VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ STUDIJŲ PROGRAMA

Stalo teniso žaidimo rezultato sekimas

Tracking Result of Ping Pong Game

Magistro baigiamasis darbas

Atliko:	Giedrius Visokinskas	(parašas)
Darbo vadovas:	partn. prof. dr. Vytautas Ašeris	(parašas)
Recenzentas:	dr. Vytautas Valaitis	(parašas)

Vilnius – 2022

Anotacija

Šis magistro baigiamasis darbas nagrinėja galimybės realiu laiku analizuoti stalo teniso žaidimo eigą mobiliajame telefone. Tiriamos greitai judančių objektų atpažinimo, trajektorijų ekstrapoliacijos bei stalo teniso žaidimo modeliavimo problemos. Tyrimo metu sukurtas algoritmas, gebantis atpažinti stalo teniso žaidimo eigą esant idealiomis sąlygomis bei pavyzdinis įgyvendinimas „iOS“ operacinėje sistemoje. Nustatyta, kad stalo teniso žaidimo sekimo algoritmą įmanoma įgyvendinti „Apple iPhone 12 Pro“ mobiliajame telefone, tačiau norint pasiekti tvarius rezultatus dinamiškoje aplinkoje, reikia gebėti atsižvelgti į žaidėjų sudaromą „triukšmą“, pavyzdžiui, kamuoliuko mušinėjimą į stalą prieš jį paduodant.

Raktiniai žodžiai: stalo tenisas, kompiuterinės regos sistema, trajektorijų ekstrapoliacija, konvoliuciniai neuroniniai tinklai

Summary

This master thesis analyses the ability to track a table tennis match in real time using a mobile phone. The thesis focuses on the detection of fast-moving objects, trajectory extrapolation, and the modeling of a table tennis match. An algorithm is proposed that is able to track a table tennis match under ideal circumstances. A proof of concept implementation for the iOS operating system is provided. It was determined that, although it is possible to implement an algorithm that can track a table tennis game in "Apple iPhone 12 Pro", a more sophisticated solution is needed in order to filter out the noise that is created, for example, by players hitting the ball on the table before serving.

Keywords: table tennis, computer vision system, trajectory extrapolation, convolutional neural networks

TURINYS

Įvadas

Didžiausias užfiksuotas stalo teniso kamuoliuko greitis yra 70 mylių per valandą [Let19], o stalo teniso stalo ilgis yra 2 metrai 74 centimetrai [Fed21]. Atsižvelgiant į tai, galima daryti prielaidą, kad stalo teniso teisėjai gali padaryti klaidų vertindami žaidimo eigą. Nors jau egzistuoja sistemos, galinčios sekti stalo teniso žaidimą trimatėje erdvėje [ITT19], taip padėdamos teisėjams priimti sprendimus, jos nėra pritaikytos asmeniniam naudojimui ir reikalauja specifinio pasiruošimo.

Mobiliesiems įrenginiams spartėjant [Tri15], atsiranda vis daugiau galimybių įgyvendinti nešiojamas kompiuterinės regos sistemas, gebančias sekti, pavyzdžiui, krepšinio [Inc22b] ar lauko teniso [Inc22a] žaidimo eigą. Tikimasi, kad šio tyrimo rezultatas galėtų būti pritaikomas padėti automatizuoti stalo teniso žaidimo rezultato vertinimą mėgėjiškoje aplinkoje, naudojantis vien tik mobiliuoju telefonu.

Nors stalo tenisas yra minimas greitai judančių objektų atpažinimo tyrime [Hra17], o kitame tyrime [Old19] yra pateikiamas sprendimas atpažinti įvykius stalo teniso žaidimo metu, tačiau tiek akademinėje, tiek komercinėje aplinkoje, nėra sprendimo, įgyvendinto mobiliajame telefone, kuris gebėtų sekti viso žaidimo būklę, o ne vien pavienius įvykius žaidimo metu.

Dėl mažo kamuoliuko dydžio bei didelio greičio, yra sunku tiksliai nustatyti kamuoliuko padėtį erdvėje bei ekstrapoliuoti galimą trajektoriją. Išskirtiniais atvejais kamuoliukas gali perskristi stalą per 87.5 milisekundes [Fed21; Let19]. Šį uždavinį apsunkina ir tai, kad norima žaidimą sekti naudojantis tik viena (mobiliajame telefone esančia) kamera, dėl to nėra galimybės sukurti tikslios kamuoliuko projekcijos trimatėje erdvėje. Tyrimą apsunkina ir skirtingos aplinkos, kuriose galima žaisti stalo tenisą. Siekiant suvaldyti tyrimo apimtį, įvedami šie ribojimai:

- darbe yra tiriamos tik asmenines (kai prie stalo žaidžia du žaidėjai) stalo teniso partijos, žaidimas poromis nėra tiriamas;
- tikimasi, kad žaidėjai yra bent dalinai susipažinę su oficialiosiomis stalo teniso taisyklėmis [Fed21];
- taisyklių [Fed21] išimtys, pavyzdžiui, neįgaliesiems, nėra įtraukiamos į tyrimą;
- tiriamos partijos, kuriose žaidžiama naudojantis taisykles [Fed21] atitinkantį inventorių.

Darbo tikslas

Šio rašto darbo tikslas – pasiūlyti ir sukurti algoritmą, įgalinantį stalo teniso žaidimo sekimą bei įrodyti, kad toks algoritmas gali būti naudojamas sekti mėgėjiškas stalo teniso rungtynes.

Darbo uždaviniai

Šiam tikslui pasiekti keliami tokie uždaviniai:

1. Atlikus literatūros apžvalgą, įvertinti:
    - reikalavimus, keliamus techninei įrangai;
    - algoritmus, skirtus užfiksuotam kadrui apdoroti;
    - algoritmus, gebančius atpažinti stalo teniso kamuoliuką;

    - galimybes atpažinti statinius objektus kadre;
    - egzistuojančius programinės įrangos karkasus, palengvinančius įgyvendinimą.
2. Sudaryti stalo teniso duomenų rinkinį, skirtą algoritmui įgyvendinti bei testuoti.
3. Remiantis atliktu tyrimu, pasiūlyti algoritmą, galintį sekti stalo teniso žaidimą bei rezultatą.
4. Pagal stalo teniso federacijos taisykles sukurti būsenų automatą, kuriuo naudojantis galima vertinti žaidimo eigą.
5. Pagal gautas tyrimo išvadas įgyvendinti pavyzdinę mobiliosios programėlės realizaciją.
6. Sudaryti kriterijus, kuriais remiantis galima įvertinti algoritmo efektyvumą.
7. Testuoti algoritmo efektyvumą pavyzdinės mobiliosios programėlės realizacijoje.
8. Įvardyti algoritmo trūkumus bei pasiūlyti tolesnes tyrimų kryptis.

Tyrimas atliekamas naudojantis „Apple iPhone 12 Pro“, šis įrenginys buvo pasirinktas, nes:

- geba filmuoti 240 kps greičiu [App22c], tai yra svarbu, nes norint užtikrinti užfiksuoto vaizdo ryškumą, reikia parinkti kamerą, galinčią filmuoti bent 120 kadrų per sekundę greičiu [Luk16];
- palaiko kompanijos „Apple“ sukurtą „Vision“ [App17] karkasą, palengvinantį pavyzdinį įgyvendinimą;
- turi aparatinį neuroninių tinklų akceleratorių [App22c], didinantį algoritmo našumą;
- atitinką literatūros analizės metu iškeltus reikalavimus kamerai.

1. Literatūros apžvalga

Šiame skyriuje yra apžvelgiami darbai, susiję su kompiuterine rega, objektų klasifikavimu, greitai judančių objektų atpažinimu.

Analizėje taip pat apžvelgiamas ir kompanijos „Apple“ 2017 metais „WWDC“ renginio metu [App17] pristatytas „Vision“ karkasas, naudojamas atpažinti ir identifikuoti objektus tiek nuotraukose, tiek ir realaus laiko vaizdo sraute.

Literatūros analizės rezultatai yra naudojami nustatyti magistrinio darbo techninius reikalavimus, eksperimentus bei algoritmus, kuriais remiantis galima vykdyti tolesnį tyrimą apie stalo teniso žaidimo sekimą mobiliajame telefone.

Literatūros apžvalga yra suskirstyta į šias dalis:

1. Techninė įranga - techninių reikalavimų įrangai, skirtai atpažinti objektus kadre, aptarimas. Šiame skyriuje aptariami literatūros darbuose identifikuoti reikalavimai techninei įrangai: kadro raiškai, kamerų kiekiui, kadrų per sekundę skaičiui.
2. Kadro paruošimas - metodų ir algoritmų, kuriais naudojantis galima paruošti kadrą, siekiant atlikti objektų identifikavimą, aptarimas, remiantis literatūros analize.
3. Kamuoliuko atpažinimas - remiantis literatūra, pateikiami skirtingi būdai, kuriais naudojantis galima nustatyti kamuoliuko judėjimą, atpažinti jį bei nustatyti jo padėtį erdvėje.
4. Statinių objektų atpažinimas - aptariami skirtingi būdai, kuriais naudojantis galima identifikuoti objektus, kurie kadre nesikeičia, bet yra svarbūs stalo teniso žaidime, pavyzdžiui stalas, tinklelis.
5. „Vision“ karkaso panaudojimas - apžvelgiami būdai, kaip galima atlikti stalo teniso žaidimo sekimą naudojantis kompanijos „Apple“ sukurtu „Vision“ karkasu mobiliesiems įrenginiams.
6. Išvados - analizės rezultatų apibendrinimas bei išvados.

1.1. Techninė įranga

Viena svarbiausių kompiuterinės regos dalių - techninė įranga, naudojama užfiksuoti kadrui. Dėl stalo teniso žaidimo dinamiškumo svarbu užtikrinti, kad kamuoliukas būtų nesusiliejęs, stalas išsiskirtų iš kadro fono, būtų galima atpažinti žaidėjų judesius. Priklausomai nuo pasirinkto sensoriaus (ir jo parametrų) galima išvengti ir artefaktų užfiksuotose kadruose. Vaizdo fiksavimą galima atlikti su viena kamera [Old15; Rad18] arba su kelių kamerų sistema [LH19]. Išsiliejimo, atsirandančio prie mažo KPS skaičiaus, efektą, galima sumažinti naudojantis „VOCUS“ sistemą [JZS+18].

1.1.1. Kadro fiksavimas viena kamera

Pirminio kadro fiksavimą galima atlikti su paprastam vartotojui prieinamomis (ne profesionaliomis) kameromis. Lyginant šiuos tris įrenginius [Rad18]: „Venus” WEB kamerą, „SONY NEX-C3” fotoaparatą, „Apple iPhone 8” telefoną, „Venus” WEB kamera turi prasčiausias techninės charakteristikas, tačiau gali būti tiesiogiai prijungiama prie kompiuterio siekiant pagreitinti

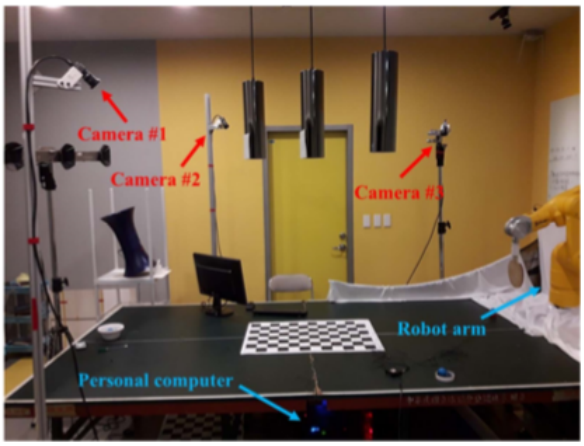
testavimo procesą. Iš likusių „SONY NEX-C3" bei „Apple iPhone 8" įrenginių labiau tinkamas vaizdo apdorojimui yra „Apple iPhone 8", nes šis mobilusis telefonas geba filmuoti iki 240 KPS (kadrų per sekundę) skaičiumi. Naudojantis „iPhone 8" galima atpažinti 84.4% nelegalių padavimų. Svarbu paminėti, kad kadrų per sekundę skaičius turi didelę įtaką norint išvengti vaizdo netikslumo ar triukšmo kadre.

Galima teisingai atpažinti 95.9% stalo teniso žaidimo įvykių naudojantis tik viena, paprastam vartotojui prieinama, kamera [Old15]. Likusių 4.1% atvejų negalima nustatyti, nes naudojant tik vieną kamerą, žaidėjai gali uždengti kamuoliuką arba kitus svarbius (stalą, tinklelį) kadre esančius objektus. Siekiant užtikrinti kadro atpažinimo kokybę su viena kamera, reikia, kad:

- kamera gebėtų greitai sufokusuoti judančius objektus kadre, vaizdas nebūtų išblukęs;
- kamera turėtų didelį užrakto greitį, siekiant sumažinti kamuoliuko iškraipymą, tačiau kameros sensorius turėtų būti ir pakankamai jautrus, nes didėjant užrakto greičiui, mažėja šviesos, patenkančios į sensorių, kiekis;
- kamera turėtų galimybę fiksuoti vaizdus dideliu KPS skaičiumi, geriausiu atveju tai turėtų būti 420Hz (arba KPS), tačiau geri rezultatai yra gaunami ir fiksuojant 100-200Hz dažniu;
- kamera turėtų turėti platų matymo kampą, kad būtų galima kamerą padėti arčiau stalo teniso stalo, tačiau ne per didelį, kad neatsirastų vaizdo iškraipymai.

Lyginant šiuos aštuonis prietaisus [Old15]: „Olympus E-PL1™",„Photron FastCam™ Ultima APX", „Canon HG10", „Sony HDR-AS15", „Casio Exilim™ ZR-300", „Philips ToUcam™ II", „Apple iPhone™ 5" bei „Nokia Lumia™ 900" - nustatyta, kad mobilieji telefonai ir WEB kameros negali atlikti atpažinimo realiu laiku, nes jie nesugeba greitai fiksuoti kadrų (neturi didelio KPS skaičiaus). Svarbu paminėti, kad šiandien rinkoje jau galima rasti mobiliųjų telefonų, galinčių filmuoti bent 100 KPS greičiu. Vienas iš tokių telefonų - „Apple iPhone 12 Pro" [App22c], kuris taip pat atitinka ir visus nustatytus reikalavimus kamerai.

1.1.2. Kadro fiksavimas keliomis kameromis

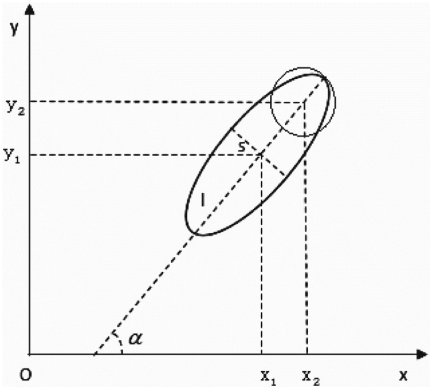


1 pav. Trijų kamerų išdėstymas [LH19]

Siekiant išvengti atvejų, kai žaidėjai uždengia kadrą ir neleidžia algoritmui atpažinti stalo teniso įvykių, galima naudoti kelių kamerų sistemą [LH19]. Galima naudoti 3 „IDS" tipo ka-

meras, gebančias filmuoti spalvotai, su 1.3 megapikselių raiška (1 pav.), 169 KPS greičiu. Kelių kamerų panaudojimas leidžia sukurti trimatę projekciją ir identifikuoti kamuoliuko padėtį erdvėje. Tačiau, naudojantis kelias kameras, reikia užtikrinti įvesties iš daugelio kamerų vienu metu sinchronizaciją. Svarbu, kad kameros taip pat turėtų ir didelį KPS skaičių, kad kamerų fiksuoti vaizdai būtų neišsilieję. Visgi, ši trijų kamerų sistema yra per daug sudėtinga paprastiems vartotojams, nėra tikėtina, kad ji gali būti panaudojama be kruopštaus pasirengimo. Tikint, kad 95.9% stalo teniso žaidimo įvykių galima atpažinti naudojantis tik viena kamera [Old15], šis būdas nėra naudingas siekiant sukurti paprastam vartotojui skirtą stalo teniso žaidimo atpažinimo sistemą.

