KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

Karolis Kriščiūnas

Objektų sekimo algoritmų tyrimas ir taikymas vaizdo atpažinimo sistemoje

Magistro darbas

Darbo vadovas

prof. E. Bareiša

Kaunas, 2012

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

Karolis Kriščiūnas

Objektų sekimo algoritmų tyrimas ir taikymas vaizdo atpažinimo sistemoje

Magistro darbas

Recenzentas

doc. A. Lenkevičius

2012-05-21

Vadovas

prof. E. Bareiša

2012-05-21

Atliko

IFME-0/2 gr. stud.

Karolis Kriščiūnas

2012-05-21

Kaunas, 2012

PAVEIKSLĖLIŲ SĄRAŠAS

OBJEKTŲ SEKIMO ALGORITMŲ TYRIMAS IR TAIKYMAS VAIZDO ATPAŽINIMO SISTEMOJE

SANTRAUKA

Vizualių objektų sekimas yra vienas iš svarbiausių kompiuterinės regos uždavinių. Šiam uždaviniui spręsti yra sukurta nemažai įvairių algoritmų ir technologijų. Šiame darbe analizuojamas objekto sekimo uždavinys ir objektų sekime iškylančios problemos. Aptariamos trys objektų sekimo algoritmų grupės; du algoritmai - optinio srauto Lucas - Kanade ir CAMShift - nagrinėjami detaliai.

Taip pat darbe pristatomas šiuos du algoritmus apjungiantis objektų sekimo metodas, realizuotas sukurtoje vaizdo atpažinimo sistemoje. Metodas kiekvienam atpažintam objektui parenka tinkamesnį algoritmą. CAMShift yra parenkamas vienspalviams objektams, o Lucas - Kanade - spalvotiems. Pradiniam objekto aptikimui kadre naudojama SURF algoritmas. Jei sekimui parenkamas Lucas - Kanade metodas, SURF algoritmas naudojamas sektinų taškų apskaičiavimui. Jei pametamas kritinis sektinų taškų skaičius, taikomas taškų tikslinimas naudojant atpažinto objekto modelį.

Darbe siūlomas metodas vaizdo atpažinimo sistemoje be sudėtingo išankstinio apmokymo leidžia sekti tiek vienspalvius, tiek daugiaspalvius objektus, objekto sekimas tampa patikimesnis ir labiau atsparus objekto uždengimams.

"RESEARCH AND APPLICATION OF OBJECT TRACKING ALGORITHMS IN VISUAL RECOGNITION SYSTEM"

SUMMARY

Visual object tracking is one of the most important tasks of computer vision. Many various algorithms and technologies have been developed for this task. In this thesis, task of object tracking and difficulties of it are described. Three groups of object tracking algorithms are discussed, while two of the algorithms - Optical Flow Lucas - Kanade and CAMShift - are studied in detail.

The combined method of these two algorithms, which is implemented in visual recognition system, is then presented. The method dynamically selects more suitable algorithm for each recognized object. CAMShift is being selected for single-color objects, while Lucas - Kanade best suits for tracking multi-colored objects. SURF algorithm is used for initial object detection. If Lucas - Kanade method is selected, SURF algorithm is used for calculating points to track. If critical number of tracking points is lost, correction of points from recognized object model is applied.

Proposed method enables recognition system to track both single-color and multi-color objects without initial learning, makes tracking more robust and occlusions-proof.

TURINYS

1. ĮVADAS

Paskutiniu metu, paplitus pakankamai galingiems kompiuteriams bei labai greitai populiarėjant mobiliesiems įrenginiams su aukštos raiškos vaizdo kameromis, vaizdo atpažinimo technologijos tampa viena svarbiausių kompiuterinės regos sričių. Šios technologijos plačiai taikomos pramonėje, robotikoje, viešosios tvarkos užtikrinimui, kariniais tikslais. Taip pat kartu su mobiliaisiais įrenginiais jos vis labiau populiarėja kasdieniniame gyvenime.

Tačiau daugumai vaizdo atpažinimo technologijų taikymų nepakanka vien tik atpažinti kameros regos lauke esantį objektą ar žmogų. Pavyzdžiui, praturtintosios realybės (angl. *augmented reality*) sistemoje, kuri ekrane ant aptikto objekto atvaizduoja virtualų vaizdą, svarbu įvertinti aptikto objekto pozicijos kitimą erdvėje. Vaizdo kamerų sekimo (angl. *surveillance*) sistemose taip pat dažnai reikia ne tik identifikuoti objektą, tačiau ir stebėti jo elgseną laike. Taigi, minėtiems, kaip ir daugumai kitų taikymų, itin svarbus atpažintų objektų sekimas.

Yra sukurta nemažai algoritmų, skirtų objektų sekimui. Vieni iš jų skirti tik tam tikro tipo objektų sekimui ir yra tinkami tik siauriems taikymams, kiti yra universalesni. Kai kuriems algoritmams reikia išankstinio apmokymo, kiti seka objektą be išankstinių duomenų ar mokosi realiu laiku. Objektų sekimui naudojami MeanShift, CamShift, optinio srauto, RANSAC ir kiti algoritmai. Praturtintosios realybės sistemose, skirtose išmaniesiems telefonams objektų sekimui skirti algoritmai neretai apjungiami su mobiliojo įrenginio judesio sensoriumi [14].

Šiame darbe detaliai nagrinėjama objekto sekimo uždavinys, bei pristatomas modifikuotas objekto sekimo algoritmas, kuris yra realizuotas magistratūros studijų metu sukurtoje universalioje vaizdo atpažinimo sistemoje.

Antrame darbo skyriuje pristatoma objektų sekimo uždavinio formuluotė bei pateikiami taikymų pavyzdžiai. Apžvelgiama populiariausi objektų sekimo algoritmai, bei detaliai pristatomi du iš jų.

Trečiame skyriuje glaustai pristatoma sukurta universali vaizdo atpažinimo sistema bei joje panaudotas jungtinis objektų sekimo algoritmas.

Penktame skyriuje pateikiama realizuoto objektų sekimo algoritmo eksperimento aprašymas ir rezultatai.

2. OBJEKTO SEKIMO UŽDAVINIO IR ALGORITMŲ ANALIZĖ

2.1. Objekto sekimo samprata ir taikymai

Objekto sekimo uždavinys gali būti apibrėžtas kaip paveikslėlių sekos plokštumose judančio objekto įvertinimas. Kitaip sakant, objekto sekimo algoritmas kiekviename iš kameros ar kito vaizdo srauto gautame kadre nustato sekamo objekto koordinates. Be to, priklausomai nuo taikymo srities, objekto sekimo algoritmai taip pat suteikia papildomos informacijos apie objektą – nustatomas jo pasukimas erdvėje, plotas, forma ar judėjimo greitis.

Matematiškai objekto sekimą galima aprašyti taip. Jeigu I ir J yra du kadrai, iš kameros paimti atitinkamai laiku t ir $t+\Delta t$. $I(\mathbf{p}) = I(x,y)$ ir $J(\mathbf{p}) = J(x,y)$ yra paveikslėlių reikšmės taške $\mathbf{p}=(x,y)$. Tada, jei I paveikslėlio taškas $\mathbf{u}=(u_x,u_y)$ priklauso objektui, kurį norima sekti, tai objekto sekimo algoritmo tikslas yra J paveikslėlyje surasti tokį tašką $\mathbf{v}=\mathbf{u}+\mathbf{d}=(u_x+d_x, u_y+d_y)$, kad $I(\mathbf{u})$ ir $J(\mathbf{v})$ reikšmės būtų panašios ir taškas $\mathbf{v}$ priklausytų sekamam objektui. Čia $\mathbf{d}$ yra taško $\mathbf{u}$ pokytis per laiko tarpą Δt.

Objekto sekimo uždavinį galima suskirstyti į tris dalis:

1. Dominančio objekto aptikimas / atpažinimas
2. Objekto koordinačių pradiniame taške nustatymas
3. Dominančio objekto koordinačių ir/arba papildomos informacijos nustatymas sekančiuose kadruose.

Priklausomai nuo taikymo srities, dominantis objektas aptinkamas įvertinus judesį kadre (pradedama sekti didžiausią judėjimo greitį turintis objektas), aptikus objektą pagal iš anksto žinomą vizualų požymį (spalvą, kampus, kontūrą) arba atpažinus objektą ir nustačius jo koordinates naudojantis įvairiais vietinių požymių (angl. *features*) algoritmais.

Objekto sekimas yra sudėtingas dėl keleto priežasčių:

1. Dėl informacijos praradimo projektuojant 3D pasaulio vaizdą į 2D paveikslėlį.
2. Dėl kadre esančių triukšmų.
3. Dėl sudėtingo objekto judėjimo.
4. Dėl objekto gebėjimo keisti formą.
5. Dėl dalinio ar visiško objekto uždengimo tam tikru laiko momentu.
6. Dėl sudėtingų objekto formų.
7. Dėl sekamo objekto ar jo aplinkos apšvietimo pasikeitimų.
8. Objekto sekimo algoritmai turi veikti realiu laiku.

Objekto sekimo uždavinys gali būti supaprastintas įvedant judesio, ar objekto išvaizdos apribojimus. Pavyzdžiui, dauguma algoritmų daro prielaidą, jog objekto judėjimas yra lygus, be staigių pasikeitimų. Taip pat kai kurie algoritmai įveda apribojimą, kad objekto judėjimo greitis arba pagreitis yra nekintamas. Neretai algoritmuose naudojama išankstinė informacija apie sekamus objektus - jų forma, spalva, kiti vizualūs požymiai.

Objektų sekimo algoritmai gali būti taikomi daugelyje įvairių sričių. Keletas iš galimų taikymų yra:

- Robotikoje. Kompiuterinė rega ir objektų sekimas robotikoje gali būti naudojamas roboto judėjimo trajektorijos apskaičiavimui. Pvz., robotas seka judantį objektą, arba objekto sekimo algoritmas taikomas siekiant išvengti roboto susidūrimo su judančiomis kliūtimis.
- Video stebėjimo sistemose neįprastai veiklai nustatyti ar stebėti ir registruoti tam tikrų judančių objektų bei žmonių elgseną laike.
- Praturtintosios realybės sistemose siekiant virtualų vaizdą atvaizduoti tiksliai ant norimo objekto, įvertinant jo judėjimą ir pasukimą.
- Žmogaus - kompiuterio sąveikos sistemose gestų atpažinimui ir sekimui [17], akies rainelės, žvilgsnio sekimui ir panašiai.
- Eismo stebėjimui, siekiant įvertinti eismo srautą

2.2. Panašios sistemos

Yra keletas į sukurtą vaizdo atpažinimo sistemą panašių projektų.

Viena iš populiariausių sistemų, labai panaši į magistratūros studijų metu sukurtą vaizdo atpažinimo sistemą yra „Google Goggles“. Tai kompanijos „Google“ sukurta nemokama programa, veikianti Android OS ir iOS operacinėse sistemose. Ši programa yra skirta ieškoti panašių paveiksliukų internete ir pateikti šių paveiksliukų aprašymus. Programos savybės:

- Žyminių kodų (barcodes) paieška.
- QR kodų paieška.
- Populiarių aplinkos vietų (landmarks) paieška.
- Užsienio kalbomis parašyto teksto vertimas.
- Teksto analizavimas, panaudojant OCR.

Tačiau ši sistema atpažintų objektų neseka.

SURFTrack [2] - kliento serverio architektūra pagrįsta sistema, skirta objektų atpažinimui ir sekimui. Sistema prie judančio objekto atvaizduoja etiketę su objekto aprašymu. Sistemos autoriai pristato algoritmą, skirtą nuolatiniam paveikslėlio atpažinimui ir požymių

deskriptorių sekimui. Vietoj sekimo 2D paveikslėliuose, sistemoje objekto kandidato paieška ir požymių palyginimas vykdomas 3D paveikslėlių piramidėje neskaičiuojant požymių deskriptorių. Sistemoje tiek objektų atpažinimas, tiek sekimas vykdomas realiu laiku.

[14] straipsnyje pristatomoje praturtintosios realybės sistemoje objektų atpažinimas taip pat vykdomas serveryje. Sistema geba atpažinti ir sekti tiek atviroje, tiek uždaroje erdvėje esančius objektus. Objektų sekimui mobiliojo telefono kliento pusėje naudojama vietiniai požymiai, o objekto pozicijos patikslinimui taikoma iš mobiliojo telefono jutiklio gauta informacija.

2.3. Sekamo objekto požymių išskyrimo analizė

Tinkamų vizualių objekto požymių parinkimas yra itin svarbus objekto sekimui tiek pradinės sekamo objekto pozicijos nustatymo atveju (objekto aptikimui), tiek sekamų taškų parinkimui ir tikslinimui. Viena iš svarbiausių vizualaus požymio savybių yra jo unikalumas – požymis turi būti išsiskiriantis iš kitų.

Unikalaus objekto požymio išskyrimą galima suskaidyti į tris pagrindinius žingsnius. Pirmiausiai, ryškiose ir skirtingose paveikslėlio vietose, tokiose kaip kampai, didesnės spalvos dėmės ir pan. pasirenkami dominantys taškai (požymių taškai). Svarbiausia požymių taškų išskyrimo savybė yra pakartojamumas. Tai reiškia, jog turi būti galimybė tuos pačius objekto požymių taškus aptikti skirtingomis sąlygomis. Taip pat požymių taškai turi būti nepriklausomi nuo objekto orientacijos erdvėje.

Kitas žingsnis - kiekvieno pasirinkto taško aplinkiniai taškai (aplinka) aprašomi požymių vektoriumi (deskriptoriumi). Šis vektorius turi būti savitas ir charakterizuojantis tašką, tačiau tuo pat metu turi būti kiek įmanoma nekintantis paveikslėlyje keičiantis triukšmo, apšvietimo lygiui bei atsparus geometrinėms objekto deformacijoms (pasukimui, suspaudimui ir pan.). Objektų atitikimų dviejuose skirtinguose paveikslėliuose suradimui kiekvieno paveikslėlio požymių deskriptoriai yra sulyginami. Požymių vektorių palyginimas paremtas atstumu tarp vektorių. Taigi palyginimo laiką tiesiogiai įtakoja deskriptoriaus dydis. Siekiant greitesnio aptikimo, pageidaujama, jog deskriptoriaus matmenys būtų kiek įmanoma mažesni. Kita vertus, mažesnių matmenų požymių vektoriai yra mažiau unikalūs.

Vienas iš populiariausių požymių išskyrimo ir palyginimo algoritmų, atitinkančių aukščiau minėtas sąlygas, yra SIFT algoritmas. Šis algoritmas požymių išskyrimui skaičiuoja taško aplinkoje esančių gradientų histogramą ir ją saugo 128D vektoriuje. Šis deskriptorius yra unikalus ir gana greitai apskaičiuojamas. Tačiau pagrindinis SIFT deskriptoriaus trūkumas yra jo daugiadimensiškumas. Dėl to atitikimų paieška užtrunka santykinai ilgai.

Siekiant pagreitinti palyginimo žingsnį H. Bay pristatė [8] SURF (angl. *Speeded-Up Robust Features*, Pagreitinti Patikimi požymiai) vietinių požymių išskyrimo ir palyginimo algoritmą. Toliau apžvelgiamas šis algoritmas.

2.3.1. SURF požymių algoritmas

SURF (angl. *Speeded-Up Robust Features*) [8][9] - tai nuo mastelio ir pasukimo nepriklausantis požymių išskyrimo ir deskriptorių sudarymo algoritmas. SURF pagal patikimumą, taškų pakartojamumą ir deskriptorių unikalumą prilygsta kitiems požymių išskyrimo algoritmams, tačiau SURF deskriptorius gali būti daug greičiau apskaičiuotas ir palygintas. Algoritmas paremtas integruotų paveikslėlių naudojimu paveikslėlio sąsūkų skaičiavimui, Hessian matricų naudojimu taškų aptikimui bei pasiskirstymu paremtais deskriptoriais.

Kaip ir dauguma vietinių požymių algoritmų, taip ir SURF susideda iš dviejų pagrindinių dalių - dominančių taškų aptikimo bei dominančių taškų deskriptorių apskaičiavimo ir sulyginimo.

Dominančių taškų aptikimas

SURF algoritmas dominančių taškų aptikimui naudoja integralinius paveikslėlius. Integralinio paveikslėlio reikšmė taške $\mathbf{x}=(x,y)^T$ atitika visų paveikslėlio I taškų, esančių stačiakampiame regione, sumą. Stačiakampis regionas sudaromas nuo pradinio taško iki **x.**

$$I_{\Sigma}(x) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j) \quad (2\text{-}1)$$

Apskaičiavus integralinį paveikslėlį, tam tikro regiono intensyvumo sumą galima paskaičiuoti vos trimis sudėties veiksmais. Taigi, intensyvumo apskaičiavimo laikas nepriklauso nuo paveikslėlio dydžio.

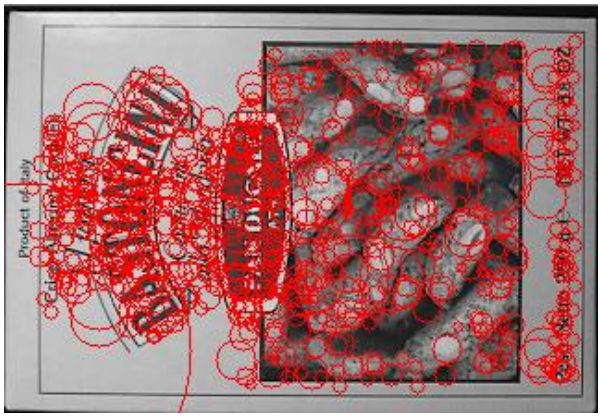
Turint integralinį paveikslėlį, dominančių taškų aptikimui SURF algoritmas taiko Hessian matricą. Naudojant šią matricą, vietose, kur determinantas yra maksimalus, nustatomos paveikslėlio dėmių (angl. *blob*) struktūros. Paveikslėlio I taške **x**=(x, y) ir masteliu σ Hessian matrica apibrėžiama 2-2 formule

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix} \quad (2\text{-}2)$$

Kur $L_{xx}(x,\sigma)$ yra Gauso antros eilės išvestinės $\frac{\partial^2}{\partial x^2} g(\sigma)$ sąsūka paveikslėlio I taške **x.**

Apskaičiuotas Hessian matricos determinantas atitinka tam tikros dėmės (angl. „blob“) ryškumą paveikslėlio taške **x**. Šie duomenys apskaičiuojami skirtingiems masteliams. Dominančių taškų nustatymui taikoma maksimumų mažinimas, tada Hessian matricos maksimalus determinantas interpoliuojamas mastelių atžvilgiu ir gaunamos dominančio taško koordinatės bei dydis.

