TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Informaatika eriala

Tarmo Paavel

Liikumistuvastusega kadudeta video salvestamine Android seadmel

Bakalaureusetöö (6 EAP)

Juhendaja: M. Niitsoo, PhD

Autor: ..... ”.....” mai 2014
Juhendaja: ..... ”.....” mai 2014

Lubada kaitsmisele
Professor: …..... ”.....” mai 2014

TARTU 2014

Liikumistuvastusega kadudeta video salvestamine Android seadmel

Lühikokkuvõte:

Käesolev töö kirjeldab miks ja kuidas loodi liikumistuvastusega kadudeta videot salvestav mobiilirakendus Android platvormile. Töös antakse ülevaade liikumise tuvastamisest ja NV21 formaadist. Töö viimane osa kirjeldab rakenduse teostust.

Võtmesõnad:

Liikumise tuvastamine, video salvestamine, Android rakendus, NV21 formaat.

Lossless video recording using motion sensor on Android platform

Abstract:

This thesis describes why and how a motion detecting and lossless video recording application was created for Android platform. The paper gives short overview of motion detection and NV21 format. The last part describes implementation of the application.

Keywords:

Motion detection, video recording, Android application, NV21 format.

Sisukord

Sissejuhatus

Õppimisel on oluline saada kiiret tagasisidet oma vigade kohta. Eriti oluline on see spordis, kus pikalt valesti treenides on lihasmälu tõttu hiljem ümberõppimine keeruline. Lisaks võib suureneda vigastuste oht. Vigu on kergem tähele panna kõrvaltvaatajal, kelleks ideaalis on sportlase alal pädev treener. Kuid treener ei saa alati kõrval olla, eriti amatöörspordis, ning sellepärast tuleb leida alternatiivseid lahendusi. Üks võimalus on panna raja äärde treeningut salvestama tänapäeval laialt levinud nutitelefon. Sel moel saab sportlane salvestise abil analüüsida, mis tuli välja hästi ja mis halvasti, ning seeläbi arendada oma jooksu-, hüppe- või mõnd muud tehnikat. Lihtsam on vigadest aru saada, kui näed tehtut oma silmaga, eriti kui seda saab näha aegluubis.

Töö autor harrastab Eestis vähe levinud motosporti, mille nimeks on traiel (i.k. *motorcycle trials, observed trials*). Oluline osa on traielis erineva raskusastmega takistuste ületamisel. Treeningul õigete sõiduvõtete omandamiseks saab kasutada takistuse kõrvale seatud nutitelefoni, mis toimuvat salvestab. Paraku on aga probleem selles, et pärast telefoni salvestama panemist möödub piisavalt palju aega, enne kui sportlane takistuse ületamiseni jõuab, sest aega kulub kinnaste kätte panemisele, mootorratta käivitamisele, takistuseni jõudmisele jms. Treeningutel ületatakse sama takistust mitmeid kordi järjest. Takistuste ületamise vahepeal on sõitja pikalt kaadrist väljas, mille tõttu jääb salvestisele ka palju staatilist pilti. Analoogselt kulub ka kaugus- ja kõrgushüppajal katseks ettevalmistumiseks rohkelt aega. Seega on hiljem sellise salvestise põhjal ebamugav tehtut analüüsida, sest ligi 90% osas on tegemist staatilise pildiga ja palju aega kulub videost soovitud osade ülesleidmisele.

Lahenduseks oleks mobiilirakendus, mis tuvastaks liikumist ning alustaks seejärel video salvestamist. Liikumist tuvastavaid rakendusi on Android platvormile tehtud küll mitmeid, kuid ükski neist pole antud olukorra jaoks piisavalt mugav ega ole võimeline salvestama videot ilma, et algusest mõne sekundi jagu liikumist salvestamata jääks.

Antud bakalaureusetöö käigus luuakse videot salvestav rakendus, kus liikumist tuvastatakse kaamera videorežiimis, analüüsides kaamera poolt edastavaid kaadreid mingi

fikseeritud intervalliga (nt iga 0,5 sekundi järel). Seeläbi jääb ära aeganõudev üleminek pildistamise režiimist video salvestamise režiimi. Saadud kaadrite analüüsimise teel tuvastatakse liikumine ning alustatakse video salvestamist. Samal ajal jätkatakse liikumise tuvastamist ning liikumise lõppedes lõpetatakse ka video salvestamine. Antud töö käigus püütakse leida sobiv liikumise tuvastamise algoritm, mis tuvastaks liikumist piisava täpsusega, kuid arvestaks ka nutitelefoni väiksema protsessori jõudlusega. Lisaks peaks rakendus tagama, et salvestisele jääks kogu liikumine, sealhulgas ka liikumise tuvastamise hetkele eelnev liikumine. Tuvastades videokaadris liikumise suudab rakendus puhvris hoitud kaadrid salvestada, kindlustades sellega, et kaduma ei lähe hetkegi salvestatavast liikumisest. Seega oleks rakenduse väljundiks videoklipp, mis sisaldab ainult lõike, mil toimus liikumine.