1.1.3. Mažo KPS skaičiaus artefaktų šalinimas



2 pav. Ryšys tarp išsiliejusio kamuoliuko bei jo realaus centro [JZS+18]

Nors apžvelgti darbai akcentuoja didelį KPS skaičių, tačiau pasirinkus kamerą, kuri negali filmuoti didesniu nei 30 KPS greičiu, galima pritaikyti algoritmą [JZS+18], skirtą atlikti greitai judančių objektų atpažinimą su mažo KPS skaičiaus kamera. Šis algoritmas remiasi „VOCUS“ sistema, kuri gali identifikuoti dėmesio taškus kadre naudodama spalvas, iš kurių kadras susideda (panašiai į tai, kaip tai atlieka žmogus). Toks „VOCUS“ sistemos panaudojimas leidžia apskaičiuoti tikslų kamuoliuko centrą net ir esant artefaktams nuotraukoje. 2 pav. pateikiamas ryšys tarp kamuoliuko realaus centro bei to, kas yra užfiksuota kadre, filmuojant su mažo KPS kamera. Nors šis algoritmas geba atpažinti kamuoliuką ribotame kiekyje atvejų, svarbu paminėti, kad būtų galima išvengti papildomų skaičiavimų, jeigu pradinis vaizdas nebūtų išsiliejęs ir būtų užfiksuotas greita kamera. Taip pat technologijos, naudojamos šiame sprendime nėra pakankamai pažengusios ir yra tikėtina, kad jos negalėtų būti pritaikomos esant kitokioms aplinkybėms.

1.1.4. Techninės įrangos apibendrinimas

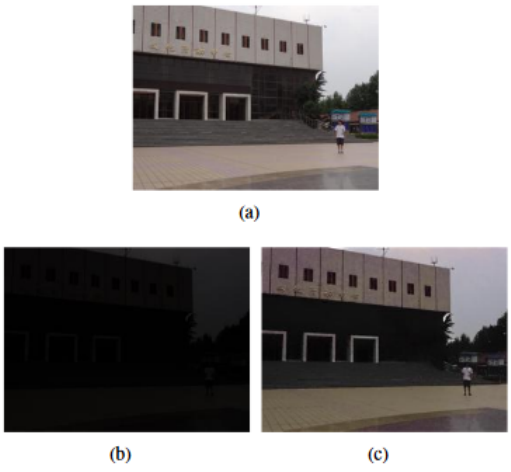
Ištyrus literatūros šaltinius galima teigti, kad svarbiausia turėti kamerą su dideliu KPS skaičiumi. Analizės rezultatai rodo, kad optimalus KPS skaičius yra bent 100 KPS. Nors egzistuoja galimybė apdoroti vaizdo srautą, kuris yra išsiliejęs [JZS+18], šis būdas reikalauja papildomų skaičiavimo išteklių, kurie galėtų būti panaudojami kitiems skaičiavimams. Nors kelių kamerų sistema [LH19] išsprendžia problemą, kai žaidėjai uždengia kamuoliuką ar kitus kadre esančius objektus, bet 95.9% [Old15] žaidimo įvykių galima atpažinti ir su viena kamera. Nors kameros

rezoliucijai nėra skirta daug dėmesio apžvelgtuose darbuose, remiantis tyrimu [Rad18], galima teigti, kad ji turėtų būti bent 1080p raiškos, norint, kad neatsirastų papildomų artefaktų nuotraukoje. Remiantis techninės įrangos literatūros analize, galima teigti, kad „Apple iPhone 12 Pro“ [App22c] atitinka keliamus reikalavimus ir gali būti naudojamas siekiant atpažinti stalo teniso žaidimą. Remiantis tuo, kad 95.9% žaidimo įvykių galima atpažinti su viena, paprastam vartotojui skirta, kamera [Old15], šiame magistriniame darbe tikimasi įrodyti, kad mobilusis telefonas „Apple iPhone 12 Pro“ yra tinkamas stalo teniso žaidimo sekimui bei įvykių atpažinimui.

1.2. Kadro paruošimas

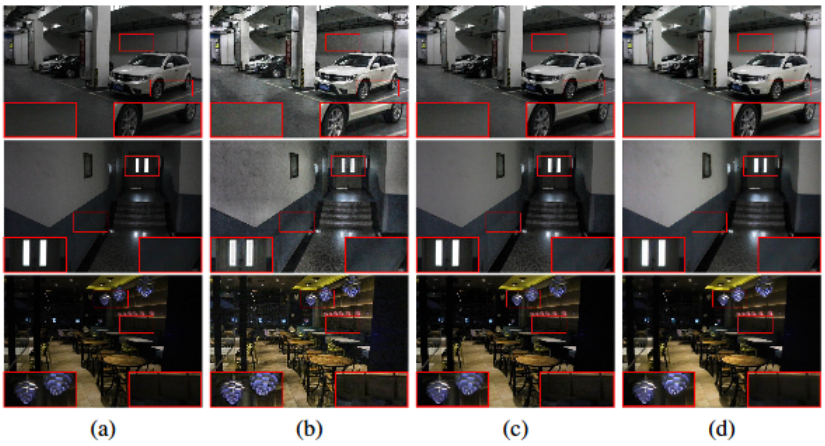
Užfiksavus kadrą, reikia jį apdoroti siekiant gauti kuo tikslesnius atpažinimo rezultatus. Šiame skyriuje apžvelgiami literatūros šaltiniai aprašantys skirtingus būdus skirtus kadro paruošimui. Kadangi yra tikėtina, jog stalo tenisas bus žaidžiamas prie blogų apšvietimo sąlygų, yra tiriamos galimybės panaikinti prie per mažo apšvietimo atsirandančius artefaktus nuotraukoje bei ją paryškinti [AB14; MOW+07; MRT13; SJ17; ZSL+12]. Tai svarbu filmuojant su „iPhone 12 Pro“, nes dėl mažo patenkančio šviesos skaičiaus filmuojant 120 arba 240 KPS greičiu, nuotraukos būna tamsesnės. Šiame skyriuje taip pat apžvelgiami metodai, skirti sumažinti lęšių sudaromą kadro iškraipymą [KKL+20]. Tyrime atsižvelgta ir į algoritmų našumą, nes filmuojant 240 KPS greičiu, vienam kadrui apdoroti tenka tik 4.1 ms (1000 ms. / 240 kadrų).

1.2.1. Mažo apšvietimo kadrų pagerinimas



3 pav. (a) Nuotrauka, esant normalioms apšvietimo sąlygoms. (b) Įvesties nuotrauka. (c) Gautas rezultatas [ZSL+12]

Vienas iš būdų pagerinti nuotraukas, darytas esant nepakankamo apšvietimo sąlygoms - taikyti pagerintą miglos pašalinimo (angl. *haze removal*) algoritmą [ZSL+12]. 3 pav. pateikiamas šio algoritmo pritaikymo rezultatas. Šis algoritmas veikia su invertuotu kadru. Siekiant sumažinti triukšmo kiekį kadre bei išlaikyti objektų kraštines, kartu su minėtuoju algoritmu turi būti naudojamas jungtinis dvišalis (angl. *joint-bilateral*) filtras. Atlikti eksperimentai parodė, kad šis algoritmas veikia geriau, negu to meto rinkoje plačiai naudoti algoritmai [DWP+11; RJW96].



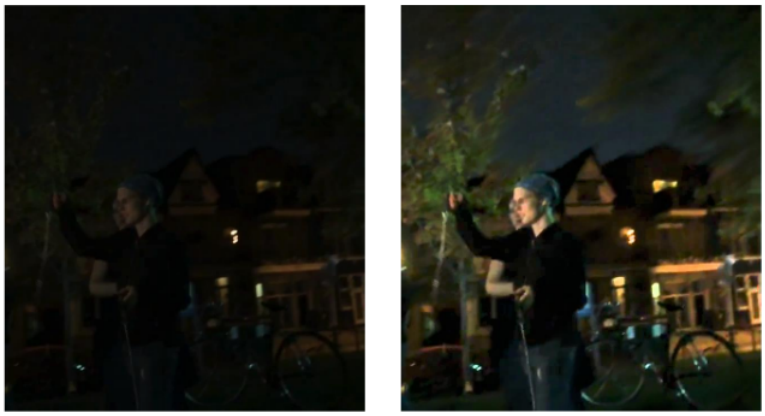
4 pav. (a) Originalios nuotraukos. (b) ACEWC algoritmas. (c) CADIE algoritmas. (d) [SJ17] algoritmas.

Kitas būdas pagerinti nuotraukas - dviejų žingsnių triukšmo šalinimo algoritmas [SJ17], galintis pagerinti kadrų, užfiksuotų esant blogoms apšvietimo sąlygoms, kontrastą bei sumažinti artefaktų skaičių juose. Šis algoritmas:

1. sukuria artefaktų histogramą kadre pagal triukšmo lygio funkciją ir, naudodamasis ja, padidina nuotraukos kontrastą;
2. pritaiko JND (tik pastebimo skirtumo) modelį, pagal kurį pašalina triukšmą iš kadro.

4 pav. pateikiami šio algoritmo taikymo rezultatai bei palyginimas su kitais populiariais algoritmais. Tačiau nėra žinomi šio ir pirmo algoritmo našumo įvertinimai, dėl to yra tikėtina, kad taikant šiuos algoritmus „iPhone 12 Pro“ telefone, nebūtų galima užtikrinti, kad liks pakankamai laiko atpažinti kadre esančius objektus.

Siekiant užtikrinti kadro apdorojimo našumą „Apple iPhone 12 Pro“ telefone, reikia rinktis algoritmą, gebantį išnaudoti ne tik procesorių, bet ir vaizdo plokštę. Vienas iš tokių - paskirstyto skaičiavimo [MOW+07] algoritmas, tinkamas naudoti norint realiu laiku sumažinti artefaktų, atsirandančių esant mažo apšvietimo sąlygoms, skaičių. Šis algoritmas leidžia paskirstyti skaičiavimus per visus vaizdo plokštės branduolius, taip pagerinant algoritmo našumą. Algoritmas taip pat geba pagerinti kokybę kadruose, kuriose fiksuojamas judėjimas, taip išvengiant išsiliejimo efekto. Algoritmas remiasi bei pagerina struktūrinį adaptyvųjį anizotropinio vaizdo filtravimo algoritmą. Kadangi šis algoritmas yra abstraktus ir konfigūruojamas jis yra tinkamas naudoti daugelyje skirtingų scenarijų, tačiau, kadangi nėra pateikiamas pavyzdinis šio algoritmo įgyvendinimas, jo įgyvendinimas „Apple iPhone 12 Pro“ nėra trivialus.



(a) (b)

5 pav. (a) Originali nuotrauka. (b) [MRT13] algoritmas.

Siekiant palengvinti kadro pagerinimo algoritmo įgyvendinimą „iOS“ operacinėje sistemoje, galima remtis [MRT13] algoritmu, kuris yra pateikiamas kartu su pavyzdiniu įgyvendinimu. Šis algoritmas veikia su mobiliuoju telefonu „Apple iPhone 5“ ir geba pagerinti kokybę naudojantis vien tik telefono vaizdo plokšte. Naudojamas histogramos išlyginimas bei kadro triukšmo šalinimas naudojantis tikimybiniu laiko vidurkiu, gautu iš paslėptojo Markovo modelio. Šio algoritmo rezultatai matomi 5 pav. Naudojantis vien „Apple iPhone 5“ vaizdo plokšte, kadras apdorojamas per 1.30 ms. Šis kadro paruošimo algoritmas taip pat geba apdoroti ir kadre esantį judėjimą, nes jis buvo kuriamas su tikslu pagerinti vaizdo medžiagą, užfiksuotą koncertuose. Iš rezultatų galima daryti išvadą, kad šis sprendimas yra taip pat tinkamas ir stalo teniso žaidimo filmavimui, nes jis palieka sąlyginai daug laiko (2.8ms) atlikti kadre esančių objektų identifikavimą net ir su „iPhone 5“ mobiliuoju telefonu.

Pasirinkus artefaktų skaičiaus mažinimu nuotraukose algoritmą, jo našumą galima įvertinti su [AB14] duomenų rinkiniu. Šis duomenų rinkinys galėtų būti naudojamas siekiant užtikrinti, kad pasirinktas stalo teniso žaidimo kadro paruošimo metodas iš tiesų gali kokybiškai pašalinti triukšmą iš kadro bei jį paruošti tolesniam apdorojimui.

1.2.2. Lęšių iškraipymo mažinimas

Norint, kad stalo teniso stalas bei tinklelis būtų identifikuojami teisingai, reikia užtikrinti, kad kadras neturėtų optinių lęšių iškraipymų. Tokie iškraipymai gali paveikti kompiuterinės regos algoritmą bei padidinti neatpažintų arba klaidingai identifikuotų atvejų skaičių [SEG10]. Lęšių sudaromi iškraipymai gali paveikti ir atstumų skaičiavimo algoritmą. Lyginant šiuos populiarius metodus, skirtus sumažinti lęšių sudaromą kadro iškraipymą, nustatyta [KKL+20], kad:

- dvimatis Euklidinio atstumo skaičiavimas nėra tinkamas realaus laiko vaizdų analizei;
- apskrito lanko skaičiavimo metodas užtrunka per daug laiko;
- Hough parametrizuotos erdvės metodas labai priklauso nuo parametrų;
- iteratyvus Hough transformacijų optimizavimas nėra patikimas esant blogoms apšvietimo sąlygoms arba išblukusiems vaizdams;
- nuo modelio priklausantis γ-liekamojo ištaisymo koeficiento metodas negali būti plačiai

taikomas;

- metodai, paremti konvoliuciniais neuroniniais tinklais, yra jautrūs aplinkos parametrams arba negali ištiesinti kadrų neprarandant pikselių;
- metodai, paremti generatyviaisiais rungimosi tinklais, negali apdoroti didelių iškraipymų.

Siekiant išvengti išvardintų trūkumų, galima naudoti kadro išskirčių algoritmą [KKL+20], gebantį sukurti stabiliausius kadrus esant įvairiems aplinkos faktoriams. Tačiau, šio algoritmo praktinis panaudojimas yra sudėtingas, nes nėra pateikiama pavyzdinė realizacija bei našumo specifikacijos. Šis metodas galėtų būti pritaikomas realaus laiko kompiuterinės regos uždaviniams, tačiau nėra tai įrodančių eksperimentų. Atsižvelgiant į tai, kad aprašytas metodas dirba su dideliu lęšių iškraipymu, galima daryti išvadą, kad žaidimo rezultatų sekimui su „Apple iPhone 12 Pro" telefonu nereikės taikyti šio algoritmo. Iškraipymams atsiradus, galima taikyti paprastesnį ir rinkoje labiau paplitusį algoritmą.

1.2.3. Kadro paruošimo apibendrinimas

Didėjant KPS skaičiui, mažėja laikas tarp kadrų, dėl to pasirinktas algoritmas turėtų gebėti apdoroti kadrą itin greitai. Ištyrus literatūros šaltinius, galima teigti, kad [MRT13] algoritmas yra geriausias šiam tikslui pasiekti, nes jis sugeba apdoroti kadrą per 1.30 ms net su „iPhone 5". Jeigu atlikus praktinius eksperimentus paaiškėtų, kad dėl „Apple iPhone 12 Pro" lęšių kadro iškraipymo negalima atpažinti objektų kadre, arba negalima tiksliai nustatyti atstumų, reikėtų taikyti vieną iš rinkoje naudojamų ir plačiai paplitusių algoritmų, aptartų [KKL+20].