1 paveikslėlyje pateikiamas SURF algoritmu aptiktų požymių pavyzdys, kuriame apskritimai atitinka algoritmo aptiktus taškus.

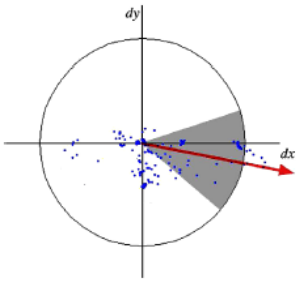


Paveikslėlis 1 SURF požymiai

Dominančių taškų deskriptorių apskaičiavimas ir palyginimas

SURF vieno aptikto požymio taško deskriptorius aprašo intensyvumo pasiskirstymą taško aplinkoje. Deskriptoriaus apskaičiavimo metu siekiant, jog deskriptorius būtų nepriklausomas nuo paveikslėlio pasukimo, pirmiausiai nustatoma dominančio požymio taško orientacija plokštumoje.

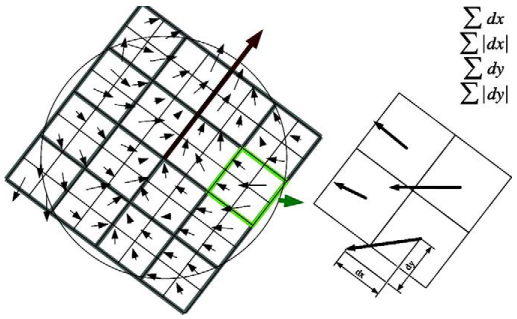
Taško orientacijos nustatymui apskaičiuojamas Haar bangų atsakas x ir y kryptimis aplink požymio tašką spinduliu 6s, kur s – mastelis, kuriame aptiktas požymis. Apskaičiuoti ir Gauso funkcija įvertinti Haar bangų duomenys atitinka taškus plokštumoje. Dominuojanti kryptis nustatoma susumuojant visus taškus, esančius slenkančio orientacijos lango, kurio dydis $\frac{\pi}{3}$, viduje (2 pav.).



Paveikslėlis 2 Požymo taško krypties vektoriaus nustatymas

Susumuojama horizontali ir vertikali reikšmės. Taip gaunamas lokalus orientacijos vektorius. Įvertinus visus taškus gautas didžiausias toks vektorius atitinka dominančio požymio taško orientaciją.

Tada sudaromas kvadratinis regionas, centruotas dominančio taško koordinatėse ir pasuktas apskaičiuoto vektoriaus kryptimi. Šis regionas padalinamas į mažesnius 4x4 taisyklingus kvadratus (subregionus) išsaugant svarbią erdvinę informaciją (3 pav.). Kiekvienam subregionui apskaičiuojami Haar bangų sumos d_x horizontalia ir d_y - vertikalia kryptimis pagal dominančio taško orientaciją. Tada kiekvieno subregiono d_x ir d_y reikšmės susumuojamos, taip gaunant pirmą rinkinį požymių deskriptoriaus vektoriuje. Siekiant įvertinti intensyvumo pokyčių poliariškumą, taip pat susumuojama ir d_x bei d_y absoliutinės reikšmės. Taigi, kiekvieną subregioną aprašo 4D deskriptoriaus vektorius $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$. Šiuos veiksmus pakartojus visiems 4x4 subregionams, gaunamas 64D deskriptoriaus vektorius.



Paveikslėlis 3 SURF Deskriptoriaus sudarymas.

Siekiant aptikti objektą iš modelio (objektas) ir tikslo (scena su objektu, kurį reikia aptikti) paveikslėlių apskaičiuojami SURF požymiai bei gauti požymių deskriptoriai yra palyginami [4pav.]. Deskriptorių palyginimo etape greitam indeksavimui naudojamas Laplaso operatorius. Įprastai, dominantys taškai yra randami dėmių (angl. "blob") tipo struktūrose. Laplaso operatorius atskiria šviesias dėmes tamsiame fone nuo atvirkštinės situacijos (tamsių dėmių šviesiame fone). Tai nereikalauja papildomų skaičiavimų, nes ši informacija apskaičiuojama požymio aptikimo etape. Palyginimo etape lyginami tik tie požymiai, kurie turi tą patį kontrasto tipą. Jei kontrastas tarp dviejų požymių taškų skiriasi (tamsus šviesiame fone prieš šviesų tamsiame fone), tikrinamas taškas laikomas neatitinkančiu. Tai leidžia pagreitinti palyginimo etapą.

Nors SURF požymių algoritmas nepriklauso nuo mastelio ar pasukimo, šis metodas požymius apskaičiuoja tik nespalvotiems paveikslėliams. Taigi algoritmas neatskiria vidiniais požymiais panašių, tačiau skirtingų spalvų objektų ir negali jų atpažinti. L. Drejeris savo

darbe [9] pristato modifikuotą SURF algoritmą, kuris išsprendžia šią problemą, SURF deskriptorius skaičiuojant kiekvienam RGB kanalui atskirai. Visgi toks sprendimas arba reikalauja daugiau skaičiavimo laiko arba šiek tiek mažina teisingų atpažinimų procentą.

Paveikslėlis 4 SURF požymių palyginimas ir objekto aptikimas

Būtent SURF algoritmas dėl patikimumo ir deskriptorių apskaičiavimo greičio buvo pasirinktas objektų požymių suradimui, saugojimui bei objektų atpažinimui magistratūros studijų metu sukurtoje vaizdo atpažinimo sistemoje.

Kadangi į sistemą įvesti objektų vizualinė informacija jau saugoma kaip SURF deskriptoriai, taip pat dėl surastų požymių taškų unikalimo šis požymių algoritmas buvo pasirinktas ir objekto sekimui - objekto taškų, kuriuos reikia sekti, pradiniam nustatymui (objekto pozicijos kadre nustatymui) bei pakartotiniam tikslinimui.

2.4. Objektų sekimo algoritmų grupės

Kaip teigiama [1], objektų sekimo algoritmai yra skirstomi į tris grupes: požymiais pagrįsti, modeliu pagrįsti ir optinio srauto analize pagrįsti algoritmai. Nors algoritmai išskiriami į tris klases, tiems patiems taikymams gali būti tinkami skirtingoms klasėms priskirti metodai.

Toliau glaustai apžvelgiamos visos trys objektų sekimo algoritmų grupės.

2.4.1. Požymiais pagrįsti objektų sekimo algoritmai

Požymiais pagrįsti objektų sekimo algoritmai buvo sukurti nedidelio kiekio ryškiausių požymių sekimui. Šios grupės algoritmai iš kadro išskiria dominančius regionus (požymius) ir identifikuoja požymių atitikimus kiekvienam iš sekančių kadrų. Kadangi šio tipo algoritmai ieško specifinių atitikimų tarp kadrų, tai padeda sumažinti klaidingo sekimo tikimybę ir padidina sekimo tikslumą.

Pagrindiniai požymiais pagrįsti objektų sekimo algoritmai yra MeanShift sekimas, paslėpti Markovo modeliai (angl. *hidden Markov models*, HMM), dibtiniai neuroniniai tinklai (angl. *artificial neural network*, ANN), Kalman filtrai [22], dalelių filtrai [20], sudėtinis hipotetinis sekimas (angl. *Multiple Hypothesis Tracking*, MHT).

Paslėptas Markovo modelis objekto sekime paprastai naudojamas išgauti transformaciją tarp dviejų paveikslėlių ar judančių 3D struktūrų. Tačiau algoritmas nėra deterministinis, todėl gali būti daugiau nei viena transformacijos tikimybė bei dėl to neteisingas tam tikro sekamo objekto požymio pozicijos nustatymas. Taigi objekto sekime gali atsirasti klaidų. HMM pritaikius apmokymo algoritmus, šis metodas yra gana tinkamas objektų sekimui sudėtingose aplinkose. Tačiau būtent dėl „apmokymo“ poreikio šis metodas, taip pat kaip ir neuroniniai tinklai, nėra idealiai tinkantis universaliam panaudojimui.

Kitas algoritmas, kurį galima būtų priskirti požymiais paremtų sekimo algoritmų grupei, yra MHT algoritmas. Šis algoritmas ciklišk ai tikslina atitikimą tarp dviejų požymių grupių tol, kol pasiekiamas pageidaujamas tikslumas. Algoritmas tikrina kelis galimus sekamo objekto kandidatus einamame kadre ir siekia surasti geriausią atitikimą sekamo objekto požymiams. Šis metodas gerai tinka kelių objektų sekimui vienu metu. Tačiau šis algoritmas intensyviai naudoja atmintį ir yra imlus laikui, todėl ne itin tinka realaus laiko objekto sekimui realiose aplikacijose.

Mean Shift objektų sekimo grupės algoritmuose tikslo regionui aprašyti naudojama spalvų histograma. Panašumui tarp šablono (ar modelio) regiono ir esamo tikslo regiono nustatyti naudojami Kullback – Leibler nuokrypis, Bhattacharyya koeficientas ar kiti matai. Objekto sekimas atliekamas nuolat ieškant atstumo matavimo funcijos minimumo. Tai populiarus algoritmas - Mean Shift ir jo modifikacijos dažnai naudojamos įvairiems objektų sekimo taikymams, algoritmui kaip tikslinimas pritaikoma dalelių filtrai[19], RANSAC ir pan..

Kadangi MeanShift algoritmo modifikacija – CamShift [7][23] algoritmas yra gana patikimas, greitai veikiantis ir gerai tinkantis vieną dominuojančią spalvą turintiems objektams, jis pasirinktas kaip vienas iš algoritmų vaizdo atpažinimo sistemoje atpažinto objekto sekimui.

MeanShift algoritmas detaliai apžvelgiamas 2.5, o šio algoritmo modifikacija CamShift – 2.6 skyriuose.

2.4.2. Modeliu pagrįsti objektų sekimo algoritmai

Modeliu paremtas sekimas nuo požymiais paremto sekimo skiriasi tuo, jog modeliu paremtame sekime požymiai yra grupuojami, vertinami, analizuojami. Šios grupės algoritmams būtini išanksiniai duomenys apie sekamą objektą. Pavyzdžiui, sekant keletą objektų vienu metu, prieš pradedant sekimą turi būti žinomi dvejetainiai objektų požymiai. Tai galima pasiekti naudojant modelio atpažinimą. Modelio sudarymui [14] dažnai naudojama objekto briaunos, spalva, SIFT požymiai ir kita.

Šiame darbe pristatomus universalioje vaizdo atpažinimo sistemoje naudojamus algoritmus taip pat galime priskirti šiai grupei, nes objekto modelis (požymiai), yra žinomi objekto atpažinimo etape. Pradinė objekto pozicija nustatoma sulyginant duomenų bazėje saugomo atpažinto objekto požymius su požymiais, išgautais iš paveikslėlio atpažinimo metu.

2.4.3. Optinio srauto analize pagrįsti algoritmai

Optinis srautas - tai vektorių laukas, parodantis, kaip paveikslėlis keičiasi laike. Reikia įvertinti, jog trijų matmenų „greičių laukas“, nufilmuotas kameros, projektuojamas į dviejų matmenų erdvę – dėl to susidaro taip vadinamas „diafragmos efektas“.

Optinio srauto metodai paprastai naudojami srauto laukų generavimui kiekvienam pikseliui apskaičiuojant srauto vektorių. Dažnai tai atliekama ir su gretimais pikseliais.

Optinio srauto [18] pagalba sekami tam tikri objekto taškai. Kituose darbuose pristatomuose algoritmuose optinis srautas skaičuojamas SURF požymiams [2] [3], judančio objekto kampams, briaunoms ar sekamo objekto siluetui.

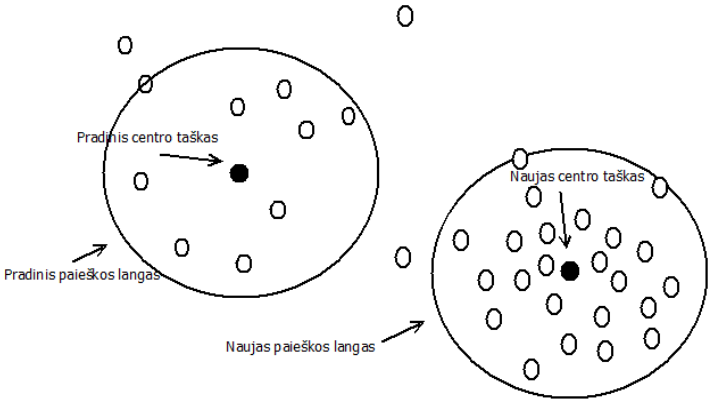
Šios grupės algoritmų viena iš problemų [4] – pradinis objekto aptikimas ir jo vietos kadre nustatymas, ypač tokiose aplinkose, kur yra intensyvus judėjimas dominančio objekto fone. Tačiau universalioje vaizdo atpažinimo sistemoje sekami atpažinti objektai, kurių pradinę poziciją kadre galima nustatyti sulyginant duomenų bazėje saugomus objekto požymius su požymiais, išgautais iš kadro atpažinimo metu.

Būtent dėl efektyvumo sekant iš anksto nustatytus taškus, vienas iš optinio srauto analize pagrįstų algoritmų grupės algoritmas – Lucas – Kanade metodas – ir buvo panaudotas vaizdo atpažinimo sistemoje.

Taigi, vaizdo atpažinimo sistemoje objektų sekimui pasirinkti du algoritmai: MeanShift algoritmo modifikacija CamShift ir optinio srauto Lucas – Kanade metodas. Tolimesniuose skyriuose detaliai apžvelgiami šie algoritmai, pristatomas jų panaudojamas realizuotoje sistemoje bei pateikiami gauti rezultatai.

2.5. Mean-Shift algoritmas

Mean Shift (arba Vidurkio Poslinkio) algoritmas yra neparametrinis metodas. Šis metodas gana tikslai suranda sekamą objektą nenaudojant nuodugnios, ilgai trunkančios paieškos. Mean Shift algoritmas pagrįstas iteraciniu procesu - pirmiausiai apskaičiuojama vidurkio poslinkio reikšmė esamoje taško pozicijoje, tada taškas perkeliamas į poziciją, atitinkančią apskaičiuotą reikšmę, tada vėl skaičiuojamas vidurkio poslinkis. Veiksmai kartojami, kol apskaičiuota reikšmė tenkina tam tikrus kriterijus (5 pav.).



Paveikslėlis 5 Vidurkio poslinkio paieška

Iteracinis metodas pradedamas nuo pradinio įvertinimo x. Tarkime, kernelio funkcija $K(x_i - x)$ yra duota. Ši funkcija apibrėžia artimų taškų svorius vidurkio pakartotiniam nustatymui. Paprastai esamo įverčio atstumui įvertinti naudojama Gauso kernelis $K(x_i - x) = \mathrm{e}^{\mathrm{c}\|\mathrm{x_i} - x\|^2}$. Svoriniu koeficientu įvertintas tankio vidurkis lange, apibrėžtame *K*, surandamas taikant 2-3 formulę

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)} \quad (2\text{-}3)$$

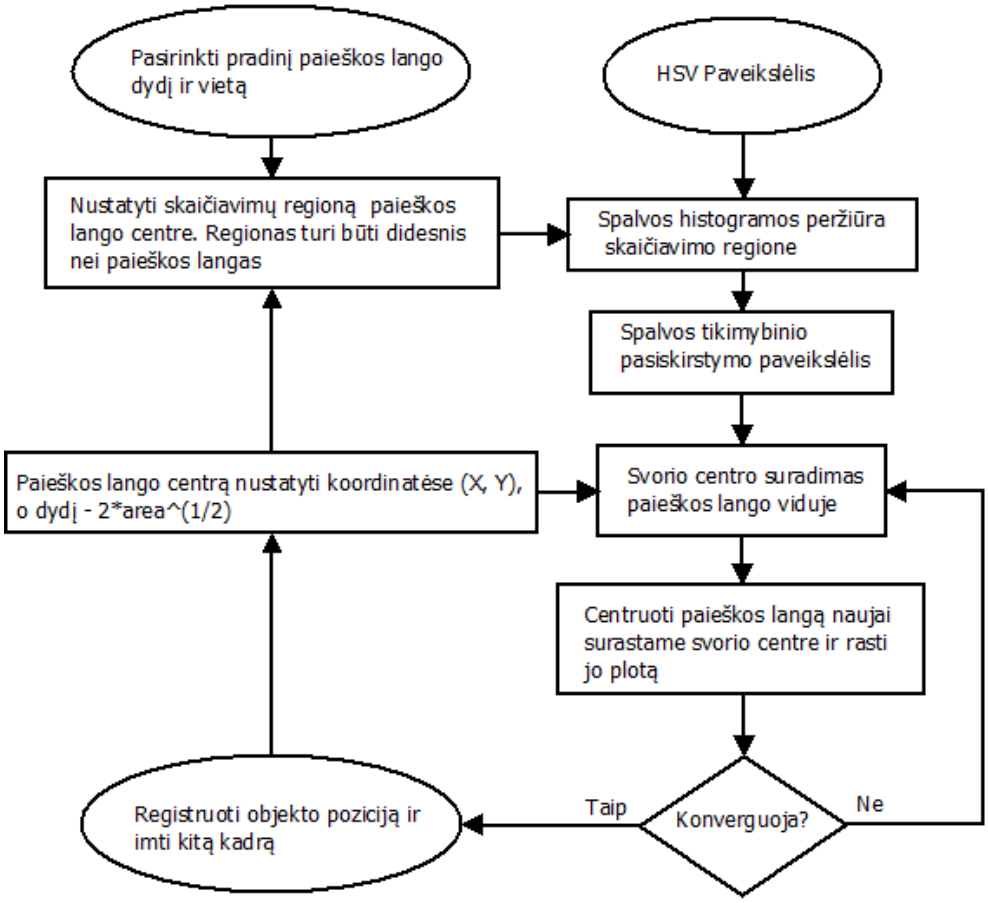
2-3 formulėje $N(x)$ yra x aplinka – taškų rinkinys, kuriems $K(x) \neq 0$. MeanShift algoritmas tada nustato $x \leftarrow m(x)$ ir kartoja įvertinimą iki *m(x)* konvergavimo.