Tõuke antud rakenduse loomiseks andis autori soov omandada Android platvormi mobiilirakenduse loomise kogemus ning tema isiklik vajadus antud rakenduse järele.

1. Tasuta rakendused ja nende puudused

Google Play poes [1] võib Android platvormile leida erinevaid liikumist tuvastavaid ja videot salvestavaid rakendusi, kuid antud töö autoril nende hulgast sobivat leida ei õnnestunud.

Järgnevalt on välja toodud autori poolt Google Play poest leitud Android rakendused, mis oskavad liikumist tuvastada ja seejärel käivitada video salvestamise:

- Motion Spy Video Recorder
- Camera Trigger
- Spy Motion DetectorLite
- Surveillance
- SECuRET Demo

Antud rakenduste paigaldamisel ja katsetamisel selgus, et kõigil rakendustel on järgnevad 4 puudust:

- Analüüsitakse pildistamise režiimis kaamerast saadud pilte ning liikumist tuvastades minnakse üle video salvestamise režiimi ja alustatakse video salvestamist. Salvestamisrežiimi vahetus ja video käivitamine aga võtavad palju aega ning video salvestamise alguseks võib oluline osa filmitavast sündmusest juba toimunud olla (kaugus- ja kõrgushüppes jäävad nägemata väga olulised hüppele eelnevad viimased sammud või lausa osa hüppest, traielis on oluline osa takistuse ületamise ettevalmistavast algfaasist juba möödas jne).
- Video salvestamist ei lõpetata automaatselt peale liikumise lõppu, vaid on määratud aeg, kui pikalt videot salvestatakse.
- Ei liideta salvestisi automaatselt üheks salvestiseks.
- Kuna rakenduste lähtekoodile puudub juurdepääs, siis puudub ka võimalus liikumise tuvastuse algoritmi muuta või parendada.

Lisaks hüppavad Motion Spy Video Recorder ja Camera Trigger programmides tihti ette ka kasutamist segavad reklaamid.

Seega ei õnnestunud töö autoril leida sobiva funktsionaalsusega videot salvestavat rakendust ning ainsaks võimaluseks on sobiv rakendus ise luua.

Loodava rakenduse nõuded:

- Rakendus on püsivalt video salvestamise režiimis, et vältida aeganõudvat režiimi vahetust ja võimaldada väikse hulga viimaste kaadrite kaadripuhvris hoidmist.
- Salvestamine algab liikumise tuvastamisel.
- Salvestamise alguses pannakse kõigepealt video algusse kaadripuhvris olevad kaadrid, tagades niiviisi liikumise kadudeta salvestamise.
- Salvestamine lõpetatakse liikumise lõppemisel.
- Kõik salvestised liidetakse kokku üheks videofailiks.
- Toetatav video resolutsioon ja kaadrite sagedus peaks olema piisavad, et salvestisel olevad objektid oleks eristatavad ja saaks teha lihtsamat liikumise analüüsi.
- Rakenduse kasutamine peaks olema kasutajale lihtne.
- Liikumise tuvastamise algoritmi tundlikkus oleks lihtsasti häälestatavad vastavalt hetke vajadustele.

Järgnev töö annab neile nõuetele vastava rakenduse loomiseks minimaalsed vajalikud teoreetilised teadmised ja kirjeldab rakenduse loomise protsessi.

2. Liikumistuvastus ja andmete esitus

Järgnevas peatükis vaatleme lihtsamaid liikumise tuvastamise meetodeid ning kirjeldame NV21 pildi formaati, mis on Android platvormil vaikimisi pildi eelvaate formaadiks.

2.1 Liikumise tuvastamine

Liikumise tuvastus on protsess, kus tuvastatakse objekti asukoha muutust ümbritseva suhtes [2].

Reaalelus ei ole liikumise tuvastamine aga lihtne, sest tihti esineb taustal müra, eriti õues (nt puude liikumine tuule käes). Müra tekitab ka digitaalse kaamera sensor.

Järgnevalt saab lühidalt välja toodud liikumise tuvastamisel kasutatavad meetodid ja lähemalt vaatleme lihtsaimat neist.

2.1.1 Liikumise tuvastamiseks kasutavad meetodid

Põhilised liikumise tuvastamiseks kasutavad meetodid:

- Pikslite lahutamine - operatsioon, mis lahutab ühe pildi igast pikslivääртusest teise pildi vastava piksli väärtuse. [3]
- Servatuvastus - meetod, mis identifitseerib punktid pildis, kus pildi heledus muutub järsult. Tulemuseks on enamasti objektide piirjooned. Eesmärgiks on leida pildilt olulised muutuste kohad. [4]
- Optiline voog - näilise liikumise muster, mida saab esitada liikumise vektorite hulgana [5]. Optilise voo abil on võimalik tuvastada üldisest liikumismustrist erinevat liikumist. Näiteks üle autotee jooksvat last, võimaldades autode kulgemist liikumise tuvastamisel kõrvale jätta.