1.3. Kamuoliuko atpažinimas

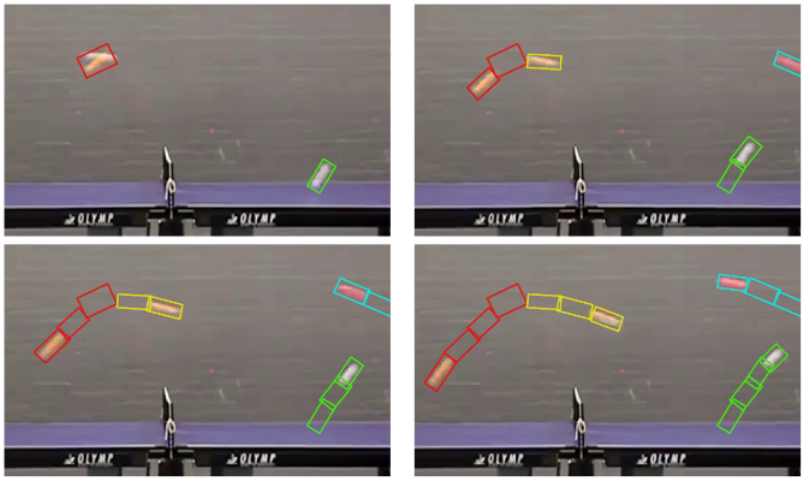
Vienas svarbiausių stalo teniso žaidimo sekimo kompiuterine rega uždavinių - tikslus kamuoliuko nustatymas erdvėje. Šiame skyriuje apžvelgiami literatūros šaltiniai aprašantys būdus identifikuoti kadre esantį kamuoliuką. Svarbu paminėti, kad identifikavimo algoritmas turėtų veikti greitai, nes darant prielaidą, kad kadro apdorojimas užtruks 1.30 ms., lieka 2.8ms atpažinti objektą bei atlikti papildomus skaičiavimus. Apžvelgus šaltinius nustatyti du esminiai būdai atpažinti kamuoliuko padėtį erdvėje: naudojantis neuroniniais tinklais su iš anksto apmokytu duomenų rinkiniu [HLC+19; RMM+18] bei naudojant procedūrinius algoritmus [Ale17; Kam04; Mao06], atsižvelgiant į tai, kad iš anksto žinomas kamuoliuko dydis bei spalva.

1.3.1. Fono šalinimo būdai

Procedūriškai kamuoliuką atpažinti galima taikant patentuotą greitai judančių objektų atpažinimo metodą [Kam04]. Šis metodas susideda iš kelių žingsnių:

1. kadro fono informacijos rinkimo - iš užfiksuotos kadrų sekos sukuriama fono nuotrauka;
2. išskirčių atskyrimo - kiti kadrai sekoje yra lyginami su fono nuotrauka, neatitikimai yra identifikuojami;
3. kamuoliuko identifikavimo - žinant kamuoliuko apimtis, iš išskirčių pasirenkami pikseliai, labiausiai atitinkantys kamuoliuką.

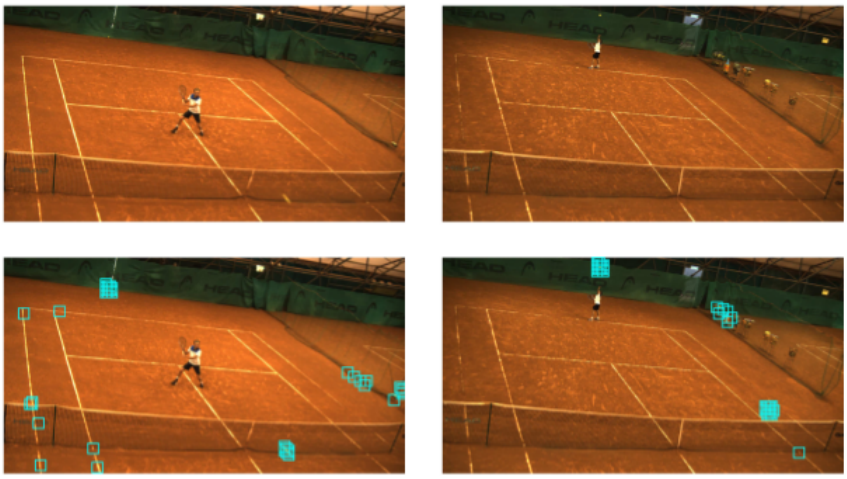
Šio metodo adaptacija taikoma algoritme, skirtame nustatyti lauko teniso kamuoliuko padėtį žaidimo metu [Mao06]. Algoritmas sugeba teisingai nustatyti lauko teniso kamuoliuką 90% atvejų. Sekimo greitis - 20 KPS, tačiau, svarbu paminėti, kad rezultatai yra gauti 2006m. ir, dėl technologijų pažangos, skaičiavimus būtų įmanoma atlikti greičiau. Šis algoritmas veikia greičiau nei neuroninių tinklų klasifikatoriai, tačiau jis labai priklauso nuo fono kintamumo. Naudojantis šiuo algoritmu, kamera turėtų būti nejudinama bei fone negalėtų būti „triukšmo“ - kitų žaidėjų, žiūrovų, trenerių. Dėl šių trūkumų, algoritmo nebūtų galima pritaikyti stalo teniso žaidimui sekti su „Apple iPhone 12 Pro“.



6 pav. Judančių objektų identifikavimas, taikant fono šalinimo algoritmą [Ale17]

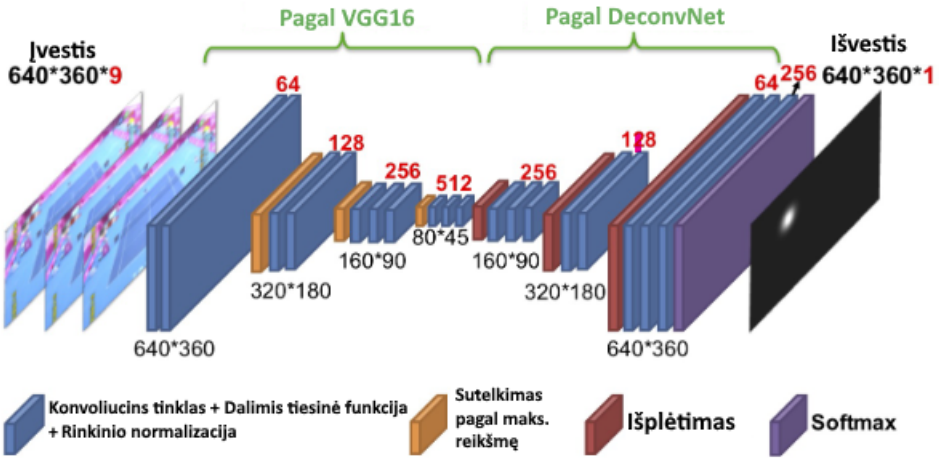
Naujesnė šio metodo adaptacija taip pat tirta ir mobiliajame telefone [Ale17]. Algoritmas buvo bandomas su stalo teniso žaidimo sekimu. Kartu su algoritmu pateikiama ir pavyzdinė „Android“ programa, galinti realiu laiku atpažinti greitai judančius objektus. Svarbu paminėti, kad ši metodo adaptacija turi tokias pačias problemas: kameros padėtis negali kisti, fonas turi būti nekintantis, neturėti „triukšmo“, taip pat, geriausi rezultatai (89 % kadrų kamuoliukas buvo atpažintas) pasiekiami filmuojant stalo teniso žaidimą iš viršaus, ko negalima padaryti nesinaudojant papildoma įranga.

1.3.2. Neuroninių tinklų klasifikatoriais paremti būdai



7 pav. Konvoliucinio neuroninio tinklo klasifikatoriaus panaudojimas [RMM+18]

Siekiant atpažinti stalo teniso kamuoliuką, galima naudoti ir neuroniniu tinklu paremtą klasifikatorių. Egzistuoja neuroninio tinklo klasifikatorius, skirtas atpažinti lauko teniso kamuoliuką [RMM+18]. Šis konvoliucinio neuroninio tinklo klasifikatorius geba atpažinti 98% kamuoliuko padėčių. Siekiant pasinaudoti šiuo tinklu, iš pradžių kadrą reikia sufragmentuoti į mažus, kvadratinius fragmentus. Neuroninis tinklas geba identifikuoti ar tame fragmente yra kamuoliukas, ar jo nėra. Remiantis klasifikatoriaus rezultatais, galima kadrą suskirstyti į vietas, kur buvo atpažintas kamuoliukas ir kur nebuvo. Kaip pastebėjo mokslininkai [RMM+18], daugiausiai teisingų fragmentų randama prie realios kamuoliuko padėties kadre. Remiantis šiuo pastebėjimu, galima tiksliai identifikuoti kamuoliuko vietą kadre. 7 pav. pateikiama šio neuroninio tinklo klasifikavimo vaizdinė reprezentacija.



8 pav. „TrackNet“ struktūra [HLC+19]

Kitas konvoliucinis neuroninis tinklas - „TrackNet“ [HLC+19], gebantis atpažinti 95.3% greitai judančių kamuoliukų sportiniuose žaidimuose. Nors šis modelis nebuvo tiriamas su stalo teniso žaidimu, buvo atlikti eksperimentai su lauko tenisu ir badmintonu. Šis neuroninis tinklas

geba apdoroti ne tik vieną kadrą vienu metu, bet ir kelis, iš eilės einančius, kadrus. Dėl to jis gali dirbti su kadrais, užfiksuotais su mažu KPS skaičiumi. Tačiau nėra žinoma, ar būtų įmanoma šį tinklą pritaikyti „Apple iPhone 12 Pro“ arba kituose mobiliuosiuose įrenginiuose.

Norint pritaikyti neuroninį tinklą apdoroti duomenis realiu laiku mobiliajame įrenginyje, galima remtis futbolo kamuolio atpažinimo algoritmu [TDR19], pritaikytu futbolą žaidžiančiame robote. Svarbiausias uždavinys - sufragmentuoti kadrą taip, kad fragmentas būtų ne didesnis nei kamuolys. Šis algoritmas geba ne tik sufragmentuoti kadrą, bet ir identifikuoti fragmentus, kuriuose, labiausiai tikėtina, bus kamuolys. Svarbu paminėti, kad norint pasinaudoti šiuo algoritmu, reikia iš karto žinoti, kokios spalvos bus kamuolys (tam, kad būtų galima kadrui pritaikyti spalvos kaukę). Algoritmas geba pasirinkti fragmentus per $O(n^2)$, kur n - unikalių vietų kadre, kuriose galėtų būti kamuolys, skaičius. Vidutiniškai, šis metodas gali suskirstyti 640×480 kadrą per 1.46 ms. bei identifikuoti kamuolį (naudojantis neuroniniu tinklu) per 1.8 ms.

1.3.3. Kamuoliuko atpažinimo apibendrinimas

Nors yra būdų atpažinti kamuoliuką nesinaudojant konvoliuciniais neuroniniais tinklais [Ale17; Kam04; Mao06], jie visi turi tuos pačius trūkumus:

- fonas negali kisti;
- kamera turi būti fiksuotoje padėtyje ir negali būti judinama žaidimo metu;
- fone negali būti triukšmo;
- kamuoliuko spalva turi būti žinoma iš anksto.

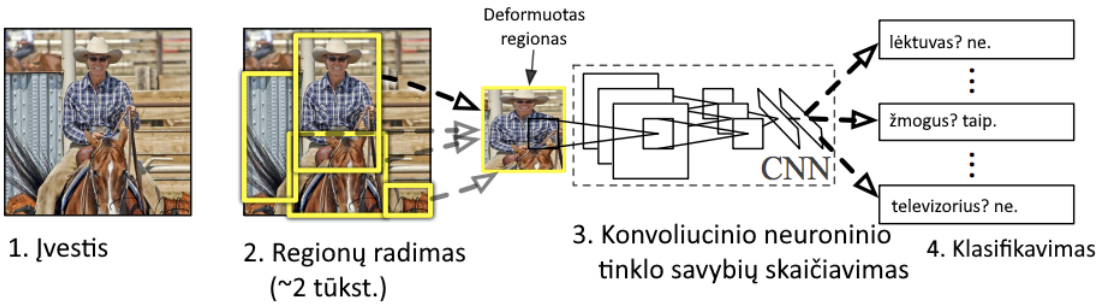
Dėl šių trūkumų, jų panaudojimas nėra praktiškas paprastam vartotojui. Kita vertus, neuroniniais tinklais [HLC+19; RMM+18; TDR19] paremti kamuoliuko atpažinimo modeliai neturi šių trūkumų, tačiau nėra tikėtina, kad jie gebėtų identifikuoti kamuoliuką per 2.8ms „Apple iPhone 12 Pro“ telefone.

Iš atliktos literatūros analizės galima daryti išvadą, kad siekiant realiu laiku atpažinti stalo teniso kamuoliuką „Apple iPhone 12 Pro“ telefone, reikėtų naudoti neuroninį tinklą bei sumažinti KPS skaičių, kad tarp kadrų būtų daugiau laiko jų apdorojimui. Jeigu atlikus eksperimentus paaiškėtų, kad mobiliajame telefone vis tiek nepakanka techninės įrangos resursų galima tirti galimybes pritaikyti kamuoliuko trajektorijos aproksimavimo algoritmą ir neapdoroti visų kadrų.

1.4. Statinių objektų atpažinimas

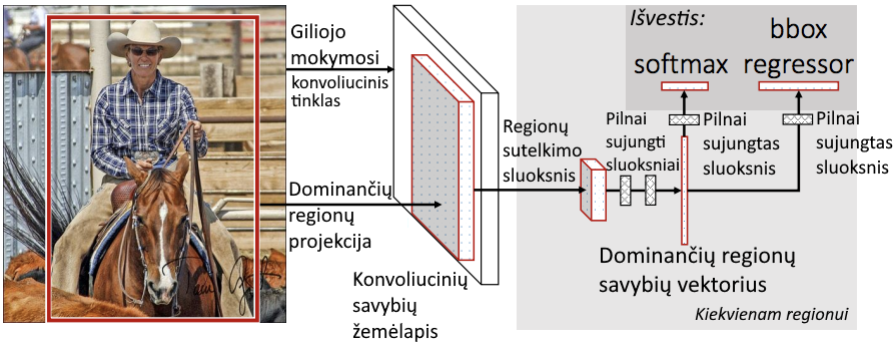
Nors stalo teniso kamuoliuką galima atpažinti nenaudojant neuroninių tinklų, statinių objektų atpažinimas, be vartotojo įvesties bei nesinaudojant neuroninio tinklo klasifikatoriumi, yra sudėtingas ir literatūroje ne dažnai sprendžiamas uždavinys. Dėl to šiame skyriuje, remiantis literatūros šaltiniais apžvelgiami skirtingi neuroninių tinklų taikymo algoritmai siekiant atpažinti statinius objektus kadre. Tikimasi, kad statinių objektų atpažinimo nereikės daryti realiu laiku, t.y. statiniai objektai gali būti atpažįstami kas kelias sekundes arba rečiau. Tiriami dveji neuroninių tinklų tipai: fragmentuojantys kadrus [GDD+14; Gir15; RHG+15] ir dirbantys su visu kadru [RF18].

1.4.1. Kadrus fragmentuojantys neuroniniai tinklai



9 pav. „R-CNN“ architektūra [GDD+14]

Vienas iš būdų objektų identifikavimui - „R-CNN“ sistema [GDD+14]. Ši sistema yra paremta konvoliuciniais neuroniniais tinklais ir turi pavyzdinius modelius, apmokytus naudojantis „PASCAL VOC“ [EEV+15] duomenų rinkiniais. Ši sistema geba identifikuoti 51% objektų, esančių VOC rinkinyje. Tačiau, ji nėra pritaikyta mobiliesiems įrenginiams bei greitam objektų atpažinimui. Taip pat, sistema viduje fragmentuoja nuotraukas į 2000 atskirų regionų ir vykdo didelį kiekį klasifikavimų, dėl ko, naudojantis net stalinio kompiuterio technine įranga, objektų identifikavimas nuotraukoje trunka apie 47 sekundes.