2.6. CamShift algoritmas

CamShift (angl. *Continuously Adaptive Mean SHIFT*) [6][21] algoritmas - tai MeanShift algoritmo modifikacija. Kiekvienas tiesiai iš vaizdo sekos paimtas ir neapdorotas kadras konvertuojamas į spalvos tikimybinio pasiskirstymo paveikslėlį naudojant sekamos spalvos histogramos modelį (pvz. odos spalva, jei algoritmas naudojamas veido sekimui). Spalvų tikimybinio pasiskirtymo paveikslėliui pritaikius CamShift algoritmą, randamas

spalvoto objekto centras ir dydis. Sekamo objekto dydis ir vieta einamame kadre pažymima (priklausomai nuo sistemos, kurioje taikomas algoritmas, reikalavimų - registruojama, objektas paveikslėlyje apvedamas, ar pan.). Taip pat einamame kadre nustatyta objekto vieta ir dydis naudojama paieškos lango vietos ir išmatavimų sekančiame kadre nustatymui. Tada procesas kartojamas, taip nuolat kadre sekant judantį objektą.

Apibendrinta CamShift algoritmo, kuris yra Mean SHIFT algoritmo modifikacija, schema pateikta (6 pav.).



Paveikslėlis 6 CamShift algoritmas

Kaip jau minėta, CamShift algoritmas taikomas 2D spalvos tikimybinio pasiskirstymo paveikslėliui, gautam iš histogramos atgalinės projekcijos („back projection“).

Taigi, CamShift algoritmo žingsniai yra šie:

1. Visam paveikslėliui nustatyti tikimybės pasiskirstymo skaičiavimo regioną.
2. Pasirinkti pradinę 2D mean shift paieškos lango vietą.
3. Paskaičiuoti spalvos tikimybės pasiskirstymą 2D regione, kuris centruotas paieškos lange, ir kuris yra šiek tiek didesnis nei Mean SHIFT paieškos langas.

4. Naudojant Mean SHIFT algoritmą surasti paieškos lango centrą. Išsaugoti nulinį momentą (moment) (plotą ar dydį) ir centro koordinates.
5. Kitame kadre paieškos lango centrą nustatyti koordinatėse, rastose 4 žingsnyje ir nustatyti lango dydį pagal funkciją priklausančią nuo nulinio momento reikšmės. Kartoti trečią žingsnį.

2.7. Lucas – Kanade optinio srauto algoritmas

2.7.1. Optinis srautas

Optinis srautas - tai objektų, jų paviršių ar kraštinių matomo judėjimo šablonas, įtakotas sąlyginio judėjimo tarp stebėtojo (kameros ar akies) ir stebimo vaizdo. Šį objekto kraštinių ar paviršių judėjimą įvertina tokie optiniu srautu pagrįsti metodai kaip kad judesio nustatymas, objektų segmentavimas, erdvės netolygumo nustatymas ir kiti.

Kompiuterinėje regoje optinį srautą galima įvardyti kaip greičio lauką, susietą su iš kameros paimto paveikslėlio pokyčiais. Kaip jau minėta, pokyčiai atsiranda dėl sąlyginio judėjimo tarp kameros ir objekto arba dėl šviesos šaltinių, apšviečiančių stebimą vaizdą, judėjimo. Dauguma optinio srauto skaičiavimų yra pagrįsti ryškumo kitimu tarp dviejų kadrų.

Daugomoje optinio srauto skaičiavimo metodų daromos dvi prielaidos:

1. Paveikslėlio ryškumas $I(x, y, t)$ vienodai priklauso nuo koordinačių x ir y didesnėje paveikslėlio dalyje.
2. Kiekvieno judančio ar nejudančio objekto taško ryškumas bent jau trumpą laiką nekinta.

Atsižvelgiant į šias prielaidas, optinio srauto metodai skaičiuoja kiekvieno pikselio judėjimą tarp dviejų kadrų, iš vaizdo srauto (kameros ar įrašo) paimtų laiku t ir $t + \Delta t$. Kadangi šie metodai pagrįsti vaizdo signalo Teiloro eilučių aproksimacija, jie vadinami diferencialiniais. Kitaip sakant, šie metodai naudoja dalines išvestines, atsižvelgiant į laiko ir erdvės koordinates.

2D+t matmenų atveju, pikselis, kurio koordinatės (x, y, t) ir ryškumas $I(x, y, t)$ tarp dviejų kadrų pajudės Δx, Δy ir Δt. Atsižvelgiant į 2 prielaidą galima sudaryti sekančią paveikslėlio apribojimo lygtį (2-4 lygtis).

$$I(x, y, t) = I\left(x + \Delta x, y + \Delta y, t + \Delta t\right) \quad (2\text{-}4)$$

Atsižvelgiant į tai, kad judesys yra nedidelis, panaudojant Teiloro eilutę, galima sudaryti sekančią lygtį (2-5 lygtis):

$$I(x+\Delta x, y+\Delta y, t+\Delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \ldots \quad (2\text{-}5)$$

2-5 lygtyje „...“ yra aukštesnio laipsnio išraiškos. Iš šių lygčių seka, kad:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0 \quad (2\text{-}6)$$

Arba, padalinus iš Δt, gauname (2-7 lygtis).

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = 0 \quad (2\text{-}7)$$

Iš 2-7 lygties gauname:

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_Y + \frac{\partial I}{\partial t} = 0 \quad (2\text{-}8)$$

Kur V_x, V_y yra $I(x,y,t)$ optinio srauto (greičio) x ir y komponentai, o $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ ir $\frac{\partial I}{\partial t}$ paveikslėlio išvestinės koordinatėse (x,y,t). Išvestinės gali būti pakeistos I_x, I_y ir I_t. Taigi, galima užrašyti:

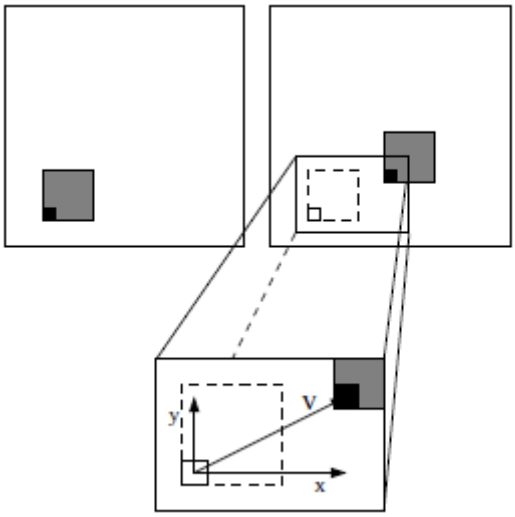
$$I_x V_x + I_y V_y = -I_t \quad (2\text{-}9)$$

Arba:

$$\nabla I \cdot \vec{V} + I_t = 0 \quad (2\text{-}10)$$

2-10 lygtyje $\nabla I = \left(I_x, I_y\right)$ yra erdvinis ryškumo gradientas, o V=(u,v) yra optinio srauto vektorius. Optinio srauto vektoriaus grafinis pavyzdys pateiktas (7 pav.).

Tačiau optinis srautas negali būti paskaičiuotas vien tik iš 2-9 ar 2-10 lygčių, nes šios lygtys turi du nežinomuosius ir todėl negali būti išspręstos. Ši situacija moksliniuose straipsniuose žinoma kaip „diafragmos“ (angl.: *aperture*) problema.



Paveikslėlis 7 Optinio srauto vektorius

Taigi, norint apskaičiuoti optinį srautą, reikia įsivesti keletą papildomų apribojimų ir sudaryti papildomas lygtis. Todėl visi optinio srauto metodai prideda papildomas sąlygas.

Optinio srauto skaičiavimui taikomi šie diferencialiniai metodai:

1. Horn - Shunck metodas;
2. Buxton – Buxton metodas – pagrįstas briaunų judesio kadrų sekoje modeliu;
3. Black – Jepson metodas – grubus optinis srautas naudojant koreliaciją;
4. Lucas – Kanade metodas.

Optinis srautas yra skaičiuojamas įvairiems sekamo objekto požymiams – kampams, briaunoms, SIFT požymiams. Taip pat literatūroje pateikiama pavyzdžių apie optinio srauto skaičiavimą įvertinant spalvų informaciją [10].

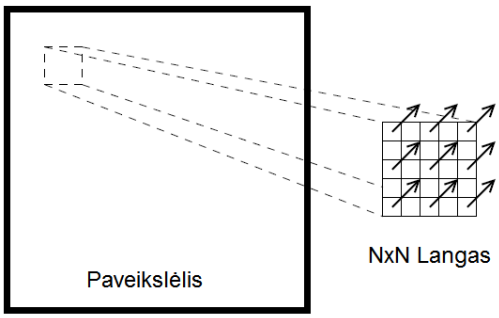
Vaizdo atpažinimo sistemoje kaip vienas iš algoritmų objektų sekimui buvo pasirinktas ir pritaikytas klasikinio Lucas – Kanade optinio srauto metodo piramidinė realizacija. Toliau detaliai pristatomas pats metodas, pateikiama paveikslėlio piramidės apibrėžimas bei šio metodo taikymas piramidei.

2.7.2. Lucas – Kanade optinio srauto algoritmas

Optinio srauto lygtį taikant grupei gretimų pikselių ir tariant, kad visi vienos grupės pikseliai turi tą patį greitį, optinio srauto skaičiavimo uždavinį galima supaprastinti iki tiesinės lygties sprendimo.

Taigi, Lucas - Kanade metodas optinio srauto lygties neapibrėžtumą išsprendžia apjungiant informaciją iš keleto gretimų pikselių. Dėl to šis metodas yra labiau atsparus paveikslėlio „triukšmui“. Tačiau Lucas – Kanade algoritmas negali įvertinti optinio srauto

vienodų paveikslėlio regionų viduje. Šis trūkumas algoritmo taikymui magistratūros metu sukurtoje vaizdo atpažinimo sistemoje nėra esminis, nes sukurtoje sistemoje šis metodas taikomas SURF požymių sekimui, o SURF požymiai nustatomi skirtinguose paveikslėlio regionuose.



Paveikslėlis 8 Optinio srauto vektorius atitinka visus NxN lango taškus

Tariant, kad dominančio paveikslėlio regiono pasikeitimas tarp dviejų artimų kadrų yra nedidelis, galima teigti, kad optinio srauto lygtis 2-9 tinka visai pikselių grupei („langui“), kurios centras p. Taigi lokalus paveikslėlio srauto vektorius (V_x, V_y) turi tenkinti šią lygtį:

$$\begin{aligned} &I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1) \\ &I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2) \\ &\vdots \\ &I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n) \end{aligned} \qquad (2\text{-}11)$$

2-11 lygtyje $q_1, q_2, \ldots q_n$ yra pikselių grupės (arba taip vadinamo „lango“) viduje esantys taškai, o $I_x(q_i), I_y(q_i), I_t(q_i)$ yra dalinės paveikslėlio I išvestinės esamu metu apskaičiuotos taške q_i atsižvelgiant į koordinates x, y ir laiką t. Šias lygtis galime užrašyti matricine forma $Av = \mathrm{b}$, kur

$$A = \begin{bmatrix} I_x(q_1)\, I_y(q_1) \\ I_x(q_2)\, I_x(q_2) \\ \vdots \quad \vdots \\ I_x(q_n)\, I_y(q_n) \end{bmatrix},\ v = \begin{bmatrix} V_x \\ \\ V_y \end{bmatrix} \text{ ir } b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix} \qquad (2\text{-}12)$$

Tačiau ši lygčių sistema turi daugiau lygčių nei nežinomųjų. Lucas-Kanade metodas šios sistemos dalinį sprendinį gauna naudojantis mažiausių kvadratų principu. Kitaip sakant, sprendžiama 2x2 sistema

$$A^T Av = A^T b \text{ arba } v = (A^T A)^{-1} A^T b \qquad (2\text{-}13)$$

kur A^T yra transponuota A matrica. Taigi, metodas skaičiuoja vektorių (V_x, V_y) pagal formulę

$$\begin{bmatrix} V_x \\ V_x \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_x(q_i) I_y(q_i) & \sum_i I_x(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix} \text{(2-14)}$$

Minėtas mažiausių kvadratų principas vienodai vertina visus viename lange esančius taškus. Tačiau praktikoje yra naudinga taškams, esantiems arčiau lango centro p, suteikti tam tikrus svorinius koeficientus. Tada 2-13 lygtis pakeičiama į 2-15

$$A^T W A v = A^T W b \text{ arba } v = (A^T W A)^{-1} A^T W b \quad \text{(2-15)}$$

Taip pat 2-14 lygtis keičiama į 2-15 lygtį

$$\begin{bmatrix} V_x \\ V_x \end{bmatrix} = \begin{bmatrix} \sum_i w_i I_x(q_i)^2 & \sum_i w_i I_x(q_i) I_y(q_i) \\ \sum_i w_i I_x(q_i) I_y(q_i) & \sum_i w_i I_x(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(q_i) I_t(q_i) \\ -\sum_i w_i I_y(q_i) I_t(q_i) \end{bmatrix} \text{(2-15)}$$

2-15 lygtyje $W_{ii} = w_i$ yra svorinių koeficientų pikselio q_i lygčiai diagonalinė matrica, o w_i - atstumo tarp q_i ir p Gauso funkcija.

Taikant mažiausių kvadratų metodą daroma prielaida, kad paveikslėlyje esančių klaidų Gauso pasiskirstymo vidurkis yra 0. Tačiau jeigu tikimasi, jog paveikslėlyje yra tam tikras procentas klaidų "nukrypimų" (pernelyg didelės įprasto Gauso klaidų pasiskirstymo neatitinkančios reikšmės), jų aptikimui taikoma statistinė analizė ir atitinkamai sumažinami svoriniai koeficientai, kas lemia mažesnę įtaką galutiniam apskaičiuotam srauto vektoriui V_x, V_y.

Aprašytas klasikinis Lucas - Kanade metodas gali būti naudojamas tik tada, kai paveikslėlio optinio srauto vektorius V_x, V_y tarp dviejų kadrų yra pakankamai mažas - dažniausiai ne didesnis už kelis pikselius. Kai optinio srauto vektorius viršija šią ribą, šis metodas taikomas pradinių, ne itin tikslių vektorių patikslinimui. Tada pradiniai vektoriai apskaičiuojami ekstrapoliuojant ankstesniuose kadruose apskaičiuotus vektorius arba Lucas - Kanade algoritmą ciklišk ai taikant sumažintiems paveikslėliams.

Būtent didelių judesių paveikslėlyje įvertinimui Jean-Yves Bouguet [12] pristatė Lucas – Kanade algoritmo realizaciją pritaikytą paveikslėlio piramidėms.

Siūlomoje Lucas - Kanade metodo modifikacijoje [12] pirmiausiai sudaroma $n_x x n_y$ dydžio paveikslėlio I piramidė. I^0 = I - tai „nulinio“ lygio (aukščiausios raiškos) paveikslėlis (paimtas tiesiai iš kameros ar kito vaizdo srauto). Šiame, pradiniame piramidės lygyje paveikslėlio plotis ir aukštis yra $n_x^0 = n_x$ ir $n_y^0 = n_y$. Tada rekursyviai sudaroma paveikslėlio piramidė: I^1 apskaičiuojamas iš I^0, I^2 apskaičiuojama iš I^1 ir taip toliau. Tegul L = 1,2... yra

vienas piramidės lygis, o I^{L-1} - paveikslėlis lygyje L-1. Atitinkamai n_x^{L-1} ir n_Y^{L-1} yra paveikslėlio I^{L-1} plotis ir aukštis. Tada paveikslėlis I^{L-1} apibrėžiamas 2-16 formule

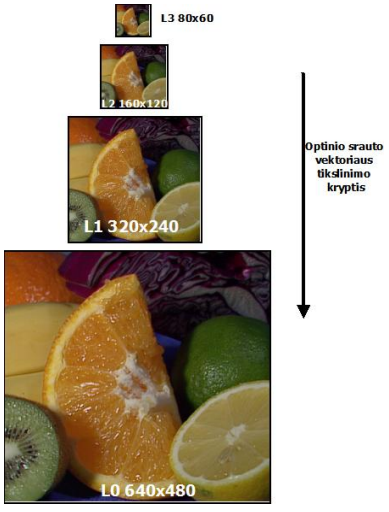
$$I^L(x,y) = \frac{1}{4} I^{L-1}(2x,2y) +$$
$$\frac{1}{8}(I^{L-1}(2x-1,2y) + I^{L-1}(2x+1,2y) + I^{L-1}(2x,2y-1) I^{L-1}(2x,2y+1)) + \qquad (2\text{-}16)$$
$$\frac{1}{16}(I^{L-1}(2x-1,2y-1) + I^{L-1}(2x+1,2y+1) + I^{L-1}(2x-1,2y+1) I^{L-1}(2x+1,2y+1))$$

Be to, paveikslėlio I^L plotis n_x^L ir aukštis n_Y^L yra didžiausi sveikieji skaičiai, tenkinantys šias dvi sąlygas

$$n_x^L \le \frac{n_x^{L-1}+1}{2}; n_y^L \le \frac{n_y^{L-1}+1}{2} \qquad (2\text{-}17)$$

Pagal šias lygtis rekursyviai sudaroma paveikslėlio I piramidė. $\{I^L\}_{L=0,\ldots,L_m}$. L_m - tai piramidės aukštis. Praktikoje L_m reikšmės dažniausiai būna 2, 3, 4, ar 5. Tačiau tai priklauso nuo paveikslėlio raiškos ir didžiausio tikėtino optinio srauto vektoriaus, kurį norima įvertinti. Kuo didesnė paveikslėlio raiška, tuo daugiau piramidės lygių reikia. Taip pat pat piramidės aukštis turi būti pasirenkamas įvertinant maksimalų tikėtiną optinį srautą, nes aukštesnio piramidės lygio paveikslėliai leidžia įvertinti didesnį judesį kadre.