2.1.2 Liikumise tuvastamine pikslite lahutamise teel

Kahe pildi pikslite lahutamise tulemuseks on pikslite hulk, mis on saadud piltide vastavate pikslite väärtuste lahutamise teel.

Kui kahe erineva ajahetke kaadri pikslite lahutamise tulemuses on nullist erinevaid piksleid rohkem, kui etteantud pildi künnis ε, siis ütleme, et antud ajavahemikul toimus oluline liikumine. Pildi künnist ε võib esitada ka kui koefitsienti pildi pikslite arvust. Sel juhul on ε reaalarv vahemikus (0;1).

Kahe erineva pildi pikslid pole aga kunagi täpselt samasugused. Ebaoluliste pikslimuutuste kõrvalejätmiseks kasutatakse kahe piksli lahutamisel ebaolulise muutuse künnist. See piksli künnis ε_p näitab, kui suur peab olema piksli muutus, et seda peetaks liikumise tuvastamisel oluliseks. Piksli künnist ε_p võib esitada ka kui koefitsienti piksli maksimaalsest väärtusest. Sel juhul on ka ε_p reaalarv vahemikus (0;1).

2.2 YUV värviruum ja NV21 formaat

NV21 on YUV värviruumi formaat, mis Android platvormil on vaikimisi seatud kaameralt saadavate kaadrite formaadiks. Ühtlasi on see ka ainuke formaat, mida Android 2.1 ühilduvus definitsiooni [6] järgi kõik Android seadmed toetama peavad. Tihti ongi see ka ainuke seadme poolt toetatav formaat. Ka töö autori nutitelefon Samsung Galaxy W GT-I8150 toetab vaid NV21 formaati.

2.2.1 YUV värviruum

YUV [7] on värviruum pildi ja video kodeerimiseks. Tegemist on värviruumiga, kus heleduse ja värviinfo on teineteisest eraldatud. Y määrab heleduse, U ja V määravad värvi.

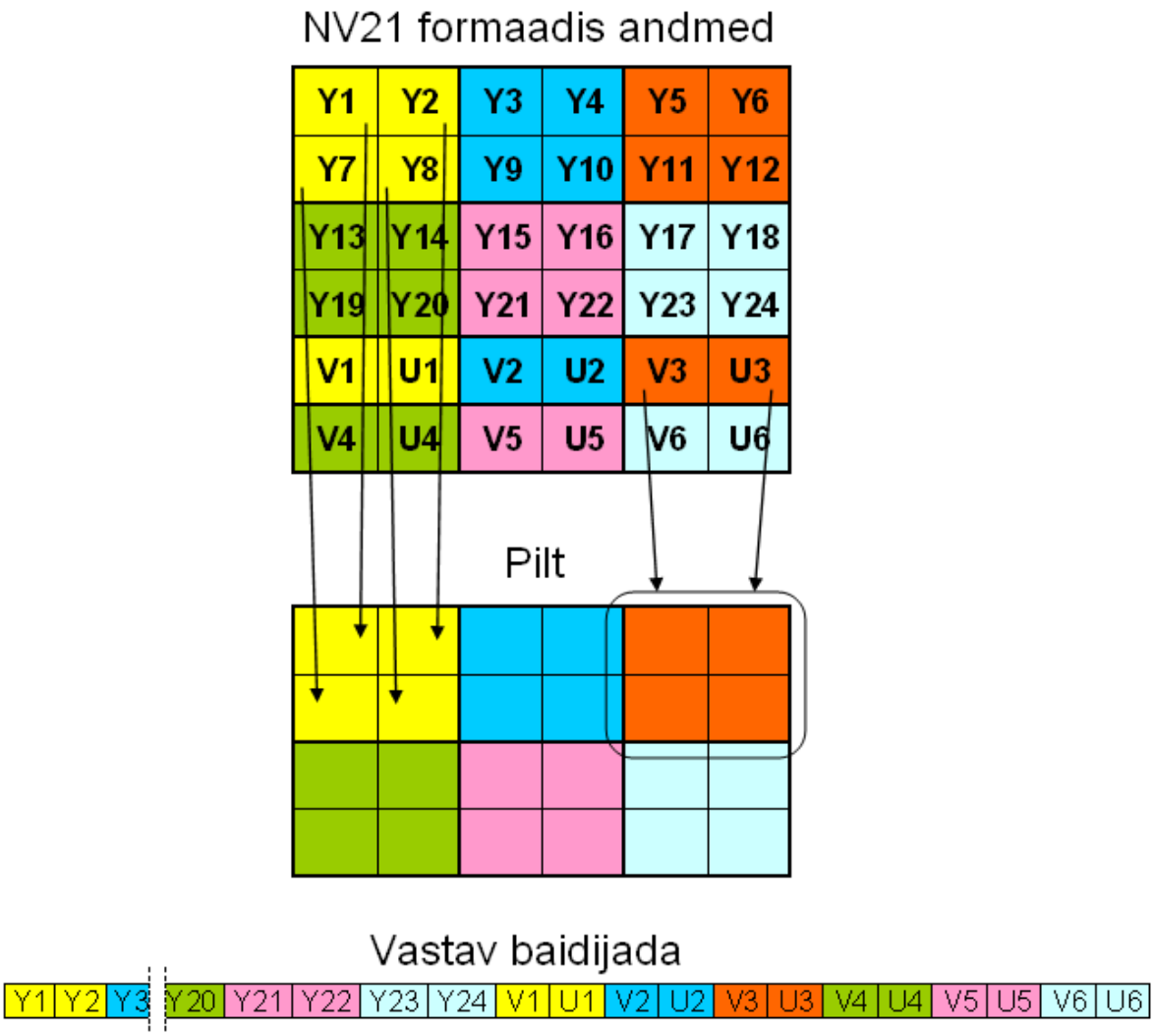
Inimese silma võrkkestal on kahte tüüpi rakke [8]:

- kepikesed – reageerivad valgusele ja liikumisele (150 miljonit rakku);
- kolvikesed – tajuvad värvi (7 miljonit)

Seega on inimese silma eraldusvõime värvuste suhtes palju väiksem kui heleduse suhtes. Tänu sellele võimaldab YUV kodeering hoida pildi edastamisel ja salvestamisel kokku mahus, jättes alles ainult osa värvi infost, ilma et see oluliselt pildi kvaliteeti mõjutaks.
YUV värviruumi pildi hea omadusena on võimalik seda pilti lihtsalt must-valgeks pildiks teisendada, jättes alles vaid Y komponendi ning eemaldades värvi määrava osa.

2.2.2 NV21 formaat

NV21 [9] formaadis pildi andmed koosnevad $pikkus * laius * 3/2$ baidist. Esimesed $pikkus * laius$ baiti on Y tasandi väärtused, kus on reserveeritud 1 heledust kirjeldav bait iga piksli kohta. Järgmised $(pikkus / 2) * (laius / 2) * 2 = pikkus * laius / 2$ baiti moodustavad V/U tasandi. V/U tasandil tulevad vaheldumisi V ja U väärtused, kumbki 1 bait. Iga V-U paar määrab $2 * 2 = 4$ algse piksli värvi.



Joonis 1: NV21 formaadis pildi andmete paigutus ning andmete ja pildi pikslite vastavus.

NV21 formaadis olevat pilti on lihtne teisendada must-valgeks, selleks tuleb faili andmetest alles jätta vaid esimesed *pikkus* * *laius* baiti. Kuna pildilt värvide eemaldamisel jäävad kujundid pildil endiselt nähtavaks, siis on must-valge pilt liikumise tuvastuse analüüsiks piisav. Seetõttu on NV21 formaat mugav lähteformaat liikumise tuvastamise realiseerimisel.

3. Rakenduse loomine

Järgnevas peatükis on kirjeldatud autori poolt väljapakutud video salvestamise võimalusi ning käesoleva töö raames valminud rakenduse Instant Motion Recorder teostust.

3.1 Loodud rakendus Instant Motion Recorder

Käesoleva töö raames on Android versioonile 2.3 loodud rakendus Instant Motion Recorder, mis võimaldab liikumise tuvastuse järgselt kadudeta video salvestamist ning liikumist sisaldavate videode automaatset üheks liitmist. Android versioon 2.3 on valitud sellepärast, et töö autori telefonil on just see versioon.

Rakenduse loomisel valis antud töö autor programmeerimiskeeks Java, kuna see on autorile tuttavam kui nt C/C++.

Rakendust on testitud järgmistel seadmetel:

- Samsung Galaxy W GT-I8150, CPU 1.4 GHz Scorpion, 512 MB RAM, Android versioon 2.3.6
- Samsung GALAXY S III Mini GT-I8190N, 1 GB RAM, CPU 1 GHz dual-core Cortex-A9, Android versioon 4.1.2
- Sony Ericsson Xperia neo V MT11i, 512 MB RAM, CPU 1 GHz Scorpion, Android versioon 4.0.4

Kõigil testitud seadmetel töötas rakendus video resolutsiooniga 640x480, saades kaadrite sageduseks 10-17 kaadrit sekundis. Salvestatud video kvaliteet oli piisav, et teha lihtsamat liikumise analüüsi. Video resolutsioonil 1280x720 hakkas aga kaadripuhver kiiremini täituma kui kaadreid videosse salvestada jõuti ja rakendus sai mälu täitumise tõttu ootamatu tõrke ning sulgus.