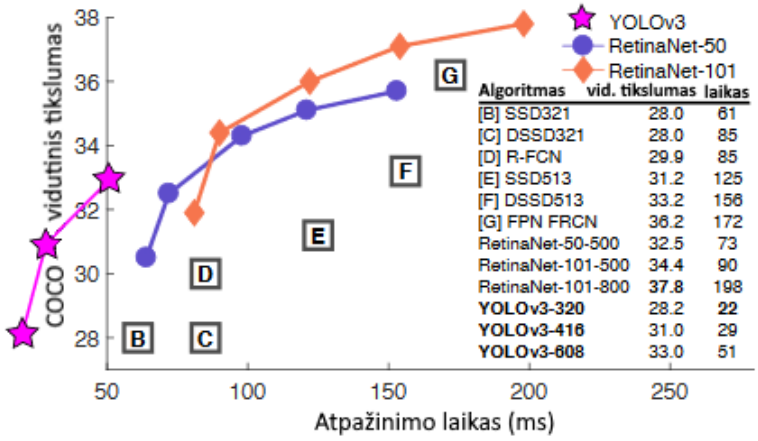
10 pav. „Fast R-CNN“ architektūra [Gir15]

Siekiant išspręsti „R-CNN“ algoritmo trūkumus, jos autoriai pasiūlė pagerintą algoritmą - „Fast R-CNN“ [Gir15], kuris panaikina reikalavimą fragmentuoti nuotraukas į 2000 atskirų regionų. Naudojant konvoliucinį neuroninį tinklą, algoritmas sukuria konvoliucinių savybių žemėlapį visam įvesties kadrui, kuris yra naudojamas tolesniuose žingsniuose. Tačiau, kaip ir „R-CNN“ algoritme, vis tiek naudojamas sąlyginai lėtas atrankinės paieškos algoritmas siekiant sukurti dominančių regionų (angl. *RoI*) žemėlapį. „Fast R-CNN“ gali būti apmokomas 9 kartus greičiau nei „R-CNN“ bei identifikuoti objektus 24 kartus greičiau.

Tiek „R-CNN“, tiek ir „Fast R-CNN“ algoritmai remiasi atrankinės paieškos algoritmu, kuris sumažina algoritmo našumą. Siekiant išspręsti šią problemą, galima naudoti pagerintą „Fast

R-CNN“ algoritmo implementaciją [RHG+15] - „Faster R-CNN“. Šis algoritmas nuo „Fast R-CNN“ skiriasi tuo, kad vietoj atrankinės paieškos algoritmo yra naudojamas kitas neuroninis tinklas, galintis identifikuoti regionus, kuriuose labiausiai tikėtina, kad bus objektai, kuriuos reikės identifikuoti. Šis neuroninis tinklas, įvedus nuotrauką, grąžina rinkinį „rėmelių“ su koordinatėmis, kuriuose, tikėtina, kad bus priekinio plano objektai. Toks „Fast R-CNN“ algoritmo pagerinimas leidžia atlikti objektų identifikavimą per mažiau nei 1s. Taip pat, šis algoritmas geba atpažinti 70% „VOC“ rinkinio objektų. Šis algoritmas sunkiai pritaikomas „Apple iPhone 12 Pro“, nes standartiniuose neuroninių tinklų modeliuose, esančiuose „CoreML“, nėra šiam algoritmui reikalingų sluoksnių, taip pat nėra ir konvertavimo įrankių, galinčių konvertuoti „Faster R-CNN“ tinklo modelį į „CoreML“.

1.4.2. „YOLO v3“ neuroninis tinklas



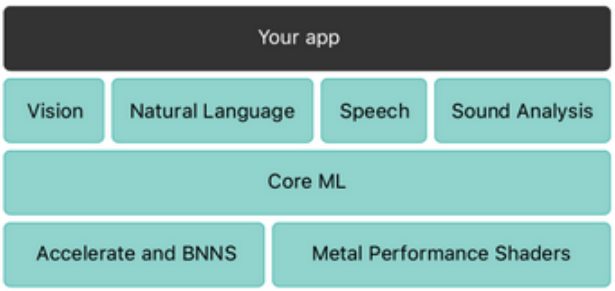
11 pav. „YOLO v3“ našumo palyginimas su kitais algoritmais [RF18]

Visiškai kitokiu principu veikia YOLO v3 algoritmas [RF18]. Vietoj pirminio kadro fragmentavimo ir dominančių regionų išskyrimo, algoritmas dirba su visu kadru. „YOLO v3“ naudoja vieną konvoliucinį neuroninį tinklą, kuris išskiria priekiniame plane esančių objektų „rėmelius“ bei juos klasifikuoja. „YOLO v3“ suskirsto pradinį kadrą į SxS dydžio matricą, tada iš matricos elementų sukuria m atskirų „rėmelių“. Kiekvienam iš „rėmelių“ neuroninis tinklas priskiria tikimybę bei identifikuotą objektą. Sekančiu žingsniu, algoritmas palieka tik tuos rėmelius, kurių tikimybės yra didesnės už iš anksto nustatytą koeficientą. Toks veikimo būdas leidžia apdoroti kadrus realiu laiku. Šis algoritmas yra 1000 kartų greitesnis už „R-CNN“ ir 100 kartų greitesnis už „Fast R-CNN“. Vidutiniškai, „YOLO v3“ gali apdoroti 45 kadrus per sekundę. Nors, kaip ir „Faster R-CNN“, „CoreML“ nepalaiko „YOLO v3“ neuroninių tinklų modelių, galima rasti skirtingų būdų pritaikyti „YOLO v3“ svorius ir modelį „CoreML“ [Ma-18]. Dėl pradinio kadro skirstymo rėmeliais, šis algoritmas nesugeba atpažinti labai mažų objektų kadre, tačiau, žinant kad tiek stalo teniso stalas, tiek tinklelis yra sąlyginai dideli, tai neturėtų sukelti problemų panaudojant šį tinklą.

1.4.3. Statinių objektų atpažinimo apibendrinimas

Tikint, kad stalo teniso žaidimo metu kamera nebus judinama ir stalą bei tinklelį identifikuoti reikės daugiausiai kartą per kelias sekundes, galima rinktis tarp „Faster R-CNN“ bei „YOLO v3“ algoritmų, nes jie veikia greičiausiai savo grupėje. Siūloma rinktis „YOLO v3“, nes yra pateikiamų pavyzdžių jo panaudojimui „iOS“ operacinėje sistemoje bei šis algoritmas yra greičiausias. Svarbu paminėti, kad nėra viešai prieinamų stalo teniso stalų bei tinklelių duomenų rinkinių, dėl to, nepriklausomai nuo pasirinktos objektų identifikavimo sistemos, ją reikėtų apmokyti bei sudaryti stalo teniso stalų bei tinklelių duomenų rinkinį.

1.5. „Vision“ karkaso panaudojimas



12 pav. „iOS“ mašininiu mokymu paremti karkasai [App20a]

Siekiant palengvinti kompiuterinės regos uždavinių įgyvendinimą „iPhone“ telefonuose, kompanija „Apple“ 2017 metais „WWDC“ renginio metu [App17] pristatė „Vision“ karkasą. Šis karkasas sumažina žingsnių, reikalingų norint apdoroti įvestį iš telefono kameros, skaičių bei leidžia naudoti „CoreML“ modelius siekiant identifikuoti objektus kadre. „Vision“ leidžia:

- atlikti pavienių kadrų analizę, siekiant identifikuoti objektus juose;
- įvardinti objektus, atkreipiančius dėmesį kadre (angl. *saliency*);
- sekti identifikuotos objektus pagal kameros įvestį;
- atpažinti objektus, turinčius kvadratinę formą, kadruose;
- identifikuoti (be papildomų modelių): žmogaus veidą, kūną, gyvūnus, brūkšninius kodus;
- atpažinti tekstą.

„Vision“ naudojamas „CoreML“ [App20a] yra taip pat kompanijos „Apple“ sukurtas karkasas, skirtas interpretuoti bei sukurti mašininio mokymo modelius. Pilna kompanijos „Apple“ sukurtų karkasų, palengvinančių mašininio mokymosi modelio pritaikymą mobiliuosiuose telefonuose, diagrama pateikiama 12 pav. Tikimasi, kad naudojantis „Vision“ galima palengvinti pavyzdinės programos įgyvendinimą „Apple iPhone 12 Pro“ telefone.

„Vision“ karkasas gali būti panaudojimas siekiant nustatyti tinklelio bei stalo padėtį kadre. Tačiau, kadangi nėra jau sukurtų modelių su stalo teniso stalu ir tinkleliu, iš pradžių jį reikėtų susikurti ir apmokyti. Tai galima padaryti apmokant neuroninį tinklą naudojantis „Apple“ sukurtu „Create ML“ įrankiu, tačiau, kadangi nuspręsta naudoti „YOLO v3“ algoritmą, galima apmokyti „YOLO v3“ tinklo modelį ir tada jį konvertuoti į „CoreML“ formatą naudojantis egzistuojančiais

įrankiais [Ma-18].

Įsikėlus „CoreML“ modelį į mobiliąją programėlę, galima naudoti „Vision“ karkaso teikiamą programavimo sąsają, kuri leidžia užkrauti šį modelį iš programėlės resursų direktorijos ir automatiškai klasifikuoti „iPhone“ kameros įvestį. „Vision“ karkasas kiekvienam apdorotam kadrui sukuria *VNRecognizedObjectObservation* tipo objektą, kuriame yra pateikiamas šios informacijos apie kadre esančius objektus masyvas:

- *identifier* - atpažinto objekto klasifikatorius (pagal „CoreML“ modelį);
- *confidence* - pasikliovimo procentas, pateikiamas intervale $[0; 1]$;
- *CGRect* tipo objektas, apibūdinantis identifikuoto objekto vietą kadre.

Toks karkaso panaudojimas leidžia lengvai atpažinti statinius objektus kadre, tačiau, „Vision“ palaiko ir judančių objektų atpažinimą. Naudojantis *VNTrackObjectRequest*, galima sukurti užklausas „Vision“ karkasui, kurios susidėtų iš *VNRecognizedObjectObservation* objektų, gautų praeitame žingsnyje. Pateikus sukurtas užklausas „Vision“ karkasui, jis pradėtų sekti atpažintus objektus bei nustatytais laiko intervalais grąžintų *VNDetectedObjectObservation* tipo objektus, kurie nusako atpažinto objekto pokytį kadre. Toks karkaso panaudojimo būdas leistų stebėti ir stalo teniso kamuoliuko judėjimą.

Tikėtina, kad „Vision“ gali būti per lėtas siekiant atpažinti visus įvykius stalo teniso žaidime. Pagal atliktą analizę, siūloma atlikti eksperimentus su „Vision“ karkasu ir nustatyti, ar yra įmanoma įgyvendinti pilną stalo teniso žaidimo sekimą naudojantis vien tik „Vision“ karkasu.

1.6. Literatūros analizės išvados

Literatūros apžvalgoje buvo apžvelgti šaltiniai, tiriantys techninei įrangai keliamus reikalavimus, kadro paruošimo algoritmus, skirtingus būdus atlikti kamuoliuko atpažinimą, statinių objektų atpažinimą. Taip pat apžvelgtas „Vision“ karkaso panaudojimas „Apple iPhone 12 Pro“ telefone. Nustatyta, kad:

- techninei įrangai keliamus reikalavimus atitinka „Apple iPhone 12 Pro“;
- kadrus, užfiksuotus prie mažo apšvietimo sąlygų, galima apdoroti remiantis [MRT13] algoritmu, nes jis yra greičiausias iš tirtų bei turi pavyzdinį įgyvendinimą, skirtą „iOS“ sistemai;
- konvoliucinis neuroninis tinklas turėtų būti naudojamas „Apple iPhone 12 Pro“ telefone siekiant atpažinti kamuoliuką, nes, kitaip nei procedūriniai algoritmai, jis nėra veikiamas besikeičiančio fono;
- „YOLO v3“ [RF18] algoritmas galėtų būti naudojamas siekiant atpažinti stalo teniso stalą bei tinklelį „Apple iPhone 12 Pro“ telefone, nes jis veikia iki 1000 kartų greičiau nei tirti kadrą fragmentuojantys neuroniniai tinklai;

2. Kompiuterinės regos sistemos projektavimas

Šiame skyriuje aprašomi principiniai sprendimai, reikalingi norint įgyvendinti kompiuterinės regos sistemą, skirtą sekti stalo teniso žaidimą naudojantis mobiliuoju telefonu. Remiantis literatūros šaltinių analize, tolesnis tyrimas yra suskirstytas į šias dalis:

- „YOLO v2“ [RF16] konvoliucinio neuroninio tinklo modelio sudarymas, jo našumo bei tikslumo vertinimas;
- kamuoliuko trajektorijos skaičiavimo algoritmas;
- kamuoliuko atsimušimo nustatymas;
- stalo teniso žaidimo projektavimas remiantis baigtinių automatų teorija.

Tyrimo metu buvo nuspręsta duomenis rinkti su „Apple iPhone 12 Pro“, dėl to kameros lęšių iškraipymo bei įtakos žaidimo sekimui nustatymas nebuvo atliekamas. Kadangi tyrimo duomenys surinkti su įrenginiu, kuriame įgyvendinimas algoritmas, tikimasi, kad tai negalės padaryti įtakos algoritmo kokybei.

2.1. Kamuoliuko atpažinimas neuroninio tinklo pagalba

Siekiant identifikuoti kamuoliuko poziciją kadre, tyrimo metu, naudojantis „Apple“ „CreateML“ įrankiu, buvo sukurtas bei apmokytas „YOLO v2“ architektūra paremtas neuroninis tinklas. Nors, atsižvelgiant į literatūros analizę, „YOLO v3“ architektūra paremtas tinklas galėtų sparčiau apdoroti kadrus, dėl sudėtingų įgyvendinimo detalių bei atsižvelgiant į tai, kad „Apple“ optimizuoja „Vision“ karkasą būtent „v2“ tipo neuroniniams tinklams, daroma prielaida, kad „YOLO v2“ neuroninio tinklo našumas neturės pastebimos įtakos algoritmo kokybei.

2.1.1. Pirminio duomenų rinkinio sudarymas

Pirminis 313 680 kadrų duomenų rinkinys buvo sudarytas tyrimo autoriaus. Kadrai užfiksuoti stalo teniso klube. Duomenų rinkinį bendrai sudaro 21 minutė ir 47 sekundės 1920 px X 1080 px raiškos, 240 kps (kadrų per sekundę) vaizdo įrašų. Duomenų rinkinys buvo surinktas filmuojant „Apple iPhone 12 Pro“ mobiliuoju telefonu. Duomenų rinkinyje išskiriami šie kadrų požymiai:

- abu žaidėjai kartu su kamuoliuku bei stalo teniso stalu yra pilnai matomi kadre;
- fone nėra kitų žaidėjų, pašalinių objektų;
- fone yra 10 kitų žaidėjų, žaidžiančių prie 5 stalų;
- kamera nėra statmena stalo teniso stalui, pavyzdžiui, filmuojant iš šono arba už žaidėjo nugaros.

Duomenų rinkinys yra nuoseklus, t.y., vaizdo įrašymas nebuvo stabdomas kamuoliukui nukritus arba setui pasibaigus. Didelė raiška bei didelis kadrų per sekundę skaičius leido tiksliai užfiksuoti stalo teniso kamuoliuką, išvengiant vaizdo išsiliejimo bei kitų artefaktų.

2.1.2. Duomenų rinkinio ženklinimas

Duomenų rinkinio filtravimas buvo atliktas rankomis išrenkant 43 953 unikalius kadrus, kuriuose varijuoja skirtingi požymiai. Siekiant šį rinkinį paruošti neuroninio tinklo mokymui „Create ML“ aplinkoje, kiekvienas iš kadrų turėjo būti anotuotas, nurodant stalo teniso kamuoliuko koordinates.

Kadrų anotavimas atliktas naudojantis atvirojo kodo „Computer Vision Annotation Tool“ (CVAT) [Too21] sistema. Ši sistema buvo pasirinkta, nes:

- geba interpoluoti objekto pozicijos pasikeitimus tarp dviejų raktinių kadrų;
- turi grafinę vartotojo sąsają bei leidžia anotacijas atlikti naršyklėje;
- ją galima paleisti nuosavame serveryje bei naudotis nemokamai;
- geba dirbti su didelio kps vaizdo įrašais.

„CVAT“ sistema, iš 43 953 kadrų, interpolavo 28 953, likę 15 000 buvo anotuoti rankomis. Rankinis anotavimo procesas užtruko 64 valandas.