Pavyzdžiui, 640x480 raiškos paveikslėlio piramidėje (9 paveikslėlis) I^1, I^2, I^3, I^4 yra atitinkamai 320x240, 160x120, 80x60 ir 40x30 raiškos. Dar labiau mažinti paveikslėlį dažniausiai nėra prasmės.



Paveikslėlis 9 Paveikslėlio piramidė

Taigi, optinio srauto sekimo uždavinys yra paveikslėlyje I esančiam taškui u surasti jį atitinkačią vietą v = u + d paveikslėlyje J, kur d - taško optinio srauto vektorius. Sudarius paveikslėlių I ir J piramides, optinio srauto sekimo algoritmas yra sekantis:

Pirmiausiai, optinis srautas apskaičiuojamas aukščiausiame piramidės lygyje L_m. Tada, apskaičiuoti rezultatai perduodami žemesniam lygiui L_{m-1} kaip pradinis L_{m-1} lygio paveikslėlio srauto vektoriaus spėjimas. Turint šį pradinį spėjimą, L_{m-1} lygyje apskaičiuojamas patikslintas optinis srautas bei rezultatai perduodami į L_{m-2} lygį. Veiksmai ciklišk ai kartojami, kol pasiekiamas nulinis piramidės lygis (originalus paveikslėlis) ir gaunamas optinio srauto vektorius d. Optinis srautas kiekviename iš lygių skaičiuojamas taikant Lucas – Kanade metodą. Sekamo taško koordinatės paveikslėlyje J randamos prie ankstesnių koordinačių pridėjus optinio srauto vektorių d.

2.8. CamShift ir Optinio srauto Lucas – Kanade metodo palyginimas

Vidurkio poslinkio (Mean Shift) CamShift paremta objekto modeliu, sudarytu pagal spalvų histogramą. Algoritmas tiksliai seka vienspalvius objektus, įvertina objekto mastelį (toliau / arčiau kameros), atsparus daliniams ar visiškiems sekamo objekto uždengimams - tarp kameros ir sekamo objekto laikinai atsiradus kliūčiai, algoritmas sėkmingai toliau seka objektą, kai tik kliūtis jo nedengia. Tačiau, kadangi CamShift sekimo algoritmas pagrįstas spalvų histograma, šis metodo tikslumas mažėja, sekamame objekte didėjant skirtingų spalvų skaičiui. Algoritmas nėra tinkamas spalvotų objektų sekimui.

Optinio srauto Lucas – Kanade metodo piramidėms algoritmas seka tam tikrus, iš anksto nustatytus taškus. Iš daugiaspalvių objektų (turinčių tam tikrą tekstūrą) iškyrus SURF požymių taškus ir nustačius jų koordinates, šis algoritmas; priešingai nei CamShift, tinka daugiaspalvių objektų sekimui. Tačiau pakankamo kiekio požymių iš vienspalvių objektų SURF algoritmas išskirti negali, nes SURF algoritmas dirba su nespalvotais paveikslėliais, o vienspalvį paveikslėlį pavertus į nespalvotą, pilkumo atspalvis būtų vienodas. Tad taikant šį metodą vienspalviams objektams, sekimas būtų netikslus, arba iš viso nebūtų nustatyti sektini taškai.

Optinio srauto Lucas – Kanade metodas nėra atsparus daliniam ar visiškam objekto uždengimui [25] – objektą uždengus, prarandama dalis taškų. Šios problemos sprendimui reikalingi papildomi veiksmai – [25] straipsnyje tam taikomi neuroniniai tinklai ir prisitaikantys filtrai.

1 lentelėje pateikiamas glaustas šių algoritmų palyginimas.

Lentelė 1 Lucas-Kanade ir CamShift algoritmų palyginimas

	Optinio srauto Lucas-Kanade metodas	**CamShift**
Objekto modelis	Tam tikri objekto taškai (požymiai)	Spalvų histograma
Vienspalvių objektų sekimas	Netinkamas	Tinkamas
Daugiaspalvių objektų sekimas	Tinkamas	Netinkamas
Sekimas po dalinio objekto uždengimo	Prarandama dalis sekamų taškų	Sėkmingai seka po dalinio objekto uždengimo
Sekimas po visiško objekto uždengimo	Prarandami sekami taškai	Sėkmingai seka po visiško objekto uždengimo

2.9. Algoritmų analizės išvados

- Kadangi SURF algoritmas sukurtoje vaizdo atpažinimo sistemoje jau naudojamas objektų atpažinimui, šis algoritmas objektų sekimo realizacijoje pasirinktas atpažinto objekto pradinės pozicijos nustatymui, taškų, kuriuos reikia sekti, išgavimui ir tikslinimui tam tikrus taškus pametus. SURF algoritmas tinka šioms užduotimis, nes deskriptorių apskaičiavimas ir palyginimas trunka santykinai neilgai (iki 0,5 sek.); taip pat jį taikant apskaičiuoti deskriptoriai yra unikalūs.
- Vaizdo atpažinimo sistemoje objektų klases apribojus iki kelių skirtingų ir lengvai apibrėžiamų grupių, išankstinio apmokymo algoritmus būtų galima panaudoti atpažintų objektų sekimui. Tačiau išankstinio apmokymo reikalaujantys objektų sekimo algoritmai sukurtai vaizdo atpažinimo sistemai yra netinkami, nes sistema atpažįsta įvairių grupių objektus, tad apmokymo procesas būtų sudėtingas arba iš viso neįmanomas.
- Vieni iš populiariausių algoritmų, kuriems nereikia sudėtingo išankstinio apmokymo – tai CamShift ir optinio srauto Lucas – Kanade algoritmai
- Lucas-Kanade algoritmas, įvertinantis optinį srautą kadre, tinka tam tikrų objekto taškų, nustatytų naudojant SURF algoritmą, sekimui. Kadangi SURF algoritmas nėra pritaikytas vienspalviams objektams, Lucas-Kanade metodas ir SURF nustatyti sektini taškai vienos dominuojančios spalvos objektams nėra tinkamas.

- Lucas-Kanade algoritmas nėra atsparus objekto taškų uždengimui – sekamą daiktą uždengus, taškai prarandami. Sukurtoje vaizdo atpažinimo sistemoje ši problema sprendžiama taškų tikslinimu naudojant objekto modelį.
- CamShift algoritmas, kaip modelį naudojantis spalvos histogramą, tinka vienspalvių objektų sekimui.
- Vaizdo atpažinimo sistemoje, siekiant sekti tiek vienspalvius, tiek spalvotus objektus, bus panaudojami ir Lucas – Kanade, ir CamShift algoritmai, dinamiškai parenkant tinkamesnį.

3. UNIVERSALI VAIZDO ATPAŽINIMO SISTEMA

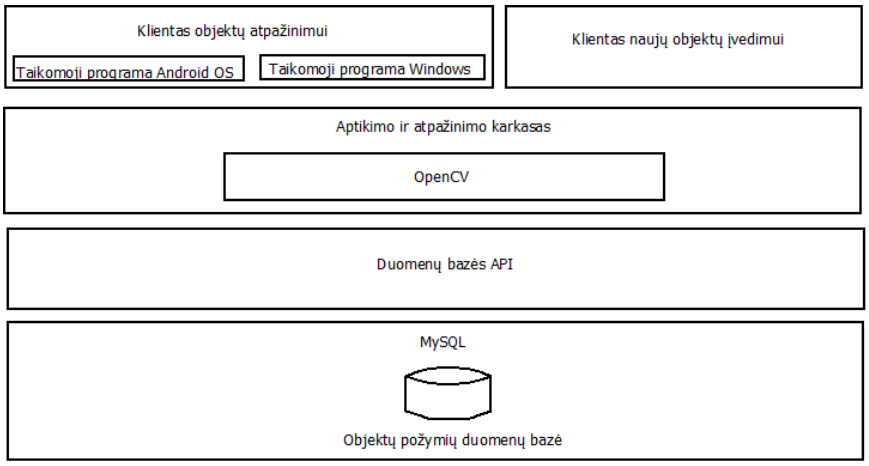
Magistratūros studijų metu sukurta vaizdų atpažinimo sistema skirta kliento telefonu ar kompiuterio kamera nufotografuotų objektų atpažinimui ir informacijos apie juos gavimui. Sistema sukurta naudojantis atvirojo kodo OpenCV [16] vaizdo apdorojimo biblioteką, parašytą C/C++ kalba.

Sistema gali būti diegiama asmeniniuose kompiuteriuose su Windows arba Linux operacine sistema arba mobiliuosiuose įrenginiuose su Android operacine sistema. Vaizdo įvedimui į sistemą naudojama internetinės arba mobiliųjų įrenginių kameros.

Sistema sukurta naudojant kliento - serverio architektūrą. Ją sudarančios posistemės pateiktos 10 paveikslėlyje Sistemos veikimo principas [15]:

- Klientas nufotografuoja norimą atpažinti ar išsaugoti objektą naudodamas vaizdo kamerą.
- Objektas yra apdorojamas naudojant SURF algoritmą ir pritaikius šį algoritmą gauti deskriptoriai, kartu su papildoma informacija, yra serializuojami panaudojant JSON formatą.
- JSON formato žinutė (žinutės pavyzdžiai pateikti 3 priede) yra pasiunčiama į Ubuntu Linux 11.10 operacinėje sistemoje veikiantį serverį.
- Šiame serveryje objekto SURF deskriptoriai lyginami su saugomais duomenų bazėje, ir randamas didžiausias atitikimas tam tikru tikslumu (atpažinimo atvejis) arba MySQL duomenų bazėje objekto informacija ir deskriptoriai yra išsaugojami („apmokymo“, įvedimo atvejis).
- Operacijos rezultatas yra grąžinamas klientui.
- Atpažintas objektas yra sekamas.

Sistema apmokoma panašiu būdu – klientas, nufotografavęs objektą, kartu įveda ir informaciją apie jį – (aprašymą, pavadinimą ir pan.); o JSON formatu perduotus duomenis serveris išsaugo duomenų bazėje.



Paveikslėlis 10 Bendradarbiaujančios posistemės

3.1. Universalios Vaizdo Atpažinimo Sistemos reikalavimai ir funkcijos

Esminiai reikalavimai sukurtai vaizdo atpažinimo sistemai [15] yra šie:

- Turi veikti naudodama kliento - serverio architektūrą.
- Turi turėti klientus Windows, Linux, Android OS operacinėms sistemoms.
- Nuotraukų apdorojimui (požymių išskyrimui) turi naudoti SURF algoritmą.
- Objektų informaciją sistema turi išsaugoti duomenų bazėje.
- Sistema duomenų apsikeitimui tarp kliento ir serverio turi naudoti JSON formatu užkoduotus duomenis.
- Sistema turi taupiai naudoti mobilaus telefono baterijos energiją.
- Sistema turi galėti klasifikuoti objektus į klases, pavyzdžiui, „knygos“, „obuoliai“, „riešutai“ ir pan.

Serverio pagrindinės funkcijos yra šios:

- Aptarnauti prisijungiančius klientus.
- Sulyginti iš kliento gautus SURF deskriptorius su saugomais duomenų bazėje, ir taip rasti atitikimus.
- Ieškoti, šalinti, atnaujinti, pridėti objektus į duomenų bazę.
- Pateikti rezultatus ir dumenų bazėje saugomo atpažinto objekto deskriptorius klientui.

Kliento pagrindinės funkcijos yra šios:

- Nufotografuoti objektus.
- Nuotraukų pradiniam apdorojimui (požymių išgavimui) naudoti SURF algoritmą.

- Komunikuoti su serveriu.
- Sekti atpažintą objektą

3.1.1. Reikalavimai atpažintų objektų sekimui

Objektų sekimo moduliui keliami šie reikalavimai:

- Objektų sekimas turi veikti realiu laiku.
- Objektų sekimas turi veikti objektų atpažinimo kliente
- Objektų sekimo modulis turi galėti sekti tiek vienspalvius, tiek spalvotus objektus.
- Sekamam objektui laikinai išėjus iš kameros stebimo lauko arba laikinai objektą uždengus, objektų sekimo modulis turi aptikti vėl į kadrą sugrąžintą objektą ir jį sekti toliau.

3.2. Universalios Vaizdo Atpažinimo Sistemos architektūra

Kaip jau minėta, sukurta sistema pagrįsta kliento - serverio architektūra. Serveris su klientais bendrauja naudojant JSON žinutes. Toliau detaliau apžvelgiama serverio ir klientų architektūra.

3.2.1. Universalios Vaizdo Atpažinimo Sistemos serveris

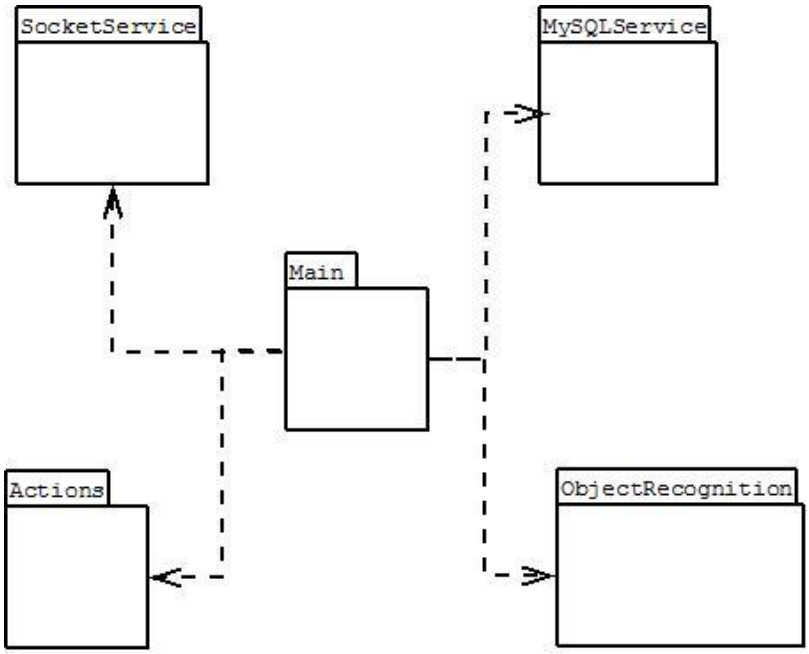
Vaizdo atpažinimo serveryje [15] naudojamos šios technologijos:

- MySQL duomenų bazių serveris.
- Ubuntu Linux 11.10 64-ių bitų operacinė sistema.
- OpenCV 2.1 biblioteka.
- MySQL/C++ Connector biblioteka.
- Boost biblioteka

Serveris sudarytas iš kelių pagrindinių komponentų (11 pav.):

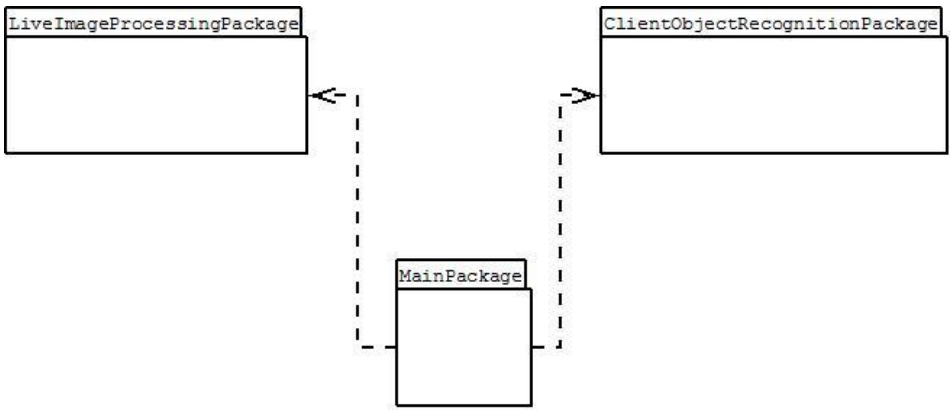
- **SocketService.** Sistemos komponentas, atsakingas už darbą su tinklu. Naudojantis šiuo komponentu serveris komunikuoja su klientų per TCP/IP protokolą ir 10 000-ąją jungtį (*port*).
- **MySQLService.** Sistemos komponentas, atsakingas už darbą su duomenų baze. Naudojantis šiuo komponentu serveris atlieka įterpimo (*create*), skaitymo (*read*), atnaujinimo (*update*), pašalinimo (*delete*) operacijas (*CRUD operations*). Šis komponentas naudoja *MySQL/C++ Connector* biblioteką.

- **Actions.** Sistemos komponentas, atsakingas už kliento pasirinktų veiksmų apdorojimą. Informacija apie kliento pasirinktą veiksmą (pridėti, pašalinti, atpažinti, atnaujinti) yra perduodama šiam komponentui, o jis atlieka norimą veiksmą ir klientui praneša veiksmo atlikimo rezultatą - ar veiksmą pavyko atlikti, ar ne.
- **ObjectRecognition.** Sistemos komponentas, atsakingas už objektų atpažinimą. Jame yra atliekamos paveikslio atpažinimo ir palyginimo operacijos.
- **Main.** Pagrindinis sistemos komponentas. Jo paskirtis - susieti visus kitus komponentus.



Paveikslėlis 11 Bendra sistemos serverio paketų diagrama

3.2.2. Universalios Vaizdo Atpažinimo Sistemos klientai



Paveikslėlis 12 Bendra sistemos kliento paketų diagrama

Vaizdo atpažinimo sistemos klientui naudojamos šios technologijos:

- OpenCV 2.1 (Windows/Linux klientui) ir OpenCV 2.3 (Android klientui) bibliotekos

- QT karkasas (Windows / Linux klientui)
- Android karkasas (Android klientui)

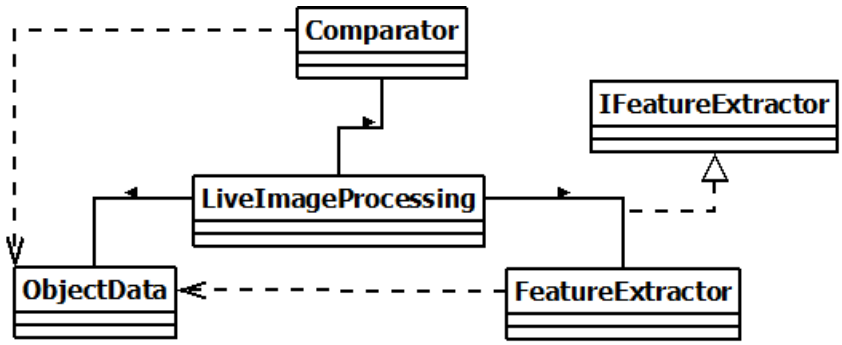
Sistemos klientas sudarytas iš trijų pagrindinių komponentų (12 paveikslėlis):

- **ClientObjectRecognitionPackage**. Šis komponentas skirtas iš kameros paimto kadro apdorojimui, nuotraukoje esančių objekto požymių išskyrimui bei kreipimuisi į serverį.
- **MainPackage**. Sistemos komponentas, atsakingas už kliento sistemos paleidimą ir kliento sistemos veikimą.
- **LiveImageProcessingPackage**. Sistemos komponentas, atsakingas už iš kameros paimto vaizdo atvaizdavimą ekrane ir atpažinto objekto sekimą.