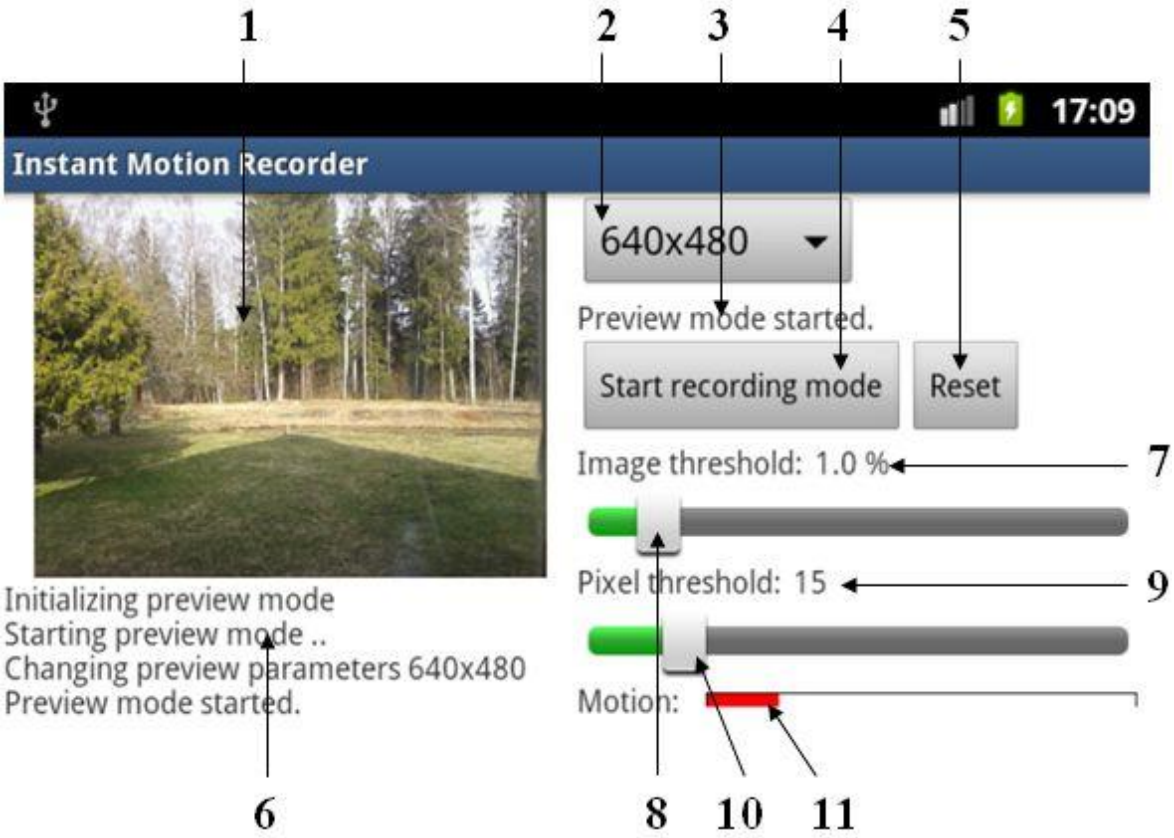
3.1.1 Kasutajaliides

Kasutajaliidese testimisel kasutatud ekraaniresolutsioonid:

- 480 x 800
- 480 x 854

Lihtsuse mõttes kasutatakse kasutajaliideses püsivalt horisontaalpaigutust. Kuna video salvestamine toimub enamasti just horisontaalvaates, siis kasutajamugavust see oluliselt ei mõjuta, kuid rakenduse arendamist lihtsustab märgatavalt.

Rakenduse kasutajaliides on välja toodud joonisel 2.



Joonis 2: kasutajaliidese ekraanitõmmis.

Kasutajaliidese komponentide selgitused:

1. Eelvaate ekraan – näitab kaamera poolt edastatavat pilti, mida uuendatakse iga 0,5 sekundi järel. Salvestamise režiimis pilti ei uuendata, et jätta rohkem protsessoriressurssi kaadrite analüüsimiseks ja video salvestamiseks.

2. Resolutsiooni valik – kuvatud kõik antud telefoni poolt toetatud salvestamise resolutsioonid.
3. Infotekst – kuvab infoteksti, milleks võib olla nt kaamera hetkesaatus või vea korral veateade.
4. Salvestamise režiimi käivitamise/peatamise nupp – käivitab/peatab salvestamise režiimi.
5. Vaikeseadete taastamise nupp – taastab liikumistuvastuse peenhäälestuse liugurite vaikeoleku.
6. Sündmuste ja vealogi – paneel, kus kuvatakse sündmuste logi ning vealogi.
7. Pildi künnise väärtus – pildi künnise liuguriga määratud pildi künnise arvuline väärtus.
8. Pildi künnise liugur – võimaldab peenhäälestada liikumise tuvastamise algoritmi poolt kasutatavat pildi künnist (*image threshold*).
9. Piksli künnise väärtus – piksli künnise liuguriga määratud piksli künnise arvulibe väärtus.
10. Piksli künnise liugur – võimaldab peenhäälestada liikumise tuvastamise algoritmi poolt kasutatavat piksli künnist (*pixel threshold*).
11. Liikumise indikaator – näitab visuaalselt, kui suur osa pildi künnisest on saavutatud. Nt kui liikumise tuvastamisel on muutunud pikslite arv 50% ette antud pildi künnisest, siis on indikaator poole peal.