2.1.3. Neuroninio tinklo mokymas

Konvoliuciniam neuroniniam tinklui kurti buvo pasirinktas „Create ML“ [App21a] įrankis. Šis įrankis buvo pasirinktas dėl to, nes:

- turi integruotą palaikymą „YOLO v2“ architektūra paremtų objektų klasifikatorių kūrimui;
- geba išsaugoti neuroninio tinklo koeficientus į formatus, tinkamus naudoti su „Vision“ karkasu;
- geba išnaudoti „Metal“ [App21b] posistemę, kuri pagreitina neuroninio tinklo mokymosi procesą naudodama vaizdo plokštę;
- turi grafinę vartotojo sąsają leidžiančią vizualiai stebėti mokymo rezultatus.

Nors „Create ML“ įrankis palengvina neuroninio tinklo apmokymo procesą, jis reikalauja specifinio duomenų rinkinio formato, kurio „CVAT“ sistema nepalaiko. Duomenų anotacijų konvertavimui buvo panaudotas autoriaus sukurtas „Python“ scenarijus, suformavęs „JSON“ objektą su kamuoliuko anotacijomis.

Konvoliucinis neuroninis tinklas buvo mokomas 55 000 iteracijų. Gautojo objektų klasifikatoriaus klaidos koeficientas – 0,216. Mokymo metu „Create ML“ sistema suskirstė duomenų rinkinį į dvi dalis - 90% duomenų buvo naudojami neuroniniam tinklui mokyti, likę 10% - validacijai. Šis suskirstymas buvo atliekamas automatiškai „Create ML“ sistemos, remiantis „Apple“ nepaviešintais algoritmais, parenkančiais optimaliausius mokymosi ir validavimo duomenų rinkinius.

2.1.4. Neuroninio tinklo architektūra

Baigus mokymo procesą, „Create ML“ įrankis sukūrė „YOLO v2“ architektūra paremtą objektų klasifikatorių. Klasifikatoriaus sluoksniai yra aprašomi žemiau pateiktoje lentelėje nr. 1.

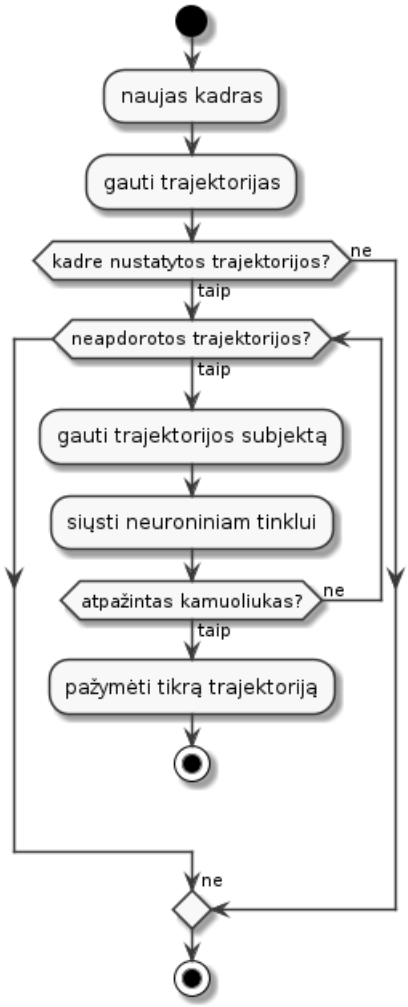
1 lentelė. Neuroninio tinklo sluoksniai

Sluoksnis	**Sluoksnio Apibūdinimas**	**Kiekis**
Convolution	Pagrindinis konvoliucinis sluoksnis, saugantis svorius	9
ActivationLeakyReLU	*Leaky Rectified Linear Unit* aktyvacijos sluoksnis	8
BatchNorm	*Batch normalization* įvesčių normalizavimo sluoksnis	8
PoolingMax	Įvesties sumažinimo sluoksnis	6
Reshape	Įvesties dimensijų keitimo sluoksnis	5
Slice	Įvesties skaidymo sluoksnis	4
Permute	Koordinačių ašių keitimo sluoksnis	3
Scale	Duomenų normalizavimo sluoksnis	2
Multiply	Įvesties daugybos sluoksnis	2
LoadConstant	Konstantų įkėlimo sluoksnis	2
ActivationSigmoid	Sigmoidinės aktyvacijos sluoksnis	2
Concat	Duomenų sujungimo sluoksnis	1
UnaryExp	Vienanarės (angl. *unary*) funkcijos sluoksnis	1
Add	Duomenų sudėties sluoksnis	1
Softmax	Minkštojo maksimumo (angl. *softmax*) sluoksnis	1

2.1.5. Kamuoliuko trajektorijos skaičiavimo algoritmas

Neuroninio tinklo įgyvendinimas įgalino trajektorijos skaičiavimo algoritmą. Tyrimo metu paaiškėjo, kad pirminė prielaida naudoti „Vision“ karkaso papildinį nepasitvirtino, nes jis negebėjo išskirti stalo teniso kamuoliuko. Dėl šios priežasties buvo pasirinkta įgyvendinti paprastąjį atpažinimo algoritmą bei jį naudoti tolesniame tyrime.

2.2. Hibridinis trajektorijos algoritmas



13 pav. Hibridinis atpažinimo algoritmas

2020m. kompanija „Apple“ pristatė savo mobiliesiems įrenginiams skirtą „Vision“ karkaso papildinį [App20b], gebantį identifikuoti vaizdo sraute judančių objektų trajektorijas. Juo naudojantis galima nustatyti objektų, judančių pagal kvadratinę funkciją, centroidų koordinates bei funkcijos koeficientus. Nors šis papildinys palengvina kamuoliuoko judėjimo atpažinimo uždavinį, jis turi ir trūkumų: kiekvienas kamuoliuko atsimušimas sugeneruoja naują trajektoriją, taip pat trajektorijos generuojamos ir kamuoliuko atspindžiams bei kitiems kadre judantiems objektams.

Dėl šių priežasčių kamuoliuko identifikavimui buvo pasirinkta įgyvendinti hibridinį algoritmą (13 pav.): kamuoliuko trajektoriją skaičiuoti remiantis „Vision“ karkaso papildiniu, o rezultatus validuoti naudojantis „YOLO v2“ architektūra paremtu autoriaus apmokytu konvoliuciniu neuroniniu tinklu. Tikėtasi, kad šis metodas sumažins tikimybę, kad atpažintos trajektorijos nepriklauso stalo teniso kamuoliukui, o, pavyzdžiui atspindžiams, ar kitiems greitai judantiems objektams kadre.

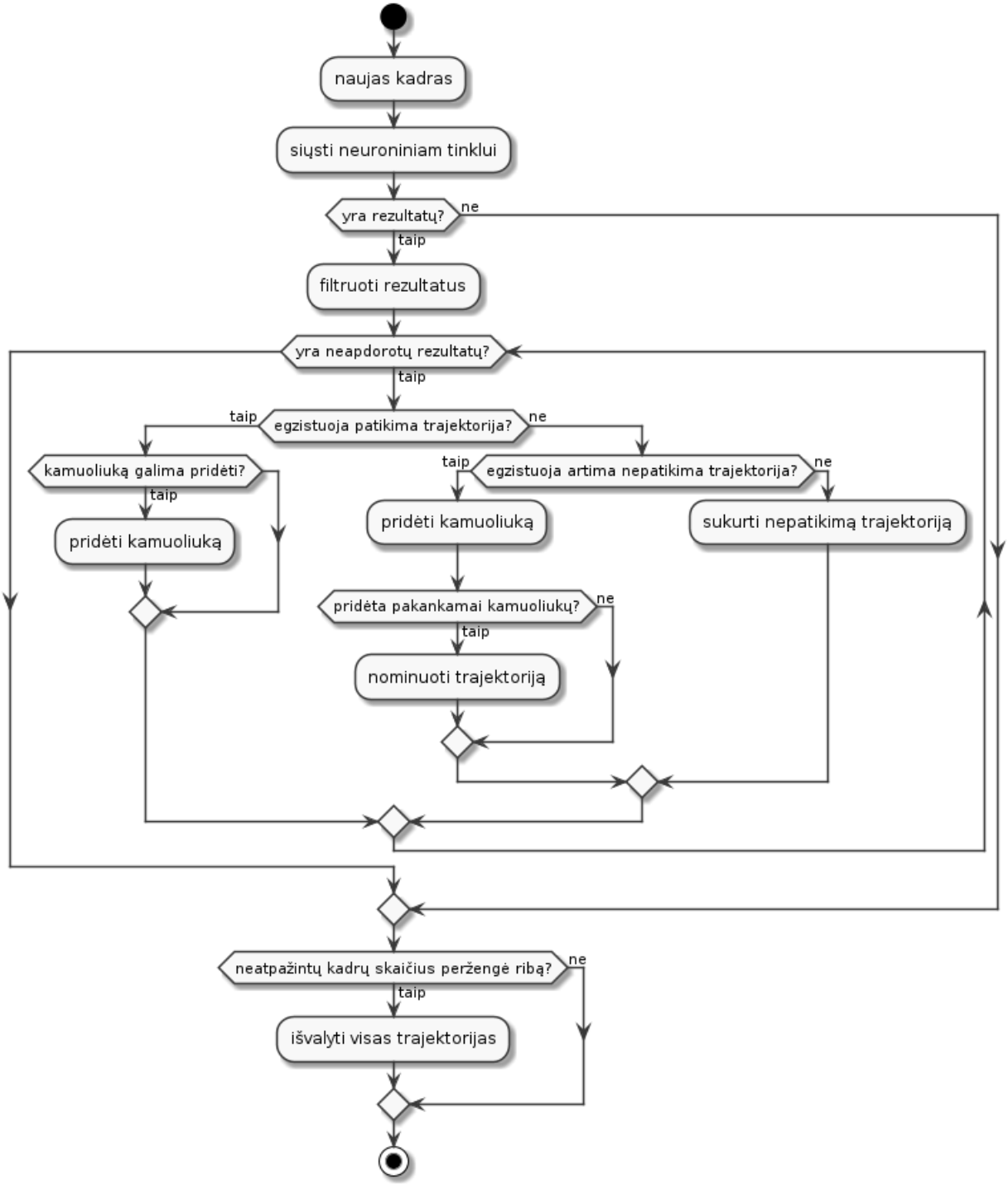
2.2.1. Hibridinio algoritmo tikslumo tyrimas

Įgyvendinus hibridinį algoritmą „iOS“ skirtoje mobiliojoje programėlėje, nustatyta, kad „Vision“ karkaso papildinys, skirtas atpažinti trajektorijoms [App20b], nėra tinkamas atpažinti greitai judančius mažus objektus, pavyzdžiui, stalo teniso kamuoliukus. Testuojant algoritmą su 2 minučių trukmės stalo teniso klube autoriaus nufilmuotu 240 kps vaizdo įrašu, esant idealiomis sąlygomis K_1, „Vision“ nesugebėjo identifikuoti nei vienos trajektorijos, susijusios su stalo teniso kamuoliuku, tačiau sugeneravo trajektorijas šiems kadre esantiems objektams: laikrodžiams, pirštams, raketėms.

Bandymai buvo atlikti su skirtingomis *trajectoryLength* reikšmėmis, atitinkamai, penkių, dešimties ir penkiolikos taškų ilgio minimaliomis trajektorijomis bei skirtingomis *frameAnalysisSpacing* reikšmėmis, atitinkamai, *.zero* (identifikacija be trūkių), 10 ms, 25 ms, 100 ms ir 250 ms trūkiais tarp kadrų. Šie parametrai neturėjo įtakos kamuoliuko atpažinimui.

Siekiant vaizdo įraše nustatyti bent vieną stalo teniso kamuoliuko trajektoriją, tolesni eksperimentai buvo atlikti nustatant *minimumObjectSize* bei *maximumObjectSize* parametrus į normalizuotą (intervale $[0,1]$) kamuoliuko skersmenį. Normalizacija buvo atlikta rankiniu būdu apskaičiuojant kamuoliuko skersmens ilgį pikseliais bei padalinant ją iš kadro pločio pikseliais. *minimumObjectSize* buvo nustatytas iš gautojo rezultato atėmus 5 %, *maximumObjectSize* buvo nustatytas prie gautojo rezultato pridėjus 5 %. Tačiau, dėl mažo kamuoliuko skersmens, šie parametrai nepadėjo pašalinti neaktualių trajektorijų bei nepadėjo „Vision“ karkasui identifikuoti stalo teniso kamuoliuko trajektorijos.

2.3. Paprastasis trajektorijos algoritmas



14 pav. Paprastasis atpažinimo algoritmas

Kadangi „Vision“ karkaso papildinys nėra tinkamas stalo teniso kamuoliuko trajektorijai nustatyti, tolesniame tyrime naudojamas autoriaus sukurtas paprastasis atpažinimo algoritmas (14 pav.). Kiekvienas kadras yra perduodamas neuroniniam tinklui, jeigu tinklas kadre atpažįsta kamuoliukus, jie yra išfiltruojami, ir, atitinkamai pridedami prie patikimos trajektorijos. Jeigu tokia neegzistuoja, atpažintame taške yra sukuriama nauja nepatikima trajektorija. Jeigu kadre neatpažįstamas nei vienas kamuoliukas, arba jeigu išfiltruojami visi kamuoliukai, padidinamas

neatpažintų kadrų skaičius. Kai neatpažintų kadrų skaičius peržengia ribą n, visos trajektorijos yra išvalomos. Trajektorijos nominuojamos surinkus pakankamą skaičių rezultatų, kuomet paliekama viena, patikima trajektorija. Pridėjimo metu, siekiant sumažinti stebėjimo triukšmą yra pritaikomas Kalmano filtras. Naudojantis mažiausių kvadratų metodu, suskaičiuojami patikimos trajektorijos kvadratinės funkcijos koeficientai.

Šis sprendimas taip pat yra lengviau įgyvendinimas, nes, kitaip nei hibridinio algoritmo atveju, visi kamuoliuko atsimušimai yra įtraukiami į bendrą trajektoriją, kuri išlaikoma iki pat taško pabaigos. Naudojantis „Vision“ papildiniu, reikėtų sujungti skirtingas trajektorijas, generuojamas kiekvieno kamuoliuko krypties pasikeitimo atveju.

2.3.1. Patikimos ir nepatikimos trajektorijos

Darant prielaidą, kad dėl neuroninio tinklo klaidos koeficiento ne visi rezultatai gali būti teisingi, algoritmas gautas trajektorijas skirsto į dvi grupes: patikimas ir nepatikimas. Taško pradžioje visi rezultatai yra pridedami prie nepatikimų trajektorijų. Kai bent viena trajektorija surenka pakankamą kiekį rezultatų (tampa įmanoma nustatyti kvadratinės lygties koeficientus), ji yra nominuojama į patikimą, o likusios trajektorijos - išvalomos.

Tuomet visi kiti stebėjimai yra pridedami prie patikimos trajektorijos tik jeigu jie yra nutolę ne daugiau nei per s pikselių nuo trajektorijos pabaigos taško. Kintamasis s taip pat yra naudojamas stebėjimų pradžioje, siekiant pasirinkti, ar kurti naują nepatikimą trajektoriją, ar rezultatą pridėti prie jau egzistuojančios.

2.3.2. Rezultatų filtravimo kriterijai

Nepriklausomai nuo trajektorijos būkles, neuroninio tinklo rezultatai iš pradžių yra išfiltruojami pagal šiuos kriterijus:

- kamuoliukas negali būti žemiau nei c pikselių nuo stalo plokštumos;
- jeigu šiuo metu yra sekamas padavimas, kamuoliukas turi būti užfiksuotas prie paduodančiojo žaidėjo;
- jeigu tikimasi sulaukti atsimušimo kairėje stalo pusėje, kamuoliukas negali būti užfiksuotas už kairiojo stalo krašto ribų;
- jeigu tikimasi sulaukti atsimušimo dešinėje stalo pusėje, kamuoliukas negali būti užfiksuotas už dešiniojo stalo krašto ribų.

