.1 töövoogude madalast kaadrisagedusest. Kui kasutusel oleks paremate tehniliste näitajatega, peamiselt võimekama GPU-ga arvuti, saaks nendes töövoogudes kaadrisagedust suurendada. Praeguse kaadrisagedusega (15 kaadrit sekundis) on töövoo töötamise ajal GPU hõivatus üle 90%.

Lisaks oleks võimekama riistvara korral võimalik katsetada erinevaid sügavõppel põhinevaid pildigenereerimise lahendusi, nagu olid seda teinud 3. peatükis kirjeldatud lahenduste autorid. Seeläbi oleks võimalik uurida, kas generatiivne pilditäitmine saavutaks kasutaja jaoks loomulikuma tulemuse, kui käesolevas töös kasutatud piksliasendus. Võimekam riistvara võimaldaks ilmselt ka alapeatükis 4.3 kirjeldatud praeguse riistvara jaoks liiga arvutusnõudlike segmentimismudelite kasutamist.

Kokkuvõte

Käesolev töö andis ülevaate segmentimise põhiolemusest, segmentimismudelite arengust ja nende praktilistest rakendustest. Töö praktilise osa käigus analüüsiti kaasaegsemaid segmentimismudeleid ning koostati nende abil inimese ja objektide segmentimiseks mõeldud reaalaja videovoogu töötlev demo. Demo eesmärgiks oli kasutajaid informeerida kaasaegsete segmentimismudelite võimekusest ning pakkuda meelelahutust. Valminud demos kasutatakse ka segmenditud objektide jälgimist üle aja ning nende taustapikslitega asendamist, et need kaadrist eemaldada. Töö sisaldab endas valminud demo arhitektuuri täpset kujunemislugu ning kirjeldust.

Loodud lahendus suudab “eemalda inimene” vaates sujuvalt, 25 fps sageduse juures, eemaldada kaadrist inimesi, asendades neid üsna loomuliku välimusega taustaga. Samuti näitlikustab programm kaasaegse tippmudeli SAM 2.1 võimekust kahes erinevas vaates. Esimene neist näitab, et see mudel suudab avastada klikitud objekti piirjooned, värvides sellele objektile kuuluva ala kaameravoos punasega. Teine SAM 2.1 vaade tuvastab klikitud objektile kuuluva ala ning peidab objekti, jälgides seda ka üle aja ning reageerides liikumistele.

Töö käigus valminud demo eksponeeriti Delta keskuse töötajatele, koguti kasutajatelt tagasisidet ning analüüsiti seda. Tagasiside põhjal võib väita, et demo täidab oma hariduslikku ning inspireerivat eesmärki ja inimesed peavad seda Deltas eksponeerimiseks sobilikuks, küll aga saaks võimekamat arvutit kasutades selle tulemusi parandada.

Kasutatud kirjandus

[1] OpenAI (2025). ChatGPT (4o): https://chat.openai.com.

[2] Wang Y, Ahsan U, Li H, Hagen M. A comprehensive review of modern object segmentation approaches. *Foundations and Trends® in Computer Graphics and Vision*. 2022 Oct 4, Vol. 13: No. 2-3, 111-283.

[3] Venkatkumar. Segmentation: Traditional & Deep learning Approaches. *Medium*, 2024. https://medium.com/@VK_Venkatkumar/segmentation-traditional-deep-learning-approaches-edd50a3308b3 (15.05.2025)

[4] Long, J., E. Shelhamer, and T. Darrell. Fully convolutionalnetworks for semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[5] Ronneberger, O., P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[6] Chen, L.-C., G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*. 2017, 40(4): pp. 834–848.

[7] Xie, E., W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P.Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*. 2021, 34: pp. 12077–12090.

[8] Hariharan, B., P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. *European conference on computer vision*. Springer, 2014. pp. 297–312.

[9] O Pinheiro, P. O., R. Collobert, and P. Dollár. Learning to segment object candidates. *Advances in neural information processing systems*. 2015, nr. 28.

[10] Pinheiro, P. O., T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. *European conference on computer vision*. Springer, 2016. pp. 75–91.

[11] He, K., G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[12] Tian, Z., C. Shen, H. Chen, and T. He. Fcos: Fully convolutional one-stage object detection. *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9627–9636.

[13] Bolya, D., C. Zhou, F. Xiao, and Y. J. Lee. Yolact: Real-time instance segmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9157–9166.

[14] Wang, X., T. Kong, C. Shen, Y. Jiang, and L. Li. Solo: Segmenting objects by locations. *European Conference on Computer Vision*. Springer, 2020, pp. 649–665.

[15] Guo, R., D. Niu, L. Qu, and Z. Li. SOTR: Segmenting Objects with Transformers. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 7157–7166.

[16] Csurka G, Volpi R, Chidlovskii B. Semantic image segmentation: Two decades of research. *Foundations and Trends® in Computer Graphics and Vision*. 2022 Oct 18, 14(1-2), pp. 1-62.

[17] Matthew-Mcmullen. Seeing the World in Pixels: The Real-World Applications of Semantic Segmentation. *Medium*, 2024. https://medium.com/https-towardsdatascience-com/seeing-the-world-in-pixels-the-real-world-applications-of-semantic-segmentation-ac40fb7b09c1 (15.05.2025)

[18] Rohit Kundu. Video Segmentation: Intro, Methods, Tutorial. *V7labs,* 2023. https://www.v7labs.com/blog/video-segmentation-guide (15.05.2025)

[19] Yu T, Feng R, Feng R, Liu J, Jin X, Zeng W, Chen Z. Inpaint anything: Segment anything meets image inpainting. *arXiv preprint*, 2023, nr 2304.06790.

[20] Wojtek Jasinski. Pushing the Boundaries with Real-Time Video Inpainting. *Medium*, 2024. https://blog.swmansion.com/pushing-the-boundaries-with-real-time-video-inpainting-35cd82c624fb (15.05.2025)

[21] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B. Swin transformer: Hierarchical vision transformer using shifted windows. *IEEE/CVF international conference on computer vision*, 2021, pp. 10012-10022.

[22] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B. Swin transformer: Hierarchical vision transformer using shifted windows. *EEE/CVF international conference on computer vision*, 2021, pp. 10012-10022.

[23] Hu W, Wang Q, Zhang L, Bertinetto L, Torr PH. Siammask: A framework for fast online object tracking and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023; nr 45(3), pp. 3072-89.

Lisad

Demo sisaldav avalikult kättesaadav repositoorium:
https://github.com/rasmusmade/public-thesis-repo.git

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Rasmus Made,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose “Reaalajas objekti valimine, jälgimine ja asendamine taustaga”, mille juhendaja on Ardi Tampuu, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada Tartu Ülikooli digitaalarhiivi kuni autoriõiguse kehtivuse lõppemiseni;

2. annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi kaudu Creative Commonsi litsentsiga CC BY NC ND 4.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni;

3. olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile;

4. kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Rasmus Made

15.05.2025