3.3. Universalios Vaizdo Atpažinimo Sistemos objektų sekimo modulis

Magistratūros studijų metu sukurtoje vaizdo atpažinimo sistemoje realizuotas objektų sekimo modulis, atsakingas už atpažintų judančių objektų sekimą – tai yra objektų apibrėžimą kadre, šiems judant. Sekimo modulis veikia objektų atpažinimo dalyje (žiūrėti bendradarbiaujančias posistemes, 10 paveikslėlis)

Modulis sudarytas iš kelių klasių (13 paveikslėlis). Pagrindinėje, LiveImageProcessing klasėje realizuota tinkamiausio atpažintam objektui sekimo algoritmo parinkimas bei CamShift ir modifikuotas Lucas-Kanade optinio srauto algoritmas atpažinto objekto taškų sekimui. Comparator klasė skirta objekto modelio (SURF deskriptorių, gautų iš serverio atpažinus objektą) ir iš kameros gauto kadro SURF požymių palyginimui bei atpažinto objekto taškų, kuriuos reikia sekti, pradinių koordinačių nustatymui – kitaip sakant, pradiniam objekto aptikimui bei sekant praradus tam tikrus taškus – jų patikslinimui. FeatureExtractor klasė naudojama iš kameros gauto kadro SURF požymių gavimui. ObjectData klasėje saugoma informacija apie objektą (dominantys taškai ir jų deskriptoriai, objekto pavadinimas ir aprašymas).



Paveikslėlis 13 Objekto sekimo klasių diagrama

Atpažintų objektų sekimas bus vykdomas sekančiu principu. Serveriui atpažinus nufotografuotą objektą, klientui kartu su atpažinto objekto informacija grąžinama ir duomenų bazėje saugomi atpažinto objekto SURF deskriptoriai. Gauti deskriptoriai naudojami atpažinto objekto aptikimui pirmame kadre. Tada nustatoma, kuris sekimo algoritmas (CamShift ar modifikuotas OpticalFlow) yra labiau tinkamas ir tas metodas naudojamas objekto sekimui. Praradus tam tikrą kiekį sekamų objekto taškų, iš serverio gauti požymiai naudojami sekamo objekto taškų patikslinimui.

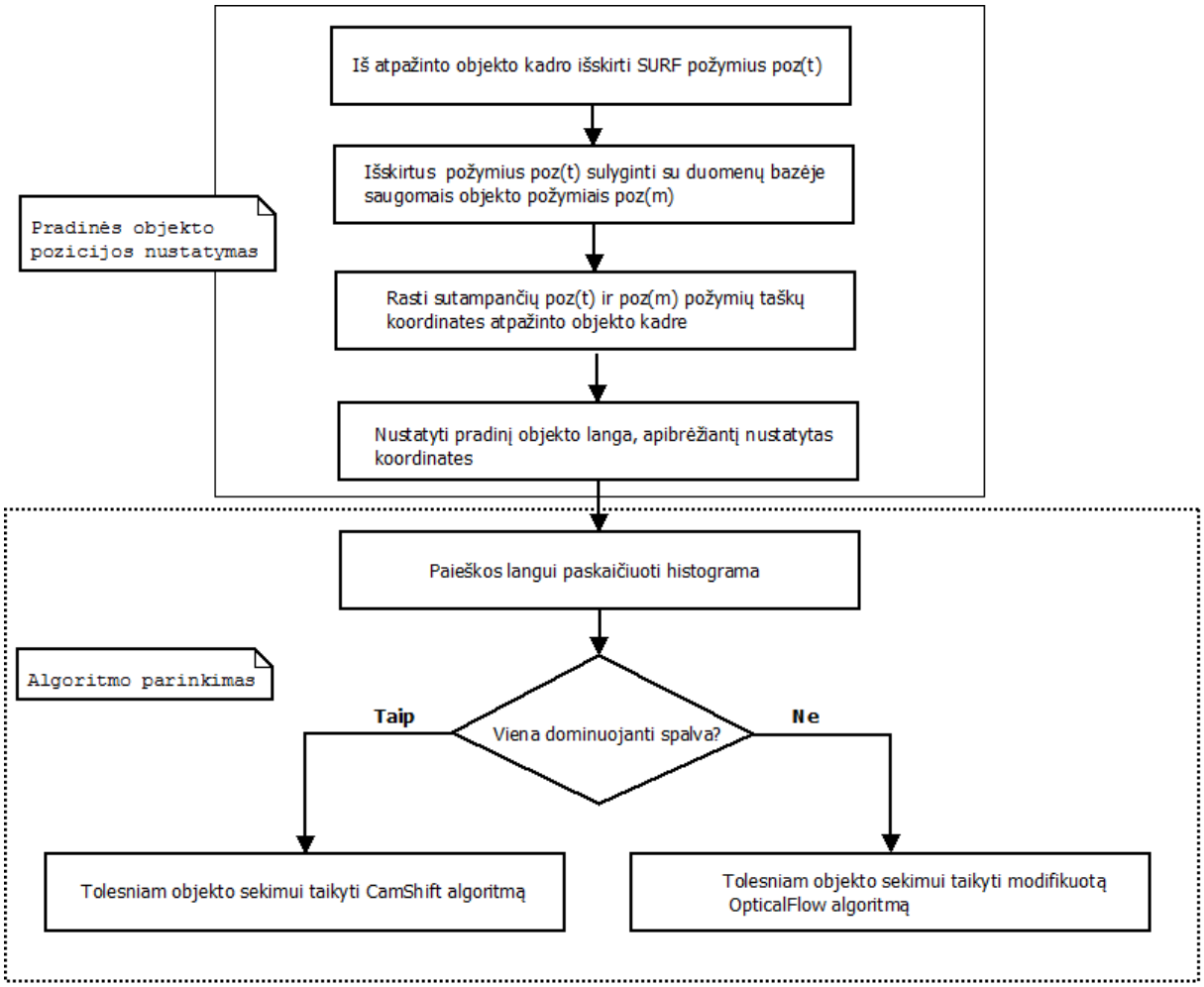
4. OBJEKTO SEKIMO ALGORITMŲ PANAUDOJIMAS VAIZDO ATPAŽINIMO SISTEMOJE

Siekiant galimybės sekti įvairių klasių objektus (tiek spalvotus, tiek vienspalvius), vaizdo atpažinimo sistemoje pritaikytas kombinuotas objektų sekimo algoritmas, panaudojant tiek CamShift, tiek Lucas – Kanade optinio srauto algoritmus. Sistemoje realizuotas sekimo modulis kiekvienam atpažintam objektui nustato geriausiai tinkantį sekimo algoritmą, ir jį taiko kituose kadruose. Jei parenkamas Lucas – Kanade algoritmas, sekimui naudojami SURF požymiai. Sekimui parinkus Lucas – Kanade optinio srauto metodą, nuolat tikslinami prarasti („pamesti“) taškai, taip išsprendžiant dalinio sekamo objekto uždengimo problemą. Toliau pristatomas realizuotas kombinuotas objektų sekimo metodas.

Pagrindiniai algoritmo žingsniai yra šie:

1. Nustatoma pradinė atpažinto objekto pozicija kadre.
2. Pagal objekto vizualias savybes (spalvą) parenkamas labiausiai tinkantis sekimo algoritmas.
3. Tolesniam sekimui taikomas parinktas algoritmas. Jei sekamo objekto taškai pametami, objektas pakartotinai aptinkamas sulyginant modelio ir kadro požymių taškų deskriptorius.

14 paveikslėlyje pateikta supaprastinta algoritmo diagrama. Ištisine linija apibraukta pradinės objekto pozicijos nustatymo dalis, o taškine – tinkamiausio algoritmo parinkimo dalis.



Paveikslėlis 14 Kombinuoto algoritmo diagrama

4.1. Pradinės objekto pozicijos nustatymas

Atpažinus kliente nufotografuotą objektą, iš serverio JSON formatu gaunama ne tik vartotojui pateikiama objekto informacija (pavadinimas ir aprašymas), bet ir duomenų bazėje saugomi objekto SURF vietinių požymių taškai su deskriptoriais. Ši informacija laikoma objekto modeliu. Jei p yra objekto modelio m SURF požymio taškas $p_m=(x_m, y_m)$, tai šio taško deskriptorius yra $poz_m(p)$; visa požymių taškų aibė yra P_m.

Turint šiuos duomenis, pradinė objekto pozicija nustatoma taip (14 paveikslėlis, ištisine linija apibraukta dalis):

1. Iš kameros gautame kadre naudojant SURF algoritmą išskiriami vietinių požymių taškai P_t ir jų deskriptoriai poz_t.
2. Kiekvienas P_t taškų deskriptorius $poz_t(p_t)$ sulyginamas su visais modelio deskriptoriais poz_m
3. Nustatomos tų P_t taškų, kurių deskriptoriai atitinka vieną iš modelio deskriptorių $poz_t(p_t)=poz_m(x)$, koordinatės.

4. Nustatomas pradinis objekto langas – apibrėžiami visi 3 žingsnyje rasti taškai.

Pradinės atpažinto objekto vietos nustatymas gali būti aprašytas šiuo pseudo kodu:

```
k=paimtiKadraIsKameros();
pozm[]=gautiModelioDeskriptorius();
pozt[]=iskirtiSURFDeskriptorius(k);
obj_taskai[];
for each(pozt[] as pozt){
   for each(pozm[] as pozm){
          if(atitinka(pozt, pozm)){
                obj_taskai[]=gautiKoordinates(pozt);
          }
   }
}
sek_objPav=gautiPaveikslelioDali(obj_taskai);
apibreztiTaskus(obj_taskai);
```

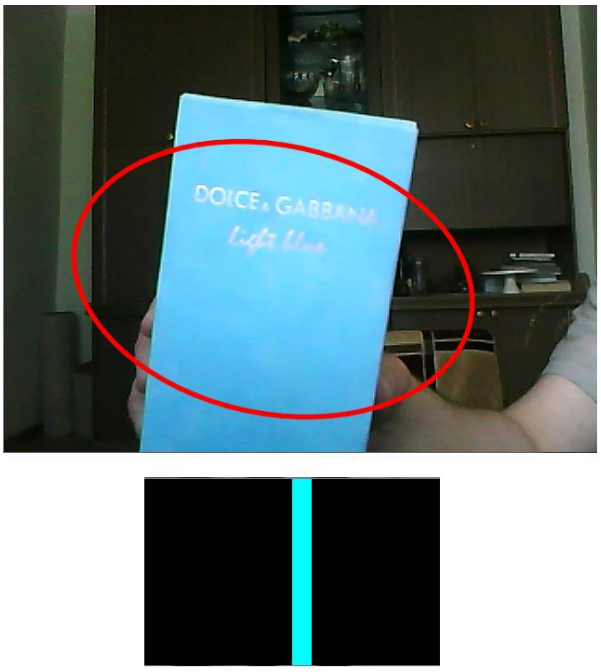
Aptikus norimą sekti objektą, nustatoma būtent šiam objektui geriau tinkantis sekimo algoritmas.

4.2. Algoritmo parinkimas

Geriausio sekimo algoritmo atpažintam ir kadre aptiktam objektui parinkimo žingsniai yra šie (14 paveikslėlyje taškinė linija):

1. Paveikslėlio dalis, apribota požymių taškus apibrėžiančio stačiakampio, verčiama į HSV formatą.
2. Šiai daliai paskaičiuojama spalvos (angl. *hue*) histograma.
3. Jei spalvos histogramoje aptinkama viena ženkliai dominuojanti spalva (pavyzdys 16 paveikslėlyje), tolimesniam objekto sekimui taikoma CamShift algoritmas.
4. Jei spalvos histogramoje dominuojanti spalva neaptinkama (pavyzdys 15 paveikslėlyje), objekto sekimui toliau taikoma Lukas – Kanade optinio srauto metodo piramidinė modifikacija, sekant objekto aptikimo žingsnyje surastus taškus.

Paveikslėlis 15 Optimalaus algoritmo parinkimas. Nėra dominuojančios spalvos. Tinkamesnis Lucas-Kanade optinio srauto algoritmas



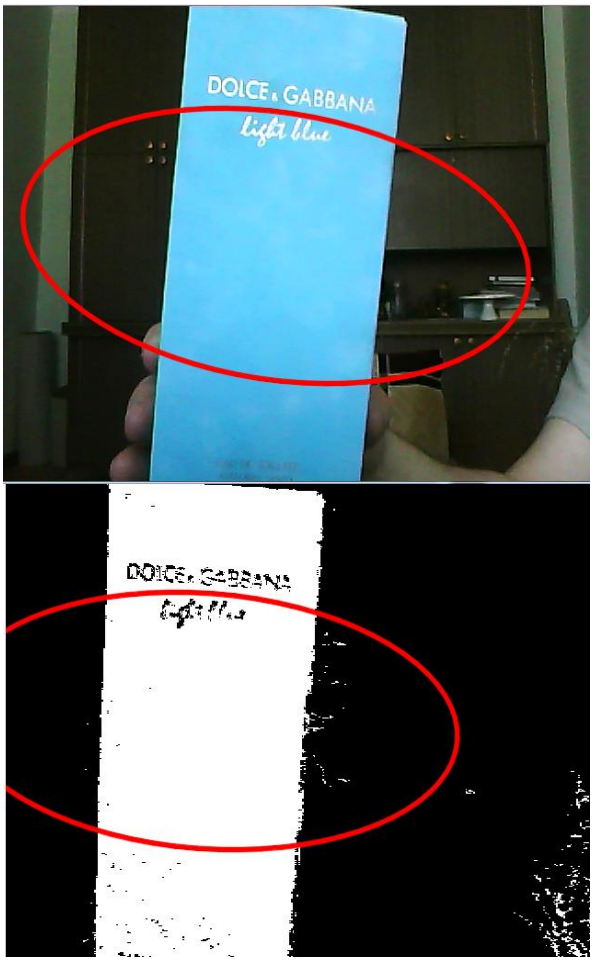
Paveikslėlis 16 Optimalaus algoritmo parinkimas. Rasta dominuojanti spalva. Tinkamesnis CamShift algoritmas

Toliau objektas sekamas naudojant atpažintam objektui geriausiai tinkantį algorimtą. CamShift algoritmo pritaikymas pristatomas 4.3. skyriuje, o Lucas – Kanade optinio srauto algoritmo pritaikymas – 4.4. skyriuje.

4.3. CamShift algoritmo pritaikymas atpažinto objekto sekimui

Atpažintą objektą sekant CamShift metodu, kiekvienam iš kameros paimtam kadrui taikomi šie žingsniai:

1. Pradedant sekimą, apskaičiuojama kadro k_1 spalvos histograma.
2. Kadro k_x histogramai paskaičiuojama atgalinė projekcija (17 pav.) (angl. *Back projection*). Atgalinės projekcijos paveikslėlyje balti taškai reiškia tikimybę, jog atitinkami originalaus paveikslėlio taškai priklauso objektui, kurio histograma naudojama.
3. k_x ir k_{x-1} atgalinėms projekcijoms taikant CamShift algoritmą (kaip paieškos langą naudojant objekto aptikimo etape gautus duomenis) randamos naujos objekto centro koordinatės kadre k_x.
4. Tolimesniems kadrams kartojami 2-3 žingsniai. 3 žingsnį vykdant k_x kadrui, pradinis paieškos langas nustatomas pagal k_{x-1} kadre gautą informaciją.

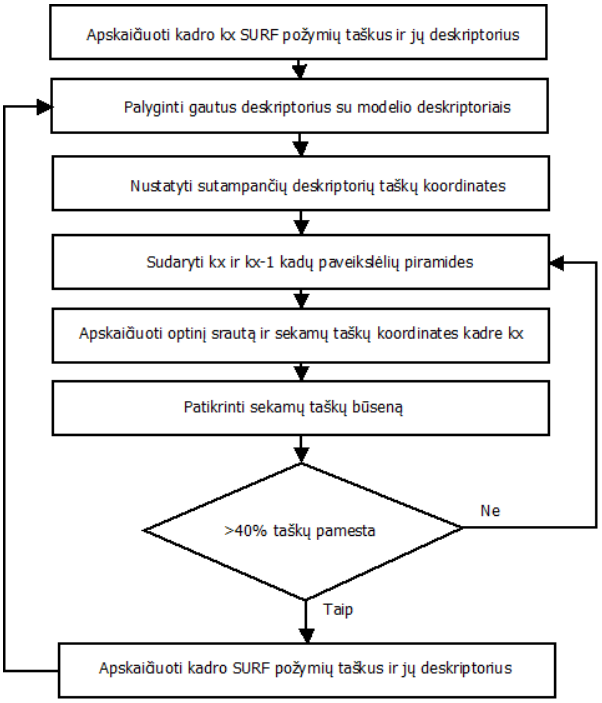


Paveikslėlis 17 Objekto sekimas CamShift metodu. Sekamas objektas (viršuje) ir kadro su sekamu objektu atgalinė projekcija (apačioje)

4.4. Lucas – Kanade optinio srauto algoritmo pritaikymas atpažinto objekto sekimui

Atpažintam objektui parinkus Lucas Kanade optinio srauto algoritmą, taikomi šie žingsniai (18 pav.):

1. Iš kameros paimtam kadrui k_x apskaičiuojami SURF požymiai ir jų deskriptoriai.
2. Gauti deskriptoriai palyginami su iš serverio gautais modelio deskriptoriais. Atpažinto objekto modelio deskriptoriai laikomi kliento pusėje, kad taškų tikslinimo metu nuolat nereiktų kreiptis į serverį. Nustatoma taškų, kurių deskriptoriai sutampa, koordinatės kadre.
3. Sudaromos esamo ir ankstesnio kadrų paveikslėlių piramidės, kurių aukštis 4 (žr. 2.7.2. skyrių).
4. Kadrų k_{x-1} ir k_x piramidėms naudojant Lucas – Kanade metodą, 2 žingsnyje surastiems taškams apskaičiuojamas optinis srautas ir taškų pozicija naujame kadre.
5. Tikrinama, ar taškai nėra pamesti.
6. Jei daugiau nei 40% sekamo objekto taškų „pamesta“, iš kadro k_x apskaičiuojami SURF požymiai ir jų deskriptoriai. Kartojamas 2 žingsnis.
7. Jei „pamesta“ mažiau nei 40% sekamo objekto taškų ar taškų nepamesta, kartojamas 3 žingsnis.