2.3.3. Kalmano filtras

Gautas neuroninio tinklo rezultatas taip pat yra filtruojamas naudojantis dvejais nepriklausomais Kalmano filtrais [Kal60], atitinkamai x ir y koordinatėms filtruoti. Atlikus praktinius eksperimentus, nustatyta, kad geriausi rezultatai gauti, kai:

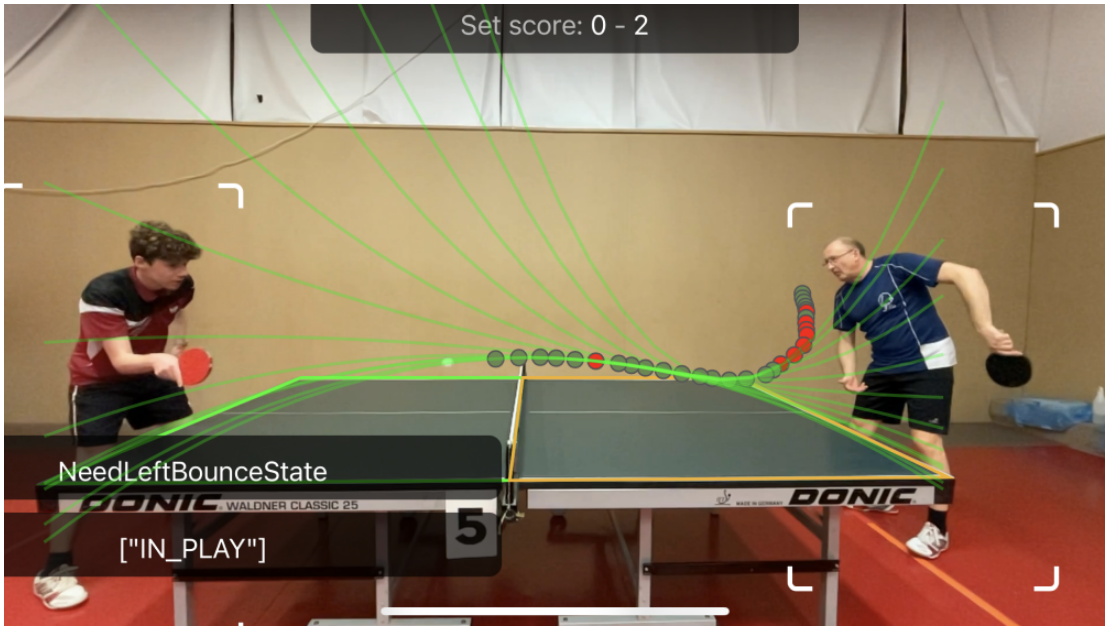
- x koordinatės proceso triukšmo kovariacija yra 0.15
- x koordinatės stebėjimo triukšmo kovariacija yra 0.5
- y koordinatės proceso triukšmo kovariacija yra 0.25
- y koordinatės stebėjimo triukšmo kovariacija yra 1.35

Kalmano filtro panaudojimas taip pat pagerino parabolės koeficientų radimo algoritmą, sumažindamas parabolės krypties koeficiento pasikeitimus, taip pagerindamas atsimušimo nustatymo kokybę.

2.3.4. Parabolės koeficientų skaičiavimas

Siekiant nominuoti trajektoriją kaip patikimą bei sekti kamuoliuko atsimušimus, po kiekvieno rezultato pridėjimo prie egzistuojančios trajektorijos, algoritmas bando suskaičiuoti kvadratinės lygties, atitinkančios kamuoliuko judėjimą, koeficientus. Tai yra atliekama, kai į trajektoriją patenka daugiau nei 4 kamuoliukai. Parabolės koeficientų skaičiavimui atlikti, remiantis „Apple“ straipsniu [App22b], pasitelkiamas mažiausių kvadratų metodas.

2.3.5. Paprastojo algoritmo tikslumo tyrimas

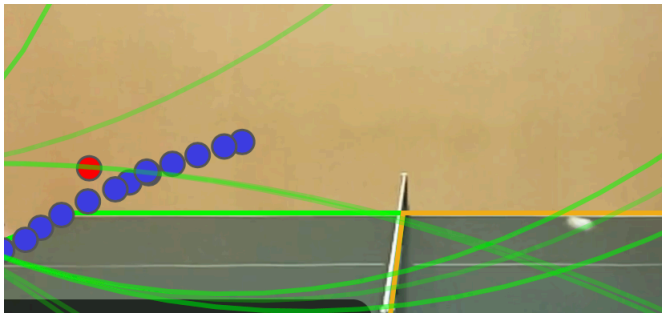


15 pav. Trajektorijos atpažinimas

Paprastasis atpažinimo algoritmas buvo įgyvendintas „iOS“ skirtoje mobiliojoje programėlėje. Testuojant algoritmą su 2 minučių trukmės stalo teniso klube autoriaus nufilmuotu 240 kps vaizdo įrašu, esant idealiomis sąlygomis K_{i}, buvo nustatyta, kad:

- neuroninio tinklo apdorojimas užtrunka 14 ± 1 ms (70 kps), dėl to trajektorijos atvaizdavimas nuo ekrane matomo vaizdo atsilieka 12 ms;
- kai kuriuose kadruose (< 500), kamuoliukas buvo atpažintas ne toje vietoje;
- kai kuriuose kadruose (< 200), buvo atpažintas daugiau nei vienas kamuoliukas;
- kai kuriuose kadruose (< 100), kamuoliukas nebuvo atpažintas.

(a) Nominuojama trajektorija, pradėta nuo klaidingai atpažinto kamuoliuko



(b) Aptiktas kamuoliukas yra toliau nei s

16 pav. Atpažinimo klaidos

Pastebėta, kad algoritmas ne visada teisingai atpažįsta trajektoriją. Dažniausios tyrimo medžiagoje pasitaikiusios klaidos:

- trajektorija uždaroma per anksti, nes neuroninis tinklas nesugeba aptikti kamuoliuko per n kadrų;
- trajektorija uždaroma per anksti, nes aptiktos kamuoliuko padėtys yra toliau nei s n kadrų iš eilės;
- nominuojama trajektorija, pradėta nuo klaidingai atpažinto kamuoliuko.

Praktinių eksperimentų metu nustatytos (ir tolesniame tyrime naudojamos) šios kintamųjų vertės: n - 30 kadrų, c - 25 pikseliai bei s - 100 pikselių. Tikėtina, kad algoritmą įgyvendinant kitame mobiliajame telefone, parametrus reikėtų pakeisti priklausomai nuo telefono ekrano raiškos bei filmavimo greičio.

2.4. Kamuoliuko atsimušimo nustatymas

Siekiant nustatyti kamuoliuko atsimušimo vietą, remiamasi kvadratinės lygties koeficientais, gautais pritaikius paprastąjį trajektorijos atpažinimo algoritmą. Daroma prielaida, kad kamuoliukas atsimušė, jeigu pasikeičia kvadratinės funkcijos $ax^2 + bx + c$ koeficiento a ženklas. Eksperimentais nustatyta, kad vidutiniškai vieno taško metu, esant idealiomis sąlygomis K_i, yra sugeneruojama iki 5 klaidingai teigiamų atšokimo įvykių. Kaip įrodyta pavyzdinės programėlės įgyvendinimo metu, juos galima išfiltruoti, atsižvelgiant į žaidimo kontekstą, bei prieš tai įvykusius taško įvykius.

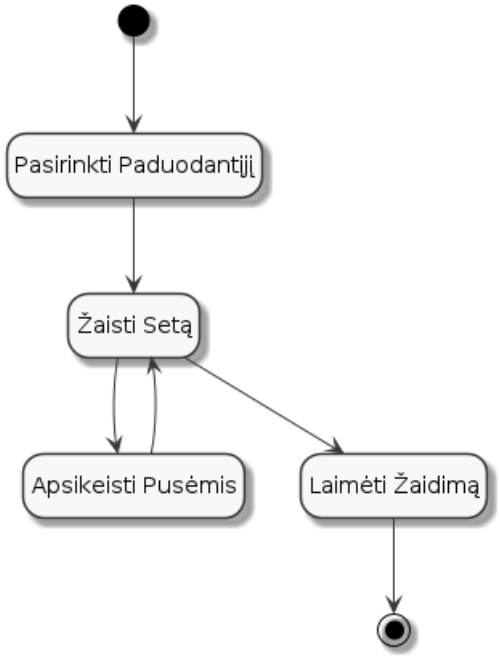
2.5. Stalo teniso žaidimo projektavimas

Remiantis oficialiomis stalo teniso taisyklėmis, galima sudaryti baigtinį automatą, padedantį kompiuterinės regos sistemai įvertinti žaidimo būklę bei sekti žaidimo rezultatą. Šiame skyriuje pateikiama baigtinių automatų būsenos mašinų schemos bei aprašomi perėjimai tarp būsenų.

Siekiant palengvinti baigtinio automato įgyvendinimą, visas žaidimo būsenų automatas yra

išskaidomas į skirtingus abstrakcijos lygius: taško, seto bei viso žaidimo. Atitinkamai, kiekvienas iš šių būsenos automatų turi būseną apibendrinančią žemesnį abstrakcijos lygmenį. Būsenų lentelių *įvykių* stulpelyje yra pateikiamos nuorodos į tarptautinės stalo teniso federacijos patvirtintas stalo teniso taisykles [Fed21].

2.5.1. Žaidimo būsenų automatas



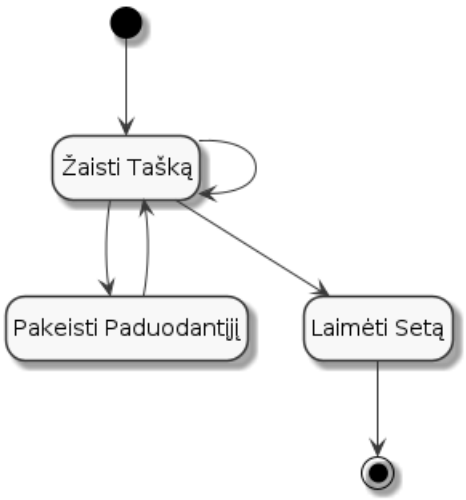
17 pav. Žaidimo būsenų automatas

Žaidimo lygio būsenų automatas apibūdina visos stalo teniso partijos būklę. Kai šis automatas patenka į baigtinę būseną yra traktuojama, kad stalo teniso žaidimas tarp žaidėjų yra pabaigtas ir galima nustatyti nugalėtoją. Žaidimo būsenų automatų būsenos bei perėjimai tarp jų yra aprašomi lentelėje nr. 2.

2 lentelė. Žaidimo būsenos

Dabartinė Būsena	**Įvykis**	**Veiksmas**	**Kita Būsena**
-	Žaidėjai susirenka prie stalo teniso stalo	Nustatyti kairįjį bei dešinįjį žaidėjus	Pasirinkti Paduodantįjį
Pasirinkti Paduodantįjį	Žaidėjai burtų keliu išsirenka paduodantįjį pagal 2.13.1	Nustatyti paduodančiąją stalo pusę	Žaisti Setą
Žaisti Setą	Seto pabaiga, kai sužaistas maksimalus setų skaičius pagal 2.12.1	-	Laimėti Žaidimą
	Seto pabaiga, kai sužaista mažiau nei maksimalus setų skaičius pagal 2.12.1	-	Apsikeisti Pusėmis
Apsikeisti Pusėmis	Žaidėjai apsikeitė pusėmis	Nustatyti kairįjį bei dešinįjį žaidėjus	Žaisti Setą
Laimėti Žaidimą	-	Užfiksuoti statistiką	-

2.5.2. Seto būsenų automatas



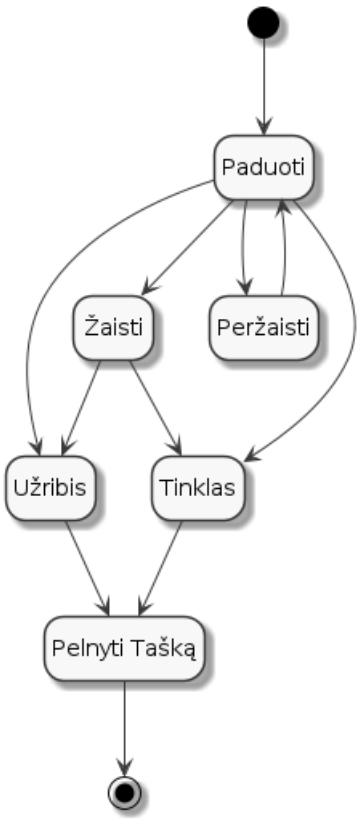
18 pav. Seto būsenų automatas

Kai žaidimo būsenų automatas patenka į *Žaisti Setą* būseną, yra aktyvuojamas seto būsenos automatas. Jo būsenų diagrama yra pateikiama pav. 18. Seto būsenų automatų būsenos bei perėjimai tarp jų yra aprašomi lentelėje nr. 3.

3 lentelė. Seto būsenos

Dabartinė Būsena	**Įvykis**	**Veiksmas**	**Kita Būsena**
-	-	Nustatyti paduodantįjį žaidėją pagal paduodančiąją stalo pusę	Žaisti Tašką
Žaisti Tašką	Taškas pelnytas, bet paduodantysis žaidėjas padavė mažiau kartų negu nurodo 2.13.3	-	Žaisti Tašką
	Taškas pelnytas, paduodantysis žaidėjas padavė tiek kartų, kiek nurodo 2.13.3	-	Pakeisti Paduodantįjį
	Taškas pelnytas, tenkinama 2.11.1 sąlyga	-	Laimėti Setą
Pakeisti Paduodantįjį	-	Nustatyti paduodantįjį žaidėją	Žaisti Tašką
Laimėti Setą	-	Užfiksuoti statistiką	-

2.5.3. Taško būsenų automatas



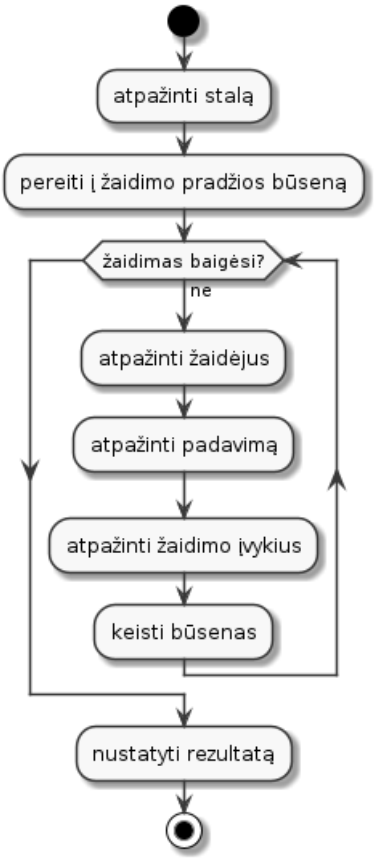
19 pav. Taško būsenų automatas

19 pav. yra pateikiama autoriaus parengta būsenų diagrama, apibūdinanti vieną tašką seto metu, remiantis lauko teniso būsenų diagrama [HC10]. Būsenų ir perėjimų aprašymai pateikiami lentelėje nr. 4.