3.2 Video salvestamine

Soovitud tulemuse saavutamiseks on video salvestamisel kaks võimalust. Üks võimalustest on salvestada video ja töödelda see sobivale kujule pärast salvestamise lõppu. Teine võimalus on tegeleda video töötlemisega salvestamise ajal paralleelselt eraldi lõimes.

3.2.1 Video töötlemine pärast salvestamise lõppu

Üheks võimaluseks rakenduse realiseerimisel oleks kasutada käivitatakse Androidi sisseehitatud kaamerarakendust, kus kasutaja saab algatada ja peatada video salvestamise. Kaamerarakendus tagastab sel juhul juba tihendatud video asukoha failisüsteemis. Seejärel

asutakse tagastatud videot töötlema, analüüsides kaadreid ning liikumistuvastuse põhjal lõigatakse salvestisest välja sobivad tükid. Seejärel liidetakse tükid kokku üheks videoks, kasutades näiteks FFmpeg raamistikku [10].

Lähenemise plussid:

- Video salvestamist on lihtne realiseerida.
- Kasutatav standardne kaamera rakenduse kasutajaliides on rakenduse kasutajale tuttav.
- Salvestamine on hästi optimeeritud ning video salvestamisel saab kasutada suurt resolutsiooni ja kaadrisagedust.

Lähenemise miinused:

- Salvestamisjärgne kaadrite analüüs ning töötlus on ressursimahukas ja aeganõudev.
- Esialgne video tuleb mahult suur, kuna sisaldab palju staatilisi kaadreid.

3.2.2 Video töötlemine salvestamise ajal

Teine variant on viia analüüs läbi reaalajas. Kui kaameralt saadakse kätte video kaadrid, siis tehakse koheselt liikumise tuvastamiseks kaadrite analüüs ning kaadritest luuakse sobivas formaadis video fail. Kaadritest video salvestamine toimub läbi FFmpeg raamistiku koheselt.

Kaadrite kohesel töötlemisel on järgmised võimalused:

- Tihendamata video saadetakse MediaRecorder klassi [11] abil stream'i, mida teisest otsast loetakse ning tehakse kaadrite analüüs ja salvestatakse video.
- Läbi Camera.PreviewCallback liidese [12] saadakse kätte kaadrid. Tehakse kaadrite analüüs liikumise tuvastamiseks ning liikumise esinemisel salvestatakse kaadrid videoks.

Lähenemise plussid:

- Kogu andmete töötlus toimub salvestamise ajal ja ajakulukat järeltöötlust ei ole vaja. Videot saab pärast salvestamise lõpetamist koheselt vaatama hakata.
- Andmekandjale salvestatakse ja tihendatakse ainult lõpptulemusse minev osa videost.

Lähenemise miinused:

- Ei saa kasutada sisseehitatud kaamerarakendust, mis on kasutajale tuttav ja hästi optimeeritud.
- Olenevalt nutitelefoni jõudlusest tekib piirang salvestatava video resolutsiooni suurusele ja kaadrite sagedusele.
- Esialgsel hinnangul keerulisem teostada.

3.2.3 Video salvestamine loodavas rakenduses

Kuna antud töö autor pidas töö raames tehtava rakenduse loomisel oluliseks head kasutajamugavust, siis otsustas ta rakenduse loomisel kasutada video salvestamiseks viisi, kus kaadrite töötlemine toimub koheselt.

Kuna esmaste katsetuste käigus tundus MediaRecorder klassi kasutamine töö autorile keerulisem kui Camera.PreviewCallback liidese kasutamine, siis otsustas töö autor kasutada just Camera.PreviewCallback liidest.

Camera.PreviewCallback liidese kasutamisel kaadrite kättesaamiseks on oluline kasutada setPreviewCallbackWithBuffer meetodit, mis kasutab kaadri andmete edastamisel taaskasutatavat mälupuhvrit ja võimaldab seeläbi hõivatud mälu taaskasutamist. Kuna nii ei toimu pidevalt uue mälu reserveerimist, siis on kaadrite edastamine kiirem.

Liikumise tuvastamine toimub iga 0,5 sekundi järel. Et kahe tuvastamise vahel saadud kaadrid kaduma ei läheks, siis hoitakse neid kaadreid puhvris ja liikumise tuvastamisel salvestatakse puhvrist videosse. Antud puhver on realiseeritud BlockingQueue klassi [13] objektina. Tegelikult hoitakse puhvris 1,5 sekundi jagu kaadreid, sest katsetamise käigus

tundus nii video kõige loomulikum ja liidetud videolõikude vahele jäi piisav paus. Loomulikult võib puhvri suuruse teha seadistatavaks, aga rakenduse esmases versioonis on see seatud konstantseks.