Paveikslėlis 18 Lucas-Kanade algoritmo taikymas SURF požymių sekimui

5. ATPAŽINTO OBJEKTO SEKIMO EKSPERIMENTAS

Pristatyto kombinuoto objektų sekimo algoritmas buvo realizuotas vaizdo atpažinimo sistemos Windows OS skirtam klientui. Realizuojant algoritmą buvo naudojamos OpenCV 2.1. bibliotekos funkcijos.

Algoritmo veikimo įvertinimui buvo padaryta keletas eksperimentų. Eksperimentams naudotas kompiuteris su Windows 7 operacine sistema, Intel Core i3 CPU, 4 GB operatyviosios atminties, objektas sekamas 640x480 raiškos (standartinė vaizdo atpažinimo sistemos raiška) iš kameros gaunamoje vaizdo sekoje. Atliekant eksperimentus objekto atpažinimo dalis buvo vertinama kaip „juoda dėžė“; dėmesys buvo kreipiamas tik į objekto sekimą.

Siekiant patikrinti tinkamiausio algoritmo parinkimo veikimą, eksperimentui buvo pasirinkti du objektai:

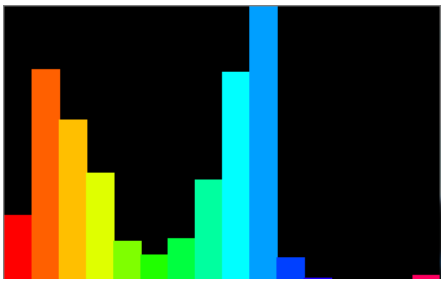
1. Nuotrauka;
2. Vienspalvė dėžutė.

Prieš eksperimentą buvo tikimasi, jog atpažintai nuotraukai algoritmas parinks Lucas-Kanade optinio srauto algoritmą, o vienspalvės dėžutės sekimui bus parinktas CamShift algoritmas.

Objektą sekant pagal Lucas-Kanade optinio srauto algoritmą, kiekviename kadre atvaizduojami sekami objekto taškai, kurie apibrėžiami žaliu stačiakampiu. Taikant CamShift algoritmą, aplink sekamą objektą brėžiama elipsė.

5.1. 1 eksperimentas. Sekimas naudojant Lucas – Kanade optinio srauto metodą su SURF taškų tikslinimu

Pirmas eksperimentas buvo atliekamas su 1 objektu (nuotrauka). Tinkamiausio algoritmo parinkimo metu nustatyta, jog objektui tinkamiausias Lucas – Kanade algoritmas. Šio objekto spalvų histogramoje, pateiktoje 19 paveikslėlyje, galima pastebėti, jog vienos ryškiai dominuojančios spalvos šiame objekte nėra, taigi tinkamiausias sekimo algoritmas parinktas teisingai.



Paveikslėlis 19 1 eksperimentas. Objekto spalvų histograma

Eksperimentas su šiuo objektu buvo kartojamas keletą kartų. Iš duomenų bazės gautas atpažinto objekto modelis iš viso turėjo 56 deskriptorius. Atpažintame objekte sekimo pradžioje vidutiniškai buvo aptikta 28 su modelio deskriptoriais sutampantys deskriptoriai, taigi buvo nustatyta vidutiniškai 28 sektini taškai. Visi pradiniai taškai buvo nustatyti objekto viduje. Eksperimento metu objektas buvo sekamas 45 sekundes 25 kadrų per sekundę greičiu.

Lucas – Kanade algoritmo eksperimentai buvo atlikti algoritmą taikant skirtingo aukščio paveikslėlio piramidei (žr. 2.7.2. skyrių). Pastebėta, jog algoritmo taikymui objekto sekimui vaizdo atpažinimo sistemoje geriausiai tinka 4 lygių paveikslėlio piramidė. Visi pateikiami rezultatai buvo gauti algoritme naudojant būtent 4 lygių paveikslėlio piramidę

20 ir 21 paveikslėliuose pateikiama pirmo eksperimento sekamų taškų skaičiaus kitimo laike diagramos – taikant sekamų taškų tikslinimą pagal objekto modelį (20 paveikslėlis) ir taškų tikslinimo netaikant (21 paveikslėlis; vienas ciklas iki taškų tikslinimo).

Kaip ir tikėtasi, eksperimento metu pastebėta, jog taikant Lucas – Kanade algoritmo piramidinę realizaciją sekamų taškų skaičius nuolat mažėja. Objektui judant tolygiai be didesnių transformacijų (pasukimų), taškų praradimas yra neženklus (žr. 20 pav. taškų skaičių iki ~745 kadro). Tačiau objektui judant intensyviau, sukantis 2D erdvėje; taip pat prie intensyvesnio foninio judėjimo sekamų taškų praradimas yra gana ryškus. Eksperimento metu per 34 sekundes buvo prarasta daugiau nei 50% sekamų taškų (21 paveikslėlis). Netaikant taškų patikslinimo iš objekto modelio, yra tikimybė, jog sekamas objektas bus pamestas.

Objektui intensyviai judant, iš dalies uždengus ar pasisukus erdvėje, „pamesti“ sekami taškai buvo atstatomi iš objekto modelio (sekami taškai nuolat patikslinami). Sekamų taškų skaičiaus kitimo laike diagramoje (20 paveikslėlis) matome, jog objektui judant tolygiai, taškų praradimas buvo nedidelis ir sekamų taškų tikslinimo nereikėjo. Sekamo objekto dalį laikinai uždengus, buvo pamesta didžioji dalis sekamų taškų, taigi buvo pritaikytas objekto taškų tikslinimas. (20 paveikslėlis, pirmas sekamų taškų skaičiaus padidėjimas 746 kadre). Matome, jog tikslinimo metu buvo nustatyta daugiau objekto modelį atitinkančių taškų, nei pradinio aptikimo metu objektą atpažinus (2 lentelė).

Lentelė 2 Sekamų taškų patikslinimų metu aptiktų taškų skaičius (1 eil. Nr. – pradinis aptikimas)

Eil. Nr	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Taškų skaičius	35	38	55	18	36	36	23	25

Lentelė 3. Objekto taškų sekimas Lucas – Kanade metodu taikant taškų patikslinimą

Taškų tikslinimo kadro nr. ***(n_1)***	**Tikslinimo metu aptiktų taškų skaičius** ***(t_1)***	**Kadro nr. prieš sekantį tikslinimą** ***(n_2)***	**Taškų skaičius prieš sekantį tikslinimą** ***(t_2)***	***t_1-t_2***	***n_2-n_1***
1	35	745	4	31	744
746	38	822	20	18	76
823	55	862	22	33	39
863	18	902	14	4	39
903	36	942	16	20	39
943	36	1005	8	22	62
1006	23	1051	21	2	45
1052	25	--	--	--	--

Eksperimento metu sekamų objekto taškų patikslinimas buvo pritaikytas 8 kartus. Kiekvieno tikslinimo metu buvo nustatytas skirtingas objekto modelį atitinkančių taškų skaičius (2 lentelė). Tai įtakojo objekto pasukimas, apšvietimo kitimas, objekto atstumas nuo kameros.

Tačiau taškų tikslinimas leido nuolat sekamų taškų kiekį išlaikyti priimtinam lygyje (> 8, išskyrus sumažėjimą 748 kadre, kai objektas buvo dalinai uždengtas) (3 lentelė, t_2) ir taip nepamesti sekamo objekto. Eksperimento metu objektui intensyviai judant, sekamų objekto tikslinimas buvo atliekamas kas 39-76 kadrus (3 lentelė, n_2-n_1) (vaizdą apdorojant 25 kadrų per sekundę greičiu, tai atitinka 1,5 – 3 sekundes). Objektui judant tolygiai, taškų tikslinimo neprireikė 744 kadrus (~30 sekundžių).

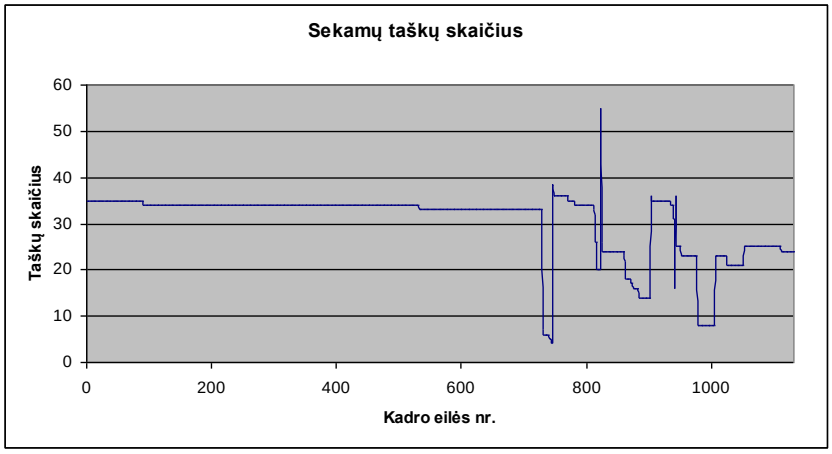
22, 23 ir 24 paveikslėliuose pateikiama pirmo eksperimento rezultatai – keletas kadrų iš vaizdo sekos. Paveikslėliuose sekami taškai pažymėti žaliai, visi taškai apibrėžti stačiakampiu, įvertinant pasukimą. Iš 22 paveikslėlio matome, jog pradiniai sektini objekto

taškai buvo parinkti visame objekto plote, tad sekamus taškus apibrėžiantis stačiakampis pakankamai tiksliai apibrėžė ir patį objektą (22 paveikslėlio 1-7 dalys). Kai kurie sekamo objekto taškai sekimo metu buvo „pamesti“, tačiau nuolatinis sekamų taškų tikslinimas pagal modelio deskriptorius lėmė tai, kad objektas ir toliau sėkmingai buvo sekamas (nors vieni taškai buvo pakeisti kitais) (20 paveikslėlio 7-8 dalys).

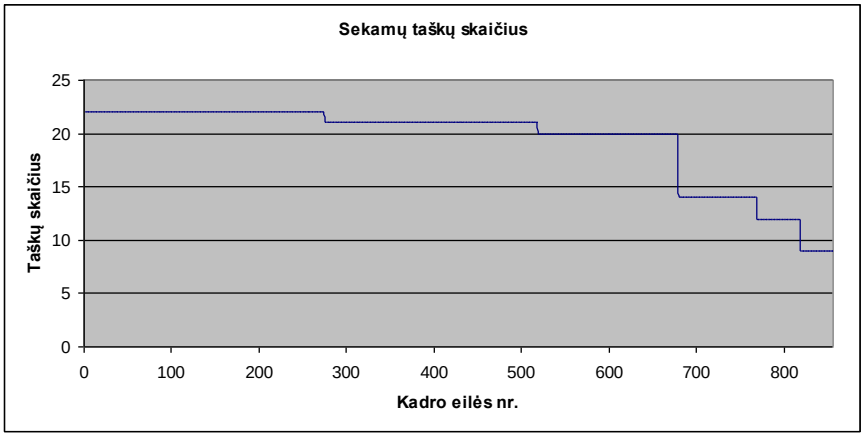
22 paveikslėlyje matome, jog objektas buvo sėkmingai sekamas tiek jį priartinus prie kameros (22 paveikslėlio 7 dalis), tiek atitolinus (22 paveikslėlio 8 dalis). Objekto taškai gana sėkmingai sekami ir objektą šiek tiek pasukus 3D erdvėje (22 paveikslėlio 10 dalis).

Iš 23 paveikslėlio matome, jog objektas sėkmingai sekamas ir jį pasukus 2D erdvėje.

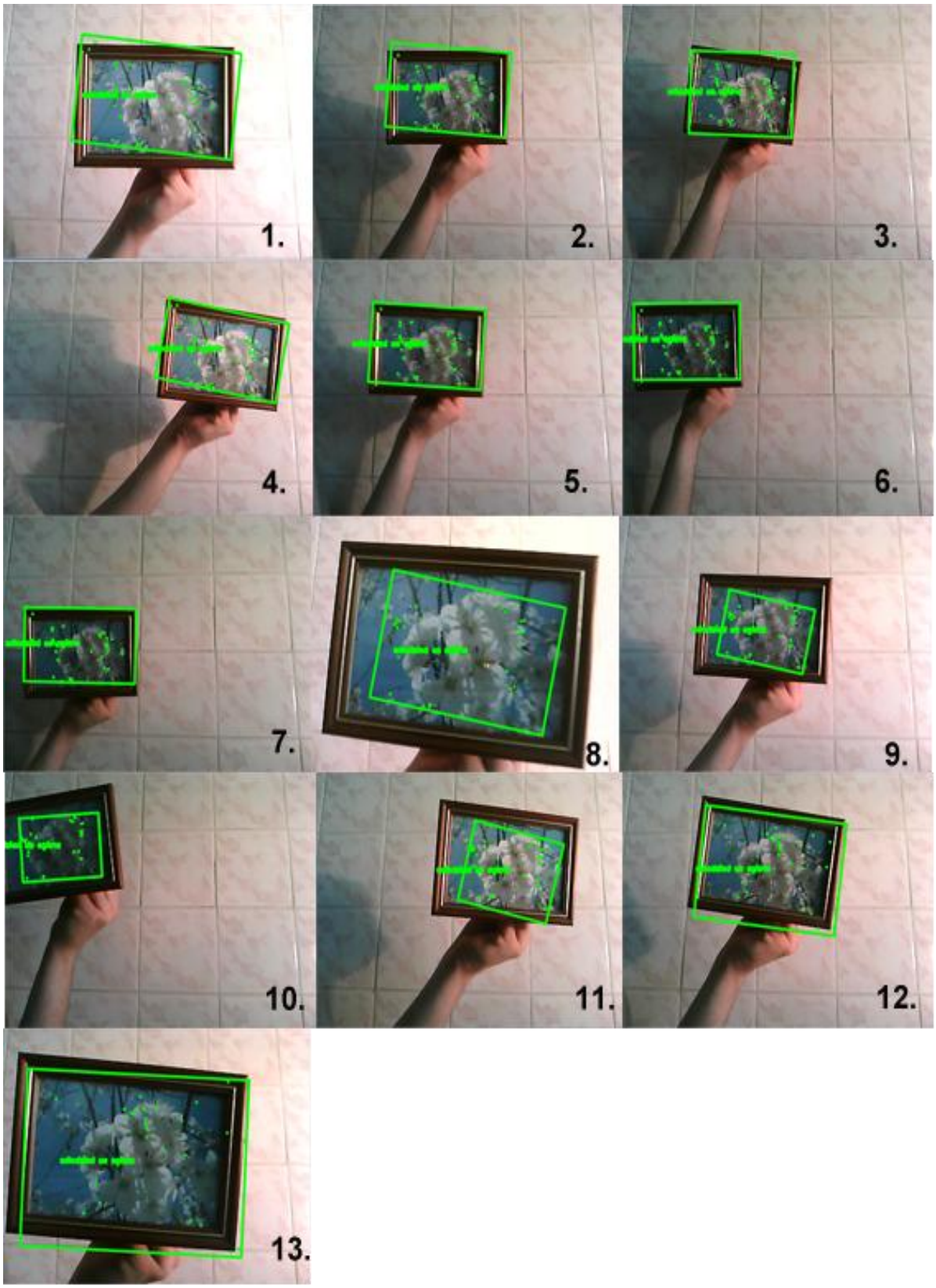
24 paveikslėlis gerai iliustruoja pakartotinį sekamų taškų tikslinimą. 24 paveikslėlio 1 dalyje pavaizduoti sekamo objekto taškai. Po keleto kadrų (24 paveikslėlio 2 dalis) šie taškai buvo „pamesti“ – 24 pav. antroje dalyje atvaizduota tik keli taškai, jie nėra objekto viduje. Esant šiai situacijai, buvo pritaikytas pakartotinis taškų tikslinimas – iš sekamo kadro išgauti SURF požymiai su deskriptoriais, palyginti su objekto modelio deskriptoriais, ir toliau sekami atitikę taškai (24 paveikslėlio 3-4 ir 5-6 dalys).