4 lentelė. Taško būsenos

Dabartinė Būsena	**Įvykis**	**Veiksmas**	**Kita Būsena**
-	Žaidėjas išmetė kamuoliuką pagal 2.6	Nustatyti mušantįjį ir atmušantįjį žaidėjus	Paduoti
Paduoti	Kamuoliukas atsimušė į tinklą arba kitaip palietė tinklo konstrukciją pagal 2.10.1.5	-	Tinklas
	Kamuoliukas kliudė tinklą pagal 2.9.1	-	Peržaisti
	Kamuoliukas atsimušė į mušančiojo bei atmušančiojo žaidėjo stalo puses pagal 2.6.3	Sukeisti mušantįjį ir atmušantįjį žaidėjus	Žaisti
	Kamuoliukas neatsimušė pagal 2.6.3	-	Užribis
Peržaisti	Žaidėjas išmetė kamuoliuką pagal 2.6	Nustatyti mušantįjį ir atmušantįjį žaidėjus	Paduoti
Žaisti	Mušantysis žaidėjas atmušė kamuoliuką	-	Žaisti
	Kamuoliukas atsimušė į atmušančiojo žaidėjo stalo pusę	Sukeisti mušantįjį ir atmušantįjį žaidėjus	Žaisti
	Kamuoliukas neatsimušė į atmušančiojo žaidėjo stalo pusę	-	Užribis
	Kamuoliukas atsimušė į tinklą arba kitaip palietė tinklo konstrukciją pagal 2.10.1.5	-	Tinklas
Užribis	-	Užfiksuoti statistiką	Pelnyti Tašką
Tinklas	-	Užfiksuoti statistiką	Pelnyti Tašką
Pelnyti Tašką	-	Skirti tašką atmušančiajam žaidėjui	-

2.6. Abstraktus sistemos veikimo principas



20 pav. Sistemos veikimo diagrama

Remiantis kamuoliuko trajektorijos atpažinimo algoritmu bei būsenų automatais, pateiktais ankstesniuose skyriuose, galima apibrėžti abstraktų sistemos veikimo principą, kurio remiantis galima įgyvendinti stalo teniso žaidimo sekimą (20 pav.).

Sistema sudaroma iš šių žingsnių:

1. stalo kraštinių bei plokštumos identifikavimo;
2. būsenų automato inicializavimo;
3. tuomet, kol automatas nepasiekė pabaigos būsenos:
   (a) žaidėjų pozicijos kadre identifikavimo;
   (b) padavimo nustatymo;
   (c) žaidimo įvykių nustatymo;
   (d) automato būsenų perėjimų vykdymo;
4. žaidimo rezultato nustatymo.

2.6.1. Tyrimo apribojimai

Atliekant praktinius eksperimentus, paaiškėjo, kad negalima vienareikšmiškai nuspręsti, kada žaidėjai pradeda padavimą. Pavyzdžiui, taškui neprasidėjus, žaidėjams stovint prie stalo, žaidėjai mėto kamuoliuką į orą arba mušinėja jį į stalą. Nors priešininkas netraktuoja šių veiksmų kaip

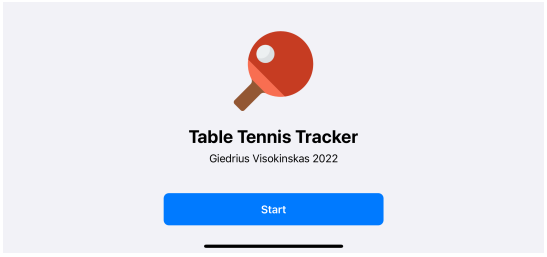
padavimo, kompiuterinės regos sistema negali jų išskirti ir dėl to juos identifikuoja kaip neleistinus padavimus. Nors yra darbų, tiriančių stalo teniso padavimą [Rad18], jie taip pat apsiriboja tuo, kad kiekvienas užfiksuotas kamuoliuko išmetimas žaidėjams esant prie stalo yra traktuojamas kaip padavimas. Žaidėjo ketinimų nustatymas galėtų būti tiriamas kitame darbe, pavyzdžiui, sukuriant neuroninį tinklą, gebantį nustatyti padavimo pradžią remiantis žaidėjo padėtimi kadre, stovėsena bei priešininko reakcija.

Praktinių eksperimentų metu taip pat nustatyti atvejai, kai kamuoliukas tik užkabina stalo kraštą, tačiau nepakeičia trajektorijos. Tai trikdo tvarų žaidimo sekimą, nes remiantis stalo teniso taisyklėmis [Fed21], tokie įvykiai turėtų būti laikomi atsimušimais. Peržiūrėjus bandymų vaizdo medžiagą, pastebėta, kad nors vizualiai kamuoliuko trajektorija nepasikeičia, tačiau galima išgirsti savitą atsimušimo garsą. Toks kamuoliuko atsimušimo nustatymo būdas taip pat galėtų būti tiriamas kitame darbe, pavyzdžiui, sukuriant sistemą, gebančią atpažinti stalo teniso žaidimo įvykius pagal garsą.

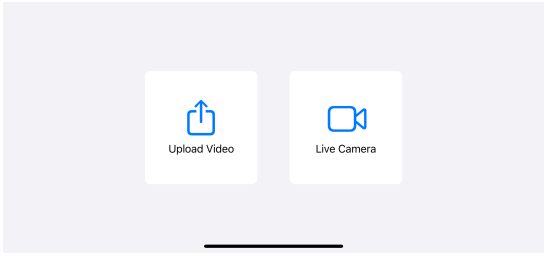
Atsižvelgiant į šias problemas bei siekiant suvaldyti tyrimo apimtį, tolesnį tyrimą bei pavyzdinės programėlės įgyvendinimą buvo nuspręsta atlikti su šiais ribojimais:

- kiekvienas kamuoliuko išmetimas, atpažintas prie vieno iš žaidėjų yra traktuojamas kaip padavimas;
- padavimo metu nėra fiksuojamas tinklelio palietimas;
- nėra fiksuojami neleistini atmušimai, tol, kol kamuoliukas atsimuša priešininko stalo pusėje;
- nėra fiksuojami atsimušimai, kai kamuoliukas tik paliečia stalo kraštą.

3. Pavyzdinės programėlės įgyvendinimas

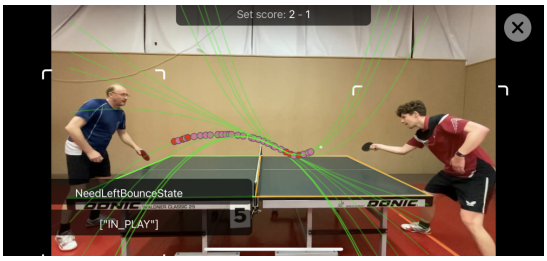


(a) Pradinis programėlės langas



(b) Įvesties pasirinkimas

(c) Stalo taškų pasirinkimas



(d) Žaidimo langas

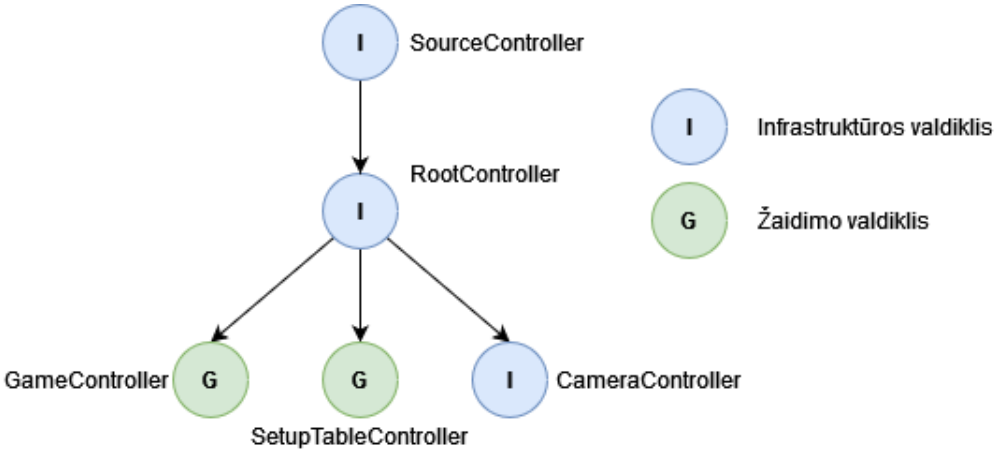
21 pav. Programėlės vartotojo sąsaja

Tyrimo metu buvo sukurta „iOS“ mobiliųjų telefonų operacinei sistemai skirta programėlė, įgyvendinti paprastąjį trajektorijos atpažinimo algoritmą bei dalinį stalo teniso žaidimo būsenų automatą. Įgyvendinimas atliktas remiantis „Apple“ kompanijos sukurtu *ActionAndVision* pavyzdžiu [App22a]. Programėlė geba atpažinti tik vieno seto rezultatus, kai žaidžia tik du žaidėjai.

Kuriant programėlę buvo panaudotas „Vision“ karkasas, skirtas palengvinti kompiuterinės regos algoritmų įgyvendinimą ir apdoroti autoriaus sukurtąjį „CoreML“ neuroninį tinklą bei „GameKit“ karkasas, palengvinantis būsenų automato įgyvendinimą. Programėlės veikimas yra suskirstytas į šiuos etapus (pav. 21):

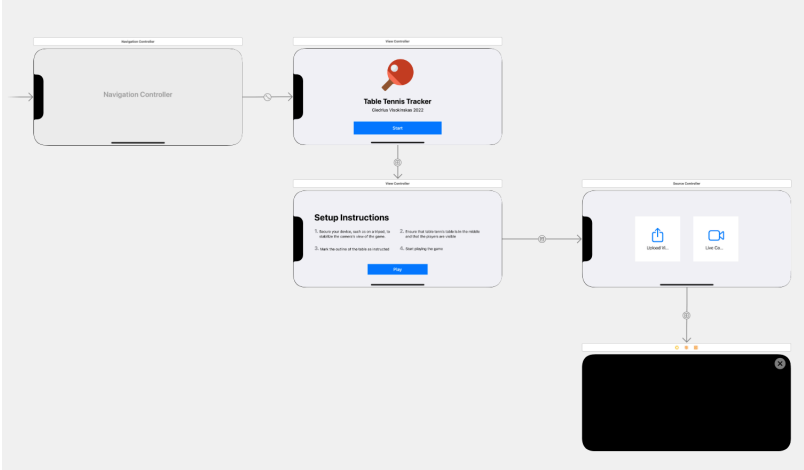
1. pradžios - laukiama, kol naudotojas paspaudžia *Start* mygtuką, jo metu į operatyviąją atmintį yra užkraunami kompiuterinės regos sistemos artefaktai;
2. įvesties pasirinkimo - leidžiantis vartotojui pasirinkti įvestį, kurią apdoros kompiuterinės regos sistema;
3. kadro paruošimo - laukiantis, kol scena stabilizuosis, o jai stabilizavusis, leidžiantis nurodyti stalo kampus bei tinklelio poziciją kadre;
4. žaidimo - laukiantis, kol kadre atsiras žaidėjai, o jiems atsiradus, pradedantis vieno seto rezultatų stebėjimą bei trajektorijų paišymą.

3.1. Architektūra



22 pav. Valdiklių hierarchija

Programėlė įgyvendinta remiantis modelio-vaizdo-valdiklio (*Model-View-Controller*) architektūra. Yra išskiriamos dvi pagrindinės valdiklių rūšys: infrastruktūros ir žaidimo logikos. Infrastruktūriniai valdikliai yra atsakingi už automato būsenos arba įvesties buferio atnaujinimo įvykių apdorojimą, ir atitinkamo žaidimo valdiklio instancijavimą (priklausomai nuo būsenos). Žaidimo valdikliai atsakingi už žaidimo logikos įgyvendinimą bei globalaus žaidimo modelio atnaujinimą.



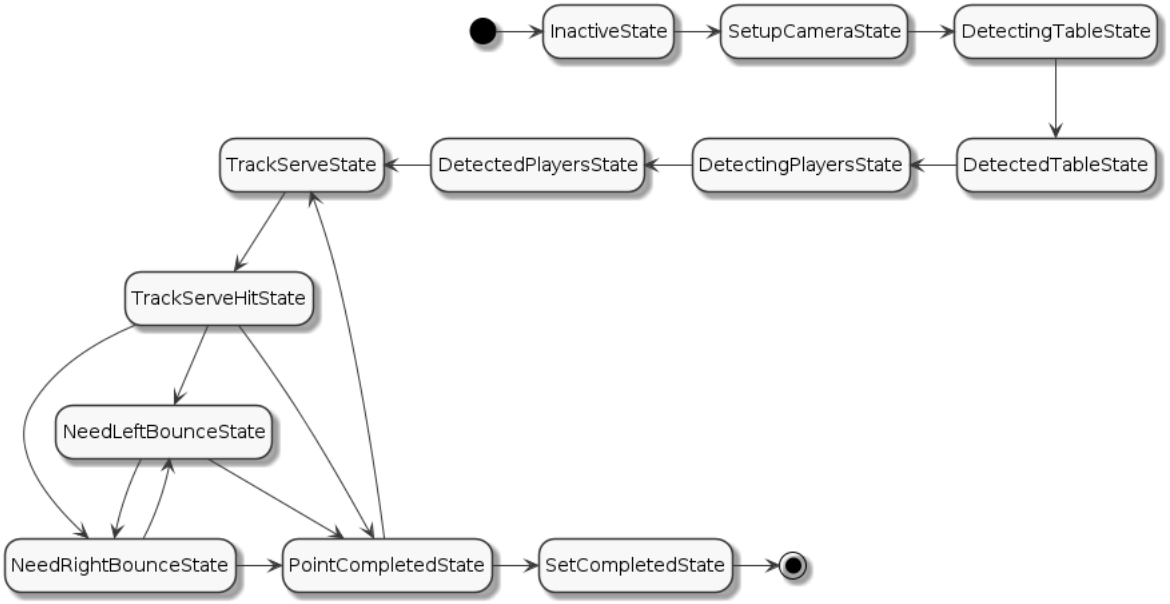
23 pav. Pradžios etapo siužetinė lenta

Programėlėje naudojami šie vaizdų tipai:

1. siužetinės lentos (*storyboard*, 23 pav.) - projektuojami naudojantis „Xcode“ įrankiu bei leidžiantys suprojektuoti daugiau nei vieną programėlės langą viename faile;
2. *xib* - projektuojami naudojantis „Xcode“ įrankiu, bet leidžiantys suprojektuoti tik vieną langą viename faile ir jį susieti tik su vienu valdikliu;
3. pagalbiniai (*helper*) - aprašomi kaip „Swift“ programavimo kalbos klasės, naudojami kaip komponentai kituose „vaizduose“.

Globalus žaidimo modelis yra įgyvendintas kaip *GKStateMachine* realizacija, sauganti visos programėlės būseną bet kuriuo programos veikimo momentu. Apie būsenos pasikeitimus valdikliai yra informuojami naudojantis *NotificationCenter* karkasą. „Vision“ vaizdo buferio apdorojimo užklausos yra vykdomos asinchroniškose darbų eilėse, taip užtikrinant, kad vartotojo sąsaja nebus paveikta ilgai trunkančių vaizdo apdorojimo algoritmų.

3.2. Būsenų automatas



24 pav. Programėlės būsenų automatas

Programėlė naudoja globalų būsenų automatą (24 pav.) su šiomis būsenomis:

1. *InactiveState* – pradžios etapo būsena, kraunami infrastruktūros valdikliai;
2. *SetupCameraState* – laukiama kol bus pasirinkta įvestis ir stabilizuosis kadras;
3. *DetectingTableState* – laukiama, kol naudotojas pasirinks stalo taškus;
4. *DetectedTableState* – stalo taškai pasirinkti;
5. *DetectingPlayersState* – laukiama, kol du žaidėjai pasirodys kadre;
6. *DetectedPlayersState* – žaidėjai nustatyti;
7. *TrackServeState* – laukiama, kol bus pradėtas padavimas;
8. *TrackServeHitState* – laukiama, kol kamuoliukas atsimuš paduodančiojo žaidėjo stalo pusėje;
9. *NeedLeftBounceState* – laukiama, kol kamuoliukas atsimuš kairėje stalo pusėje;
10. *NeedRightBounceState* – laukiama, kol kamuoliukas atsimuš dešinėje stalo pusėje;
11. *PointCompletedState* – taškas sužaistas;
12. *SetCompletedState* – setas sužaistas.

3.2.1. Perėjimai tarp būsenų

Pavyzdinis įgyvendinimas remiasi idėja, kad sužaidus tašką, kamuoliukas, atitinkantis reikalavimus aprašytus sistemos projektavimo skyriuje, kurį laiką nebus aptinkamas kadre. Dėl to, kai trajektorija yra išvaloma pagal paprastąjį algoritmą, sistema reaguoja į paskutinę automato būseną ir pagal dabartinę sistemos būklę parenka perėjimą. Pavyzdžiui, jeigu sistemai esant *NeedLeftBounceState*, kamuoliukas nėra aptinkamas n kadrų iš eilės, traktuojama, kad kamuoliukas nepasiekė kairės stalo pusės ir dėl to įvyksta perėjimas į *PointCompletedState*, ir kairėje pusėje esančiam žaidėjui yra skiriamas taškas.