Kaadrite salvestamine videoks toimub kasutades FFmpeg raamistiku. Salvestamine toimub eraldi lõimes. Kaadrite info saadetakse salvestavale lõimele läbi BlockingQueue klassi objekti. Kaadrite lisamine järjekorda ja salvestamiseks eemaldamine käib FIFO meetodil.

3.3 Tehniline teostus

Järgnevalt tuuakse välja mõningad rakenduse loomisel tehtud olulisemad otsused ja tehnilised nüansid.

3.3.1 JavaCV ja FFmpeg

Rakenduses on kasutusel JavaCV teek [14], mis sisaldab muuhulgas ka video salvestamiseks vajalikku FFmpeg raamistikku.

FFmpeg on üks juhtivaid multimeedia raamistikke, mis võimaldab muuhulgas ka salvestada ja teisendada audio- ning videofaile. FFmpeg sisaldab libavcodec teeki [15], mis on üks tuntumaid teeke heli ning video kodeerimiseks ja dekodeerimiseks.

Antud töö raames loodud rakenduses kasutatakse FFmpeg raamistikku kaadritest video loomiseks. FFmpeg võimaldab video salvestamist ka muutuva kaadrite sagedusega. Selleks antakse video salvestamisel iga kaadriga kaasa ka ajatempel, mis määrab kaadri ajalise asukoha videos.

FFmpeg võimaldab ka hääle salvestamist, kuid rakenduse Instant Motion Recorder esialgses versioonis seda ei realiseeritud.

3.3.2 NV21 formaat ja liikumise tuvastamine

Lihtsuse mõttes on antud töös eeldatud, et kasutaja ei hakka rakendust kasutama suurte liikuvate objektide läheduses nt maantee ääres, kus kaamera ette jäävad ka liikuvad autod, mis kasutajat tegelikult ei huvita. Antud rakendus peab oluliseks suvalist liikumist, mis pilti suuremal määral mõjutab.

Loodavas rakenduses otsustas autor kasutada liikumise tuvastamiseks pikslite lahutamise meetodit, sest see on teostatavuselt kõige lihtsam ja nõuab kõige vähem protsessori ressurssi. Kahe erineva kaadri pikslite lahutamiseks peab aga olema võimalik esitada piksleid arvudena.

Antud töö raames loodud rakendus saab Camera.PreviewCallback liidese kaudu NV21 formaadis pildi. Lihtsuse mõttes kasutatakse rakenduses liikumise tuvastamiseks pikslite lahutamisel ainult Y-komponenti ehk siis vaatleme must-valgeid pilte. Android platvormil on NV21 formaadis Y-komponentide väärtused täisarvud vahemikus 0-255. Selliseid väärtusi on lihtne omavahel lahutada. ε ja εp väärtused on rakenduses kasutajale muudetavad, võimaldades kasutajal iseseisvalt liikumise tuvastust peenhäälestada. Vaikimisi on rakenduses pildi künnise ε väärtuseks 0,02 ja piksli künnise εp väärtuseks 15. Seega loetakse pikslit muutunuks, kui pikslite lahutamise tulemus on vähemalt 15 ja liikumine loetakse toimunuks, kui vähemalt 2% pikslitest on muutnud.

Rakenduse antud versioonis kasutatakse liikumise tuvastamist samas lõimes, kus toimub kaameralt kaadrite saamine. Seetõttu on sujuva video saamiseks väga oluline võimalikult väike liikumise tuvastamise ajaline kestvus, eriti väiksema jõudlusega nutitelefonide puhul. Selleks on rakenduses optimeerimise mõttes vaatluseta jäetud iga teise rea ja teise veeru pikslid. Seega analüüsitakse liikumise tuvastamisel vaid ¼ pikslitest.

3.3.3 Lõimed

Kasutajaliidese mugavamaks kasutamiseks on rakenduses kasutajaliidese lõimele lisaks veel kaks lõime. AsyncTask'ina [16] realiseeritud lõimes on realiseeritud Camera.PreviewCallback liidese kaudu kaadrite kinni püüdmine. Samuti edastab see lõim kasutajaliidese lõimele liikumise indikaatori kuvamiseks vajalikud andmed. Sellest lõimest on omakorda algatatud uus lõim, mis tegeleb vaid puhvris olevate kaadrite videoks salvestamisega. Ka see lõim on realiseeritud AsyncTask'ina.

3.4 Võimalikud edasiarendused

Rakenduse valmimise käigus on autoril tekkinud erinevaid mõtteid rakenduse parendamiseks järgmistes versioonides, olulisemad neist:

- Leida võimalus video salvestamise optimeerimiseks, et ka suurematel video resolutsioonidel oleks kaadrite sagedus piisav. Üks proovitavatest võimalustest võiks olla GStreamer [17] kasutamine.
- Leida võimalus rakenduse paigalduse suuruse vähendamiseks. JavaCV on üksi juba üle 50 mb suur, kuid lähtekood on saadaval, nii et on võimalik antud rakenduse jaoks ebavajalik funktsionaalsus välja jätta.
- Kontrollida kaadrite mälupuhvrisse panekul saadaval oleva mälu hulka ja vajadusel tühjendada osa puhvrist.
- Lisada seadete leht, kuhu ümber tõsta resolutsiooni valik ning kus saaks muuta kaadrite mälupuhvri suurust.
- Muuta rakenduse kasutajaliidest mugavamaks ja visuaalselt ilusamaks.
- Võimaldada ekraanil määrata liikumise tuvastamisel analüüsitavat ala.