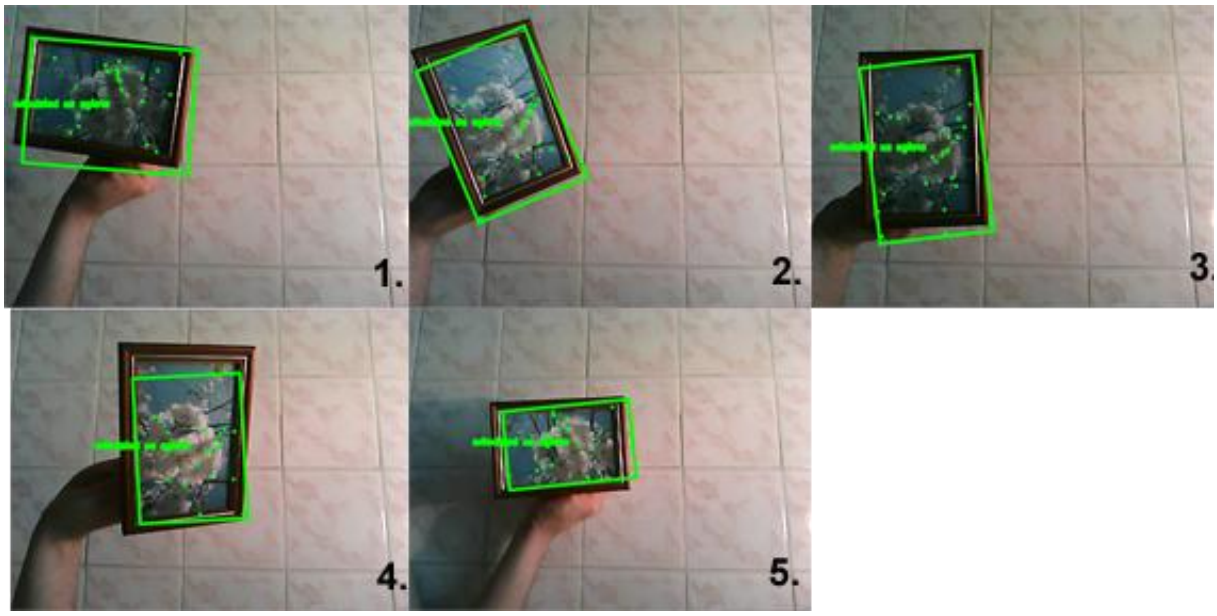
Paveikslėlis 20 Sekamų taškų skaičiaus kitimas laike taikant taškų patikslinimą



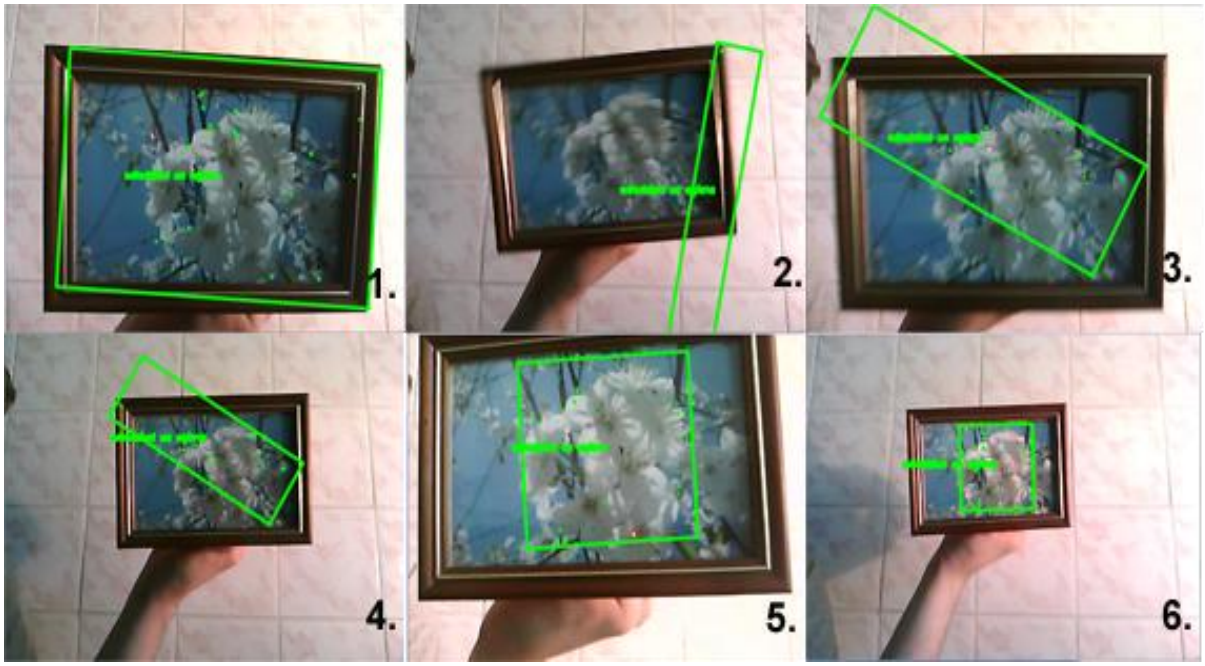
Paveikslėlis 21 Sekamų taškų skaičiaus kitimas laike



Paveikslėlis 22 1 eksperimentas. Atpažinto objekto sekimas taikant Lucas - Kanade algoritmą



Paveikslėlis 23 1 eksperimentas. Objekto 2D pasukimo įvertinimas

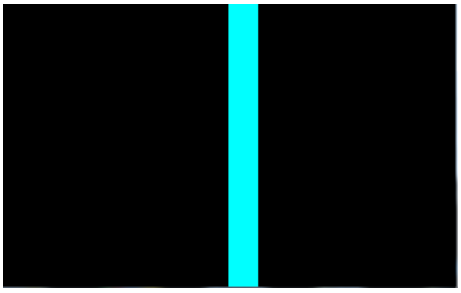


Paveikslėlis 24 1 eksperimentas. Pakartotinis taškų aptikimas

5.2. 2 eksperimentas. Sekimas naudojant CamShift metodą

Siekiant patikrinti algoritmo tinkamumą sekant vienspalvius objektus, antras eksperimentas buvo atliekamas su 2 objektu (vienspalve melsvos spalvos dėžute). Atpažįstant šį objektą buvo aptikti tik 4 unikalūs požymiai. Tinkamiausio algoritmo parinkimo metu buvo nustatyta, jog objekto sekimui labiau tinka CamShift algoritmas. Kaip matome iš šio objekto spalvų histogramos (25 paveikslėlis), šiame objekte akivaizdžiai dominuoja viena

spalva (mėlyna), taigi CamShift metodas šiam objektui yra tinkamas ir algoritmas buvo parinktas teisingai.



Paveikslėlis 25 3 eksperimentas. Objekto spalvų histograma

CamShift algoritmas eksperimento metu sekė objektą pagal mėlynos spalvos modelį (kadre buvo ieškoma mėlynos spalvos objektų). Sekamo objekto centras kadre apibrėžiamas raudonos linijos elipse. Eksperimento metu objektas buvo sekamas realiu laiku 3 minutes 25 kadrų per sekundę greičiu. 26, 27, 28 ir 29 paveikslėliuose pateikiama šio eksperimento rezultatai (keletas kadrų iš vaizdo sekos).

26 paveikslėlyje matome, jog mėlyna dėžutė buvo sėkmingai sekama prie gana ryškaus apšvietimo. Objektas buvo sėkmingai sekamas tiek judant plokštumoje (26 paveikslėlis, 1, 3, 6-9 dalys), tiek jį priartinus prie kameros (26 paveikslėlis, 2 dalis). Judant objektui, kartu judėjo ir jo šešėlis (1 – 3, 5 – 6, 8 – 9 dalys), taigi kadre esant keliems judesiams, algoritmas sėkmingai sekė atpažintą objektą.

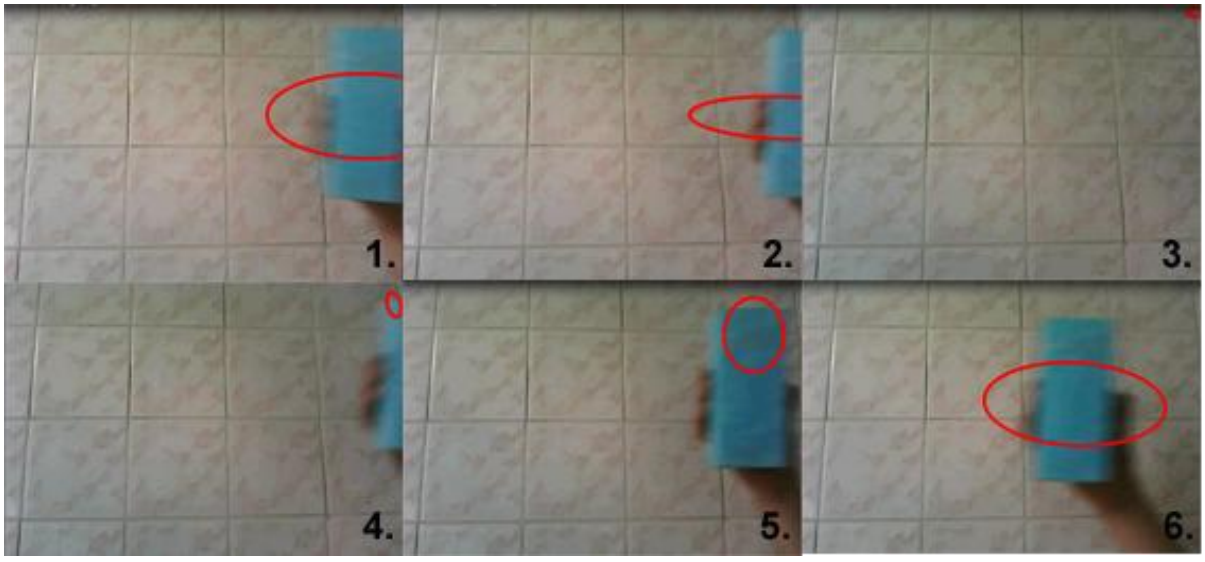
Iš 27 paveikslėlio matome, jog CamShift algoritmas sėkmingai seka iš kameros regos lauko laikinai dingusį ir vėl sugrįžusį objektą. Tačiau šio eksperimento metu pastebėta, jog CamShift algoritmas sugrįžusį objektą sėkmingai seka tada, kai objektas į kadrą grįžta tame pačiame krašte, kuriame iš kadro dingo. Objektui į kadrą sugrįžus kitoje kadro pusėje, realizuotas CamShift algoritmas jo neaptinka. Šiam atvejui reikalinga CamShift algoritmo modifikacija.

28 paveikslėlis iliustruoja algoritmo veikimą, kai objektas yra pakreipiamas ir pasukamas. Objektas buvo sėkmingai sekamas tiek jį pasukus horizontaliai, tiek vertikaliai (28 paveikslėlis, 1-9 dalys). Kadangi visas sekamas objektas turi dominuojančią mėlyną spalvą, jis buvo sekamas ir kamerai atsukus kitą objekto plokštumą (28 paveikslėlis, 11 dalis).

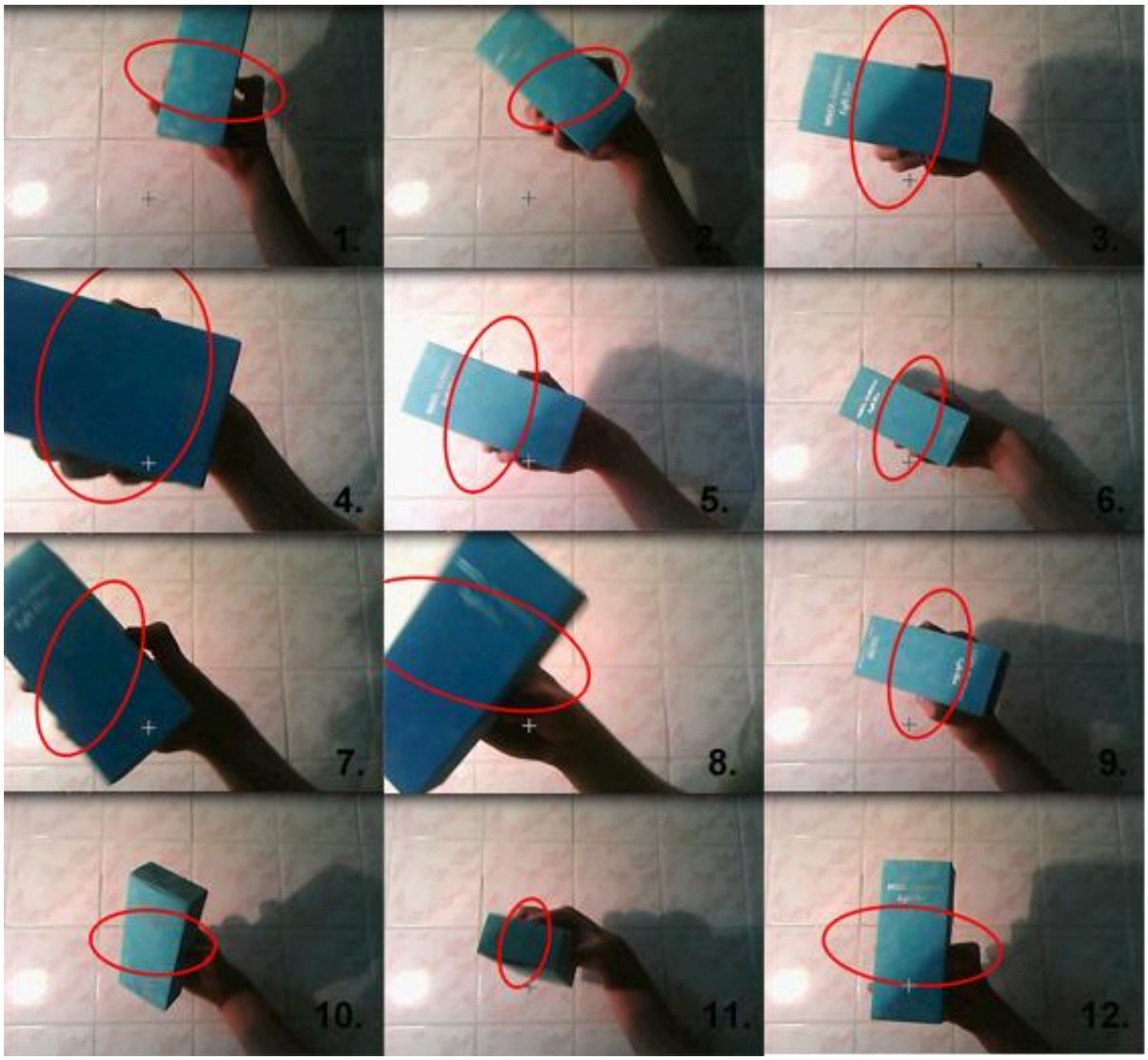
Eksperimentas įrodė, jog algoritmas atsparus daliniam sekamo daikto uždengimui (29 paveikslėlis). Sekamą objektą laikinai uždengus kitu judančiu daiktu, algoritmas nepradeda sekti uždengiančio objekto, bet toliau seka vėl į kameros regos lauką papuolusį atpažintą objektą.

Tačiau jei dengiantysis objektas yra panašios ar tokios pat spalvos, sekamas daiktas pametamas.

Paveikslėlis 26 Atpažinto objekto sekimas taikant CamShift algoritmą



Paveikslėlis 27 CamShift algoritmas. Objekto išėjimas iš kadro



Paveikslėlis 28 CamShift sekimas. Objekto pasukimas

Paveikslėlis 29 CamShift sekimas. Objekto uždengimas

5.3. Pastebėtos problemos ir galimi patobulinimai

Lucas – Kanade algoritmo taikymo eksperimento metu pastebėta, jog objekto pozicijos nustatymui pakanka sekti vos kelis unikalius to objekto taškus. Tačiau siekiant įvertinti objekto matmenis (ir pvz., tiksliai apibrėžti sekamą objektą) reikia, jog sekami unikalūs taškai būtų kuo arčiau objekto kraštinių.

SURF algoritmui unikalius taškus aptikus sekamo daikto viduryje taškus apibrėžiantis stačiakampis dažniausiai neapibrėžia viso sekamo objekto. Taikant prarastų taškų patikslinimą, kartais SURF algoritmas tam tikrus taškus aptinka objekto išorėje. Tokiu atveju apibrėžiama didesnis už sekamą objektą plotas, nors didžiausia sekamų taškų koncentracija išlieka pačiame objekte.

Siekiant išspręsti šią problemą, algoritmas gali būti tobulinamas, netoli nuo sekamus taškus apibrėžiančio stačiakampio ribų tiek stačiakampio viduje, tiek išorėje ieškant objekto briaunų ir taip patikslinant objekto plotą.

MeanShift algoritmo eksperimento metu įsitikinta, jog šis algoritmas puikiai tinka vienspalvio objekto sekimui, yra atsparus objekto daliniam ar visiškam uždengimui ir laikinam išėjimui iš kadro. Tačiau jei objektas sudarytas iš kelių skirtingų spalvų regionų (pvz. viena dalis vien tik mėlyna, kita – vien tik geltona), standartinis MeanShift algoritmas nėra tinkamas. Vaizdo atpažinimo sistemoje pristatytas tinkamesnio sekimo algoritmo parinkimo metodas tokiam objektui parenka Lucas – Kanade algoritmą. Tačiau SURF algoritmas, ieškantis sektinų taškų, tolygių spalvų regionuose unikalių taškų randa mažai (arba iš viso neranda).

Tokios problemos sprendimui MeanShift algoritmas ateityje gali būti patobulintas, atskirai sekant kiekvieną vienspalvį atpažinto objekto regioną bei tuos regionus apjungiant (apibrėžiant objektą apibrėžti visas atskiras spalvas žyminčas elipses). Panašus sprendimas siūlomas [24] straipsnyje.