3.3. „Vision“ karkaso panaudojimas

„Vision“ karkasas programėlėje naudojamas šiems kompiuterinės regos uždaviniams atlikti:

- kadro stabilumui nustatyti - naudojamas *VNSequenceRequestHandler* kartu su *VNTranslationalImageRegistrationRequest* siekiant suskaičiuoti kadrų sekos taškų slankųjį vidurkį;
- žaidėjams nustatyti - naudojamas *VNDetectHumanRectanglesRequest* nustatyti žmonėms, esantiems kadre;
- kamuoliukui identifikuoti - naudojamas *VNCoreMLRequest*, su autoriaus apmokytu neuroniniu tinklu.

Svarbu paminėti, kad „Vision“ karkasas dirba su normalizuota koordinačių plokštuma, bei „apversta“ Y ašimi. Dėl to norint atvaizduoti rezultatus naudojantis „Core Graphics“ (standartine „iOS“ biblioteka), juos iš pradžių reikia transformuoti, pridedant kadro saugumo maržas, apverčiant Y ašį bei konvertuoti gautą rezultatą iš normalizuotų koordinačių į standartines. Šis procesas taip pat turi būti atliekamas jeigu norima apdoroti tik dalį įvesties kadro, pavyzdžiui bandant aptikti žaidėją, esantį kairėje stalo pusėje. „Apple“ nepateikia tinkamų naudoti pagalbinių funkcijų, dėl to šios transformacijos turėjo būti įgyvendintos pačioje programėlėje. Tai prailgino įgyvendinimą bei apsunkino programos testavimą.

4. Kompiuterinės regos sistemos bandymai ir vertinimas

Sukurta pavyzdinė programėlė buvo testuojama su autoriaus surinktu 37 minučių ir 41 sekundės trukmės, 240 kps duomenų rinkiniu, nufilmuotu su „Apple iPhone 12 PRO“. Surinkti duomenys buvo suskirstyti į šias grupes:

- Idealių sąlygų K_{i}:
  - žaidimas vyksta aplinkoje, kuria remiantis buvo sukurtas konvoliucinio neuroninio tinklo duomenų rinkinys;
  - stalo teniso stalas yra matomas 100% viso žaidimo laiko;
  - abu žaidėjai yra pilnai matomi (įskaitant visas galūnes) kadruose bent 90% viso žaidimo laiko;
  - žaidimo metu žaidėjai laikosi oficialių stalo teniso taisyklių [Fed21], sąmoningai jų nepažeidžia;
  - kameros padėtis nekinta viso žaidimo metu.
- Įprastų sąlygų K_{n}:
  - žaidimas vyksta aplinkoje, atitinkančioje oficialias stalo teniso taisykles [Fed21];
  - stalo teniso stalas bei tinklelis yra matomas 100% viso žaidimo laiko;
  - abu žaidėjai yra matomi kadruose bent 70% viso žaidimo laiko;
  - žaidėjai laikosi stalo teniso taisyklių, bet nesistengia sąmoningai jų nepažeidinėti;
  - kameros padėtis nekinta viso žaidimo metu.

Sistemos rezultatai yra lyginami su stalo teniso teisėjo vertinimu, klaidų skaičius yra nustatomas remiantis formule $E = |P_{\text{l}} - S_{\text{l}}| + |P_{\text{r}} - S_{\text{r}}|$, kur:

- P_{l}, P_{r} - atitinkamai, „kairiojo“ ir „dešiniojo“ žaidėjų rezultatai seto pabaigoje, pagal teisėjo vertinimą;
- S_{l}, S_{r} - atitinkamai, „kairiojo“ ir „dešiniojo“ žaidėjų rezultatai seto pabaigoje, pagal sistemos vertinimą.

4.1. Duomenų rinkinio paruošimas

Dėl sistemos apribojimų, apibūdintų sistemos projektavimo skyriuje, duomenų rinkinys turėjo būti specialiai apdorotas, iškarpant įvykius, galinčius klaidinti kompiuterinės regos sistemą. Šis darbas buvo atliktas naudojantis „Blender 3.1.2“ programine įranga. Duomenų apdorojimo palengvinimui buvo naudojami autoriaus parengti „Python“ scenarijai, skirti sukarpyti įrašus per pažymėtus raktinius kadrus. Taip pat buvo parengtas „Python“ scenarijus, įgalinantis daugiagijį (angl. *multithreaded*) galutinio vaizdo įrašo sukūrimą (angl. *render*).

4.2. Bandymai

Iš surinktos medžiagos buvo sudaryti treji vaizdo įrašai: dveji atitinkantys K_{i} ir vienas atitinkantis K_{n} sąlygas. Žemiau pateiktoje lentelėje nr. 5 aprašomi eksperimentai ir jų rezultatai:

5 lentelė. Kompiuterinės regos sistemos bandymai

Aplinka	P_l	P_r	S_l	S_r	E
K_i	3	11	3	10	1
K_i	11	6	10	5	2
K_n	11	9	11	6	3

4.2.1. Pirmasis bandymas

Pirmojo bandymo metu sistema neatpažino vieno taško, dėl to nenustatė seto pabaigos. Išanalizavus atpažinimo procesą pagal sistemos derinimo žurnalą, nustatyta, kad taip įvyko dėl to, kad po paskutiniojo taško pabaigos vaizdo įrašas iškart pasibaigė. Atsižvelgiant į sistemos architektūrą, tikėtina, kad jeigu įrašas būtų buvęs n (neatpažintų kadrų skaičiaus kintamasis) kadrų ilgesnis, sistema būtų pastebėjusi, kad kamuoliukas yra užribyje ir teisingai įskaičiusi tašką.

4.2.2. Antrasis bandymas

Antrojo bandymo metu, įvertinus derinimo žurnalą, nustatyta, kad sistemos vertinimą paveikė algoritmo trūkumai, šis setas pasižymėjo greitais smūgiais, dėl kurių sistema nominuodavo neteisingas trajektorijas, taip „pamesdama“ kamuoliuko padėtį. Dėl šios priežasties nebuvo užfiksuoti dveji padavimai.

4.2.3. Trečiasis bandymas

Nors trečiojo bandymo rezultato E reikšmė to neparodo, peržiūrėjus įvykių žurnalą, nustatyta, kad sistema tokį rezultatą parodė atsitiktinai. Šis bandymas buvo atliktas ne stalo teniso klube, bet sporto salėje, dėl to skyrėsi apšvietimo sąlygos bei žaidimo aplinka. Trijų padavimų metu trajektorijos atpažinimo algoritmas nerado patikimos trajektorijos, dėl ko žaidimo sekimas buvo pradedamas ne nuo pradžių. Taip pat, buvo pastebėti dveji atvejai, kai vieno taško metu sistema užfiksavo daugiau nei vieną taško pabaigos įvykį, taip pakeičiant seto rezultatą nuo realaus.

Manoma, kad taip galėjo įvykti dėl neuroninio tinklo persimokymo, nes tinklo testavimo duomenyse nebuvo kadrų, surinktų kituose stalo teniso klubuose ar aplinkose.

4.3. Bandymų apibendrinimas

Įvertinus surinktus rezultatus, galima daryti išvadą, kad sukurta pavyzdinė programėlė geba atpažinti ir sekti stalo teniso žaidimo eigą, tačiau tai daro su klaidomis ir esant idealioms sąlygoms K_i. Prie šių sąlygų galima tikėtis iki dviejų taškų paklaidos. Tačiau, esant normaliomis sąlygomis K_n klaidų skaičius gerokai didesnis, taip nutinka dėl to, nes kamuoliuko padėtis nėra patikimai nustatoma.

Pastebėta, kad net taikant „Python“ scenarijų, testavimo duomenų parengimas užtruko ilgai, apie 8 valandas. Ši problema galėtų būti sprendžiama sukuriant sintetinių partijų generavimo sistemą, gebančią sukurti stalo teniso partiją iš testinių duomenų pagal nurodytus požymius be vartotojo įsikišimo.

Rezultatai ir išvados

Šio magistrinio tyrimo metu buvo sukurta mobilioji kompiuterinės regos sistemos architektūra bei pavyzdinis jos įgyvendinimas, kuriuo remiantis galima sekti stalo teniso žaidimą. Nors gauti bandymų rezultatai parodė, kad tokia sistema gali sekti žaidimo eigą, tačiau norint pilnai pakeisti stalo teniso teisėją, reikia pagerinti žaidimo įvykių atpažinimą bei sumažinti atpažinimo klaidų skaičių. Darbo metu daugiausiai dėmesio skirta paprastojo trajektorijos algoritmo kūrimui bei praktiniams bandymams, tolesniuose tyrimuose siūloma atsižvelgti į išimtinius atvejus.

Rezultatai

Šio darbo metu buvo atliktos šios užduotys:

1. Remiantis literatūros apžvalga, įvertinti reikalavimai, algoritmai bei programinės įrangos karkasai, galintys padėti įgyvendinti stalo teniso žaidimo sekimą.
2. Surinktas 542 640 kadrų stalo teniso duomenų rinkinys:
    - sudarytas anotuotas stalo teniso kamuoliukų duomenų rinkinys iš 43 953 kadrų.
3. Sukurtas paprastasis trajektorijos atpažinimo algoritmas:
    - apmokytas „YOLO v2“ architektūra paremtas neuroninis tinklas, skirtas identifikuoti kamuoliukus;
    - eksperimentais parinkti nepriklausomų Kalmano filtrų kovariacijos įverčiai;
    - eksperimentais parinkti algoritmo kintamųjų n, c bei s įverčiai.
4. Suprojektuotas būsenų automatas, naudojamas sekti stalo teniso žaidimo eigą kompiuterinės regos sistemoje.
5. Sukurta pavyzdinė programėlė, įgyvendinta „iOS“ sistemoje, demonstruojanti būsenų automato bei trajektorijos atpažinimo algoritmo darbą.
6. Aprašyti algoritmo efektyvumo vertinimo kriterijai.
7. Atlikti bei aprašyti bandymai su pavyzdine kompiuterinės regos sistemos realizacija.

Tyrimo eigoje taip pat buvo įgyvendinti papildomi „Python“ scenarijai, palengvinantys skirtingų neuroninio tinklo duomenų anotacijų formatų konvertavimą, „Blender“ raktinių kadrų automatinį skaidymą bei „Blender“ daugiagijį vaizdo apdorojimą.

Išvados

Tyrimo metu nustatyta, kad:

1. „Vision“ karkaso trajektorijos atpažinimo algoritmas negali atpažinti trajektorijų, kurias generuoja stalo teniso kamuoliukas.
2. Autoriaus sukurtas atpažinimo algoritmas geba atpažinti trajektorijas, tačiau tai daro su klaidomis.
3. Autoriaus sukurtas neuroninis tinklas kamuoliukams atpažinti patikimai veikia tik esant idealioms sąlygoms K_{i}.
4. Nešiojamoji kompiuterinės regos sistema gali būti įgyvendinta ir gali sekti stalo teniso žaidimą.

5. Pavyzdinis įgyvendinimas „Apple iPhone 12 Pro“ telefone geba apdoroti 70 kadrų per sekundę.
6. Stalo teniso žaidimo eigą mėgėjiškoje aplinkoje galima nustatyti su viena kamera, nekuriant trimatės (kelių kamerų) projekcijos.
7. Norint įgyvendinti pilną stalo teniso žaidimo sekimą, reikia atsižvelgti ir į išimtinius atvejus.

Galima daryti išvadą, kad mobiliajame telefone galima įgyvendinti stalo teniso žaidimo sekimo kompiuterinės regos sistemą, tačiau žaidėjai turi elgtis atsižvelgiant į sistemos ribojimus.

Tolesnės tyrimo kryptys

Norint įgyvendinti sklandų stalo teniso žaidimo sekimą, kompiuterinės regos sistema turi gebėti identifikuoti ne tik įvykius kadre, bet ir jų priežastis. Atsižvelgiant į tyrimo rezultatus bei išvadas, tolesniuose darbuose siūloma gerinti pasiūlytąją sistemą:

- ištiriant būdus nustatyti taško pradžią pagal abiejų žaidėjų padėtį kadre bei jų reakcijas;
- įtraukiant taisykles pritaikytas neįgaliesiems bei žaidimą poromis;
- praplečiant kamuoliuko atsimušimo nustatymą, pridedant garso apdorojimą;
- sudarant atnaujintą neuroninio tinklo modelį su paįvairintu duomenų rinkiniu, į jį įtraukiant kadrus iš skirtingų stalo teniso klubų.

Taip pat siūloma įgyvendinti testavimo duomenų generavimo sistemą, gebančią iš testinių duomenų sukurti sintetines stalo teniso partijas, pasižyminčias tam tikrais požymiais, taip palengvinant tolesnius tyrimus.

Literatūra

[AB14] J. Anaya ir A. Barbu. RENOIR - A Benchmark Dataset for Real Noise Reduction Evaluation, 2014. arXiv: 1409.8230.

[Ale17] H. Aleš. *Implementing and Applying Fast Moving Object Detection on Mobile Devices*. Magistro darbas, Czech Technical University in Prague, 2017.

[App17] Apple. WWDC 2017 - Videos. https://developer.apple.com/videos/wwdc2017/, 2017. Tikrinta 2020-05-23.

[App20a] Apple. Core ML | Apple Developer Documentation. https://developer.apple.com/documentation/coreml/, 2020. Tikrinta 2020-06-07.

[App20b] Apple. WWDC 2020 - Videos. https://developer.apple.com/videos/play/wwdc2020/10099/, 2020. Tikrinta 2021-11-01.

[App21a] Apple. Create ML Overview - Machine Learning - Apple Developer. https://developer.apple.com/machine-learning/create-ml/, 2021. Tikrinta 2021-11-02.

[App21b] Apple. Metal Overview - Apple Developer. https://developer.apple.com/metal/, 2021. Tikrinta 2021-11-02.

[App22a] Apple. Building a Feature-Rich App for Sports Analysis | Apple Developer Documentation. https://developer.apple.com/documentation/vision/building_a_feature-rich_app_for_sports_analysis, 2022. Tikrinta 2022-01-06.

[App22b] Apple. Finding an Interpolating Polynomial Using the Vandermonde Method | Apple Developer Documentation. https://developer.apple.com/documentation/accelerate/finding_an_interpolating_polynomial_using_the_vandermonde_method/, 2022. Tikrinta 2022-04-17.

[App22c] Apple. iPhone 12 Pro - Technical Specifications. https://support.apple.com/kb/SP831?locale=en_US, 2022. Tikrinta 2022-04-01.

[DWP+11] X. Dong, G. Wang, Y. Pang, W. Li, J. Wen, W. Meng ir Y. Lu. Fast efficient algorithm for enhancement of low lighting video. *2011 IEEE International Conference on Multimedia and Expo*, p. 1–6, 2011.

[EEV+15] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn ir A. Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111:98–136, 2015-01.

[Fed21] I. T. T. Federation. ITTF Handbook of 2021. https://documents.ittf.sport/sites/default/files/public/2021-08/2021ITTFHandbook_v2_clean_version_1.pdf, 2021. Tikrinta 2021-11-19.

[GDD+14] R. Girshick, J. Donahue, T. Darrell ir J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *Computer Vision and Pattern Recognition*, 2014. arXiv: 1311.2524.

[Gir15] R. B. Girshick. Fast R-CNN, 2015. arXiv: 1504.08083.

[HC10] Q. Huang ir S. Cox. Using high-level information to detect key audio events in a tennis game. P. 1409–1412, 2010-01.

[HLC+19] Y. Huang, I. Liao, C. Chen, T. İk ir W. Peng. TrackNet: A Deep Learning Network for Tracking High-speed and Tiny Objects in Sports Applications*. *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, p. 1–8, 2019.

[Hra17] A. Hrabalík. *Implementing and Applying Fast Moving Object Detection on Mobile Devices*. Magistrinis