Kokkuvõte

Käesoleva töö eesmärgiks oli luua liikumistuvastusega kadudeta videot salvestav mobiilirakendus Android platvormile ning saada seeläbi ka mobiilirakenduse loomise kogemus.

Töö esimeses osas anti lühiülevaade Google Play poes pakutavatest sarnastest rakendustest ning nende puudustest. Seejärel tutvustati Android platvormil kasutatavat pildiformaati NV21 ning liikumist tuvastavaid algoritme, keskendudes pisut rohkem neist lihtsamaile. Viimases osas kirjeldati pikemalt antud töö käigus loodud rakendust Instant Motion Recorder.

Rakenduse funktsionaalsus realiseeriti algselt soovitud mahus, kuid lahendamist vajavad jõudlusprobleemid, et ka suurematel resolutsioonidel oleks salvestatud videopilt piisavalt sujuv. Töö autoril on plaanis jätkata antud rakenduse parendamist ning arendamist. Kui lisaks jõudlusprobleemidele parandada ka kasutajaliidese väljanägemist ja kasutajamugavust, siis töö autori hinnangul võiksid antud rakenduse vastu huvi tunda ka teised Android platvormil baseeruvate nutitelefonide kasutajad.

Lossless video recording using motion sensor on Android platform

Bachelor's thesis

Tarmo Paavel

Summary

The goal of this thesis is to describe why and how a video recording application Instant Motion Recorder was created for Android platform.

The first part of the thesis contains small overview of similar video recording applications and their imperfections.

The second part gives quick overview of NV21 format and motion detection algorithms focusing on the simplest one.

The third part is describing mobile application Instant Motion Recorder. It is a Java based application that can detect motion. It starts video recording when motion is detected and stops when motion is stopped. The application also enables lossless video recording. To achieve this goal it has a frame buffer. The content of the buffer is saved at the beginning of the video when motion is detected and video recording is started.

The application contains the desired functionality but it also has performance problems when using larger resolutions and that needs to be dealt with before the application can be published in Google Play store. The author of this thesis believes that when performance problems will be solved and the application gets a nicer user interface it would also interest other Android users.

Viited

Kõik veebiviited on vaadatud seisuga 13 mai 2014.

[1] Google Play. *Pood*
https://play.google.com/store
[2] Wikipedia. *Motion detection*
http://en.wikipedia.org/wiki/Motion_detection
[3] Wikipedia. *Image subtraction*
http://en.wikipedia.org/wiki/Image_subtraction
[4] Wikipedia. *Edge detection*
http://en.wikipedia.org/wiki/Edge_detection
[5] Wikipedia. *Optical flow*
http://en.wikipedia.org/wiki/Optical_flow
[6] Android Compatibility Program. *Android 2.1 Compatibility Definition*
http://source.android.com/compatibility/2.1/android-2.1-cdd.pdf
[7] Wikipedia. *YUV*
http://en.wikipedia.org/wiki/YUV
[8] Prillidest priiks. *Silma ehitus*
http://prillidestpriiks.weebly.com/silm-ja-naumlgemine.html
[9] Microsoft. *4:2:0 Video Pixel Formats*
http://msdn.microsoft.com/en-us/library/windows/hardware/ff538197%28v=vs.85%29.aspx
[10] FFmpeg. *About FFmpeg*
http://www.ffmpeg.org/about.html
[11] Android Developers. *MediaRecorder*
http://developer.android.com/reference/android/media/MediaRecorder.html
[12] Android Developers. *Camera.PreviewCallback*
http://developer.android.com/reference/android/hardware/Camera.PreviewCallback.html
[13] Android Developers. *BlockingQueue*
http://developer.android.com/reference/java/util/concurrent/BlockingQueue.html
[14] GitHub. *JavaCV*

https://github.com/bytedeco/javacv

[15] Wikipedia. *libavcodec*

http://en.wikipedia.org/wiki/Libavcodec

[16] Android Developers. *AsyncTask*

http://developer.android.com/reference/android/os/AsyncTask.html

[17] Gstreamer. *What is GStreamer?*

http://gstreamer.freedesktop.org/

Lisad

I. Loodud rakenduse lähtekood

<lisatud digitaalselt>

II. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina **Tarmo Paavel** (sünnikuupäev: 08.03.1978)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

   Liikumistuvastusega kadudeta video salvestamine Android seadmel,

   mille juhendaja on **Margus Niitsoo**,

   1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

   1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, seal hulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **14.05.2014**