6. IŠVADOS

1. Darbe pristatytas kombinuotas objektų sekimo metodas panaudojant du populiariausius objektų sekimo algoritmus - optinio srauto klasikinį Lucas – Kanade metodą ir CAMShif algoritmą.
2. Pristatytas kombinuotas objektų sekimo metodas sėkmingai panaudotas vaizdo atpažinimo sistemoje atpažintų objektų sekimui.
3. Eksperimento metu įsitikinta, jog tinkamesnio algoritmo parinkimas yra korektiškas.
4. Dinamiškas tinkamesnio algoritmo parinkimas kiekvienam atpažintam objektui leidžia išnaudoti šių algoritmų privalumus ir sekti skirtingų klasių objektus – tiek vienspalvius, tiek daugiaspalvius - be sudėtingo išankstinio sistemos apmokymo.
5. Atpažinus vienspalvį objektą, dinamiškas CAMShift algoritmo parinkimas leidžia sėkmingai sekti vienspalvius objektus.
6. Atlikto eksperimento metu įsitikinta, jog CAMShift algoritmas yra atsparus atpažinto objekto pasukimui, uždengimui, laikinam dingimui iš kadro ir apšvietimo pasikeitimui.
7. Nustatyta, jog CAMShift algoritmas nėra tinkamas atpažintų objektų su keliais skirtingų spalvų regionais sekimui, tačiau ateityje gali būti tam pritaikytas, atskirai sekant skirtingų spalvų regionus ir sekimo rezultatus apjungiant.
8. Vaizdo atpažinimo sistemai atpažinus daugiaspalvį objektą, SURF vietinių požymių nustatymo algoritmas puikiai tinka pradinės objekto pozicijos ir sektinų taškų nustatymui.
9. Eksperimento metu nustatyta, jog Lucas – Kanade metodas sėkmingai seka tolygiai judančio spalvoto objekto taškus, tačiau objekto judesiui suintensyvėjus, Lucas – Kanade metodas dalį sekamų taškų nuolat pameta.
10. Pritaikytas Lucas – Kanade metodas sėkmingai įvertina sekamo objekto pasukimą, atitolinimą ar priartinimą prie kameros.
11. Vaizdo atpažinimo sistemoje panaudotas sekamų taškų tikslinimas leidžia išlaikyti pakankamą sekamų taškų kiekį objektui intensyviai judant; panaudojant taškų tikslinimą Lucas – Kanade metodas tampa atsparus objekto uždengimui ir laikinam dingimui iš kameros regos lauko.
12. Sekant atpažinto objekto SURF algoritmu aptiktus taškus, ne visada teisingai įvertinamas sekamo objekto plotas (ne visada apibrėžiamas visas objektas). SURF algoritmas, priklausomai nuo objekto, atpažinto objekto modelį atitinkančius išorinius

unikalius taškus gali aptikti ne tik objekto pakraščiuose, bet ir sekamo daikto viduryje, arba išorėje. Pastebėta, kad bet kokiu atveju didžiausia taškų koncentracija visada išlieka objekto viduje. Siekiant tiksliau įvertinti sekamo objekto plotą, Lucas – Kanade algoritmas gali būti patobulintas netoli nuo sekamus taškus apibrežiančio staciakampio ribų tiek stačiakampio viduje, tiek išorėje ieškant objekto briaunų ir taip patikslinant objekto plotą.

13. Pristatytas kombinuotas metodas sėkmingai veikia realiu laiku 20 kadrų per sekundę greičiu 640x480 vaizdo sekoje.

7. LITERATŪRA

1. Huiyu Zhou, Yuan Yuan, Chunmei Shi „Object tracking using SIFT features and mean shift“ Computer Vision and Image Understanding 113 (2009) 345–352
2. „SURFTrac: Efficient Tracking and Continuous Object Recognition using Local Feature Descriptors“ Duy-Nguyen Ta, Wei-Chao Chen, Natasha Gelfand, Kari Pulli, IEEE, 2009, 2937-2944
3. „SURF Tracking“ Wei He, Takayoshi Yamashita, Hongtao Lu, Shihong Lao, 2009 IEEE 12th International Conference on Computer Vision (ICCV), 1586 – 1592
4. Simon Denman, Clinton Fookes, Sridha Sridharan „Improved Simultaneous Computation of Motion Detection and Optical Flow for Object Tracking“, 2009 Digital Image Computing: Techniques and Applications, 2009, 175-182
5. „Object Tracking: A Survey“ Alper Yilmaz, Omar Javed, Mubarak Shah, ACM Computing Surveys, Vol. 38, No. 4, Article 13, 2006.
6. „Open Source Computer Vision Library. Reference manual“ 1999-2001 Intel Corporation
7. OULD-DRIS Nouar, GANOUN Ali, CANALS Raphaël „Imoproved object tracking with CamShift algorithm“, ICASSP 2006, 2006, 657-660
8. „Speeded-Up Robust Features (SURF)“, Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, Computer Vision and Image Understanding 110 (2008) 346–359
9. Liudas Drejeris „Turiniu paremtas paveiksleliu atpažinimas panaudojant SURF (Speeded-Up Robust Features) algoritmą ir skaidymą spalvomis“, 2012
10. "Optical flow using color information: preliminary results", Kelson R. T. Aires, Andre M. Santana, Adelardo A. D. Medeiros
11. J.L. Barron, D.J. Fleet, S.S. Beauchemin „Performance of Optical Flow Techniques“ IJCV 12:1, 1994, 43-77
12. J.-Y. Bouguet. "Pyramidal implementation of the Lucas Kanade feature tracker". OpenCV Documentation, Intel Corporation, Microprocessor Research Lab, 1999.
13. "Hybrid Feature Tracking and User Interaction for Markerless Augmented Reality" Taehee Lee,Tobias Hollerer
14. „Server-side object recognition and client-side object tracking for mobile augmented reality“ Stephan Gammeter, Alexander Gassmann, Lukas Bossard, Till Quack, Luc Van Gool, IEEE, 2010
15. Ernestas Kardzys „Vaizdo Sistemos pritaikymas objektų klasifikavimui“, 2012

16. OpenCV [žiureta 2012 03 10], prieiga per interneta: http://opencv.willowgarage.com/wiki/FullOpenCVWiki
17. Lockton, R., Fitzgibbon, A.W. „Real-time gesture recognition using deterministic boosting“, 2002, „Proceedings of the British Machine Vision Conference“, vol. II, pp. 817–826
18. Shinya Yamamoto, Yasushi Mae, Yoshiaki Shirai, Jun Miura „Realtime Multiple Object Tracking Based on Optical Flows“, IEEE International Conference on Robotics and Automation (1995), 2328-2333
19. Koichiro Deguchi, Oki Kawanaka, Takayuki Okatani „Object tracking by the mean-shift of regional color distribution combined with the particle-filter algorithm“, Proceedings of the 17th International Conference on Pattern Recognition (ICPR’04), 2004
20. Changjiang Yang, Ramani Duraiswami, Larry Davis „Fast Multiple Object Tracking via a Hierarchical Particle Filter“, Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05), 2005
21. Lei Sun, Bingrong Wang, Takeshi Ikenaga „Real-time Non-rigid Object Tracking Using CAMShift with Weighted Back Projection“, 2010 International Conference on Computational Science and Its Applications, 2010, 86-91
22. Shengluan Huang, Jingxin Hong „Moving Object Tracking System Based On Camshift And Kalman Filter“, IEEE, 2011, 1423-1426
23. J. Kovacevic, S. Juric-Kavelj, I. Petrovic „An Improved CamShift Algorithm Using Stereo Vision For Object Tracking“, Mipro 2011, May 23-27, 2011, Opatija, Croatia, 707-710
24. P. Hidayatullah, H. Konik „Camshift improvement on multi-jue object and multi-object tracking“, EUVIP 2011, 2011, 143-148
25. „Handling occlusion in optical flow algorithms for object tracking“, Eduardo Parrilla, Damian Ginestar, Jose Luis Hueso, Jaime Riera, Juan Ramon Torregrosa, Computers And Mathematics with Applications 56 (2008), 733-742

8. TERMINŲ IR SANTRUMPŲ ŽODYNAS

2D (dvimatis) – dviejų dimensijų (pločio ir aukščio).

2D + t – dviejų dimensijų (pločio ir aukščio), įvertinant laiko momentą t.

3D (trimatis) – trijų dimensijų (pločio, aukščio, ilgio).

Android OS – mobiliesiems irenginiams skirta operacine sistema, paremta Linux branduoliu.

Deskriptorius - skaičių vektorius, unikaliai aprašantis tam tikrą tašką, apskaičiuotą vienu iš vietinių požymių nustatymo algoritmų (SIFT, SURF ar kitą).

HSV – (angl. *Hue – saturation – value,* spalva – sodrumas - ryškumas) tarptautiniu mastu pripažinta, viena tiksliausių ir lengviausiai suvokiamų spalvų kodavimo sistemų.

JSON - JavaScript Object Notation.

OpenCV - C programavimo kalba parašyta atvirojo kodo biblioteka, skirta skaitmeniniu vaizdu apdorojimui. Laisvai platinama pagal BSD licenzija.

Paveikslėlių piramidė – vieno kadro skirtingo dydžio paveikslėlių rinkinys, kur k+1 lygio paveikslėlis yra dvigubai mažesnių matmenų už k lygio paveikslėlį.

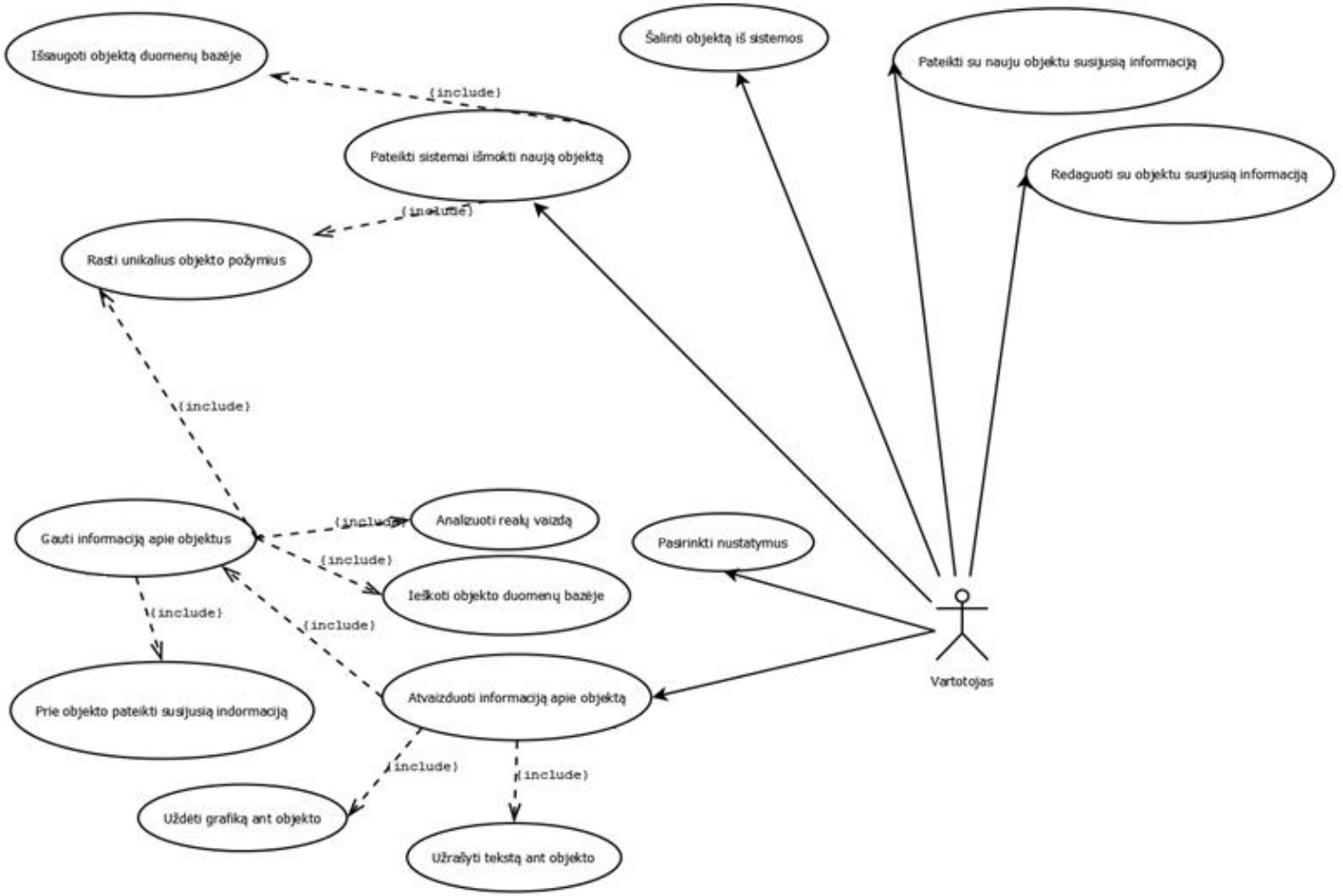
Praturtintosios realybės sistema – „Augmented Reality“ (Praturtinta realybe). Sistemos, kurios mus supancia aplinka papildo tikroveje neegzistuojanciais užrašais, objektais, garsais ir pan. Taip i mus supancia aplinka pridedama (papildoma) naujos informacijos.

QT – viena populiariausiu ir galingiausiu atvirojo kodo biblioteku vartotojo sasajai kurti, turinti daug darbo su grafika, prisijungimais prie serverio (networking) ir kitu galimybiu.

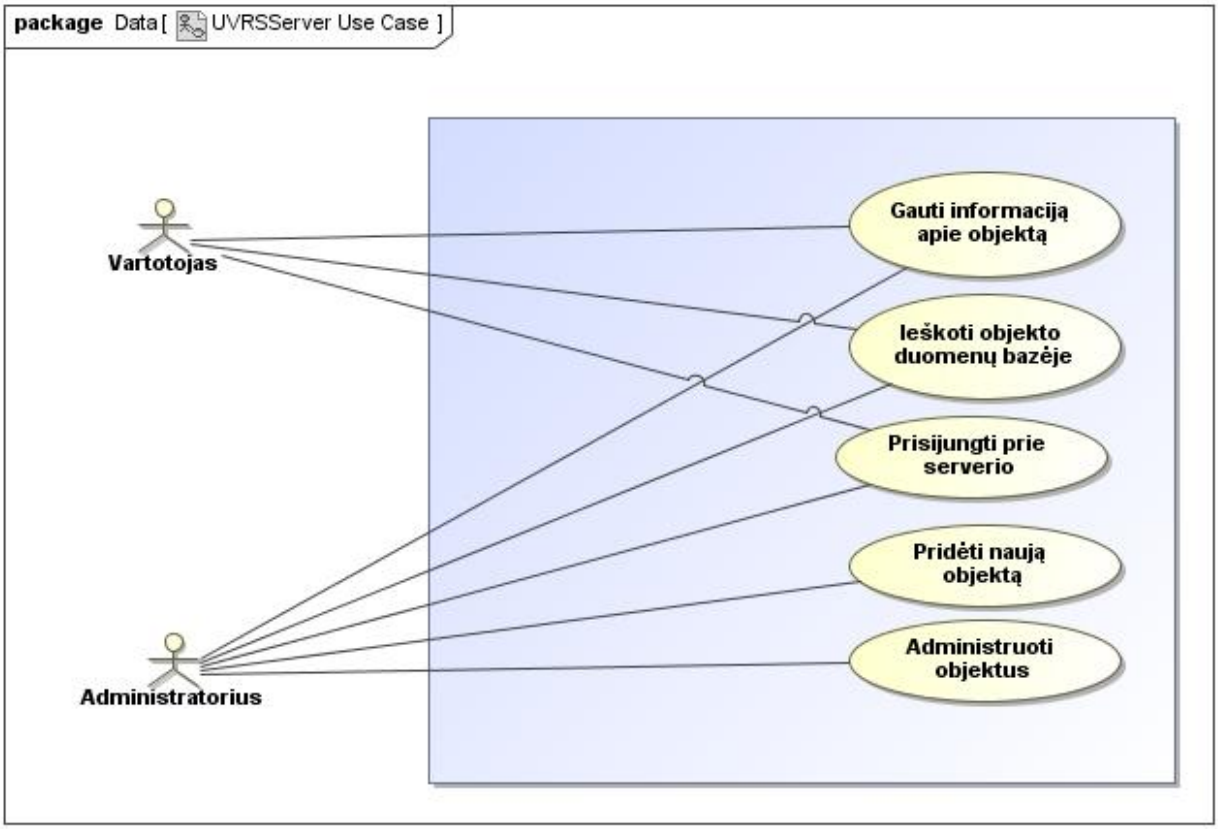
SURF - (angl. *Speeded-Up Robust Features*) nuo mastelio ir pasukimo nepriklausantis požymių išskyrimo ir deskriptorių sudarymo algoritmas.

9. PRIEDAI

9.1. 1 priedas. Sistemos kliento panaudos atvejai



9.2. 2 priedas. Sistemos serverio panaudos atvejai



9.3. 3 priedas. JSON žinučių pavyzdžiai

9.3.1. Objekto atpažinimo užklausa


```
{"data":{"action-desc": { "action": "select-object" }, "object-data":
{"id": "0", "type": "book", "descriptors":"!0;75.1821;-
1;4222.27;423.816;190.75;16;6.9833e-
007;0.000437111;0.000101985;0.000101985;0.000104188;0.000211425;0.000155246
;0.000155246;0.0189237;0.0201748;0.000616491;0.000616491;0.00522119;0.00558
453;0.0015639;0.0015639;0.0215814;0.0322698;0.00688649;0.00688649;0.0118573
;0.0128071;0.00192341;0.00192341;-
0.00116454;0.00116454;0;0;0;0;0.000738278;0.000738278;0.00271695;0.00859577
;-3.59856e-005;3.59856e-
005;0.00869906;0.00869906;0.00221265;0.00228462;0.225564;0.225564;0.0140802
;0.0155653;0.143952;0.149519;-0.00024752;0.00024752;-
0.293524;0.339232;0.283695;0.337705;-
0.0860634;0.138645;0.0967106;0.127485;-0.00676715;0.00676715;-
0.000759315;0.000759315;0;0;0.00276351;0.00292706;!<tolimesni
deskriptoriai>
"}}}
```


9.3.2. Atpažinto objekto informacijos žinutė


```
"{
"data":
{
"message-desc":
{
"type": "object-recognized-message"
},
"message-body":
{
"message": "Object recognized!",
"id": "72",
"type": "book",
"title": "Title",
"description": "Description",
"descriptors":"!0;75.1821;-1;4222.27;423.816;190.75;16;6.9833e-
007;0.000437111;0.000101985;0.000101985;0.000104188;0.000211425;0.000155246
;0.000155246;0.0189237;0.0201748;0.000616491;0.000616491;0.00522119;0.00558
453;0.0015639;0.0015639;0.0215814;0.0322698;0.00688649;0.00688649;0.0118573
;0.0128071;0.00192341;0.00192341;-
0.00116454;0.00116454;0;0;0;0;0.000738278;0.000738278;0.00271695;0.00859577
;-3.59856e-005;3.59856e-
005;0.00869906;0.00869906;0.00221265;0.00228462;0.225564;0.225564;0.0140802
```

;0.0155653;0.143952;0.149519;-0.00024752;0.00024752;-
0.293524;0.339232;0.283695;0.337705;-
0.0860634;0.138645;0.0967106;0.127485;-0.00676715;0.00676715;-
0.000759315;0.000759315;0;0;0.00276351;0.00292706;!<tolimesni
deskriptoriai>
}
}
}
"

9.3.3. Neatpažinto objekto informacijos žinutė

"{
"data":
{
"message-desc":
{
"type": "error-message"
},
"message-body":
{
"message": "Sorry, object not recognised!"
}
}
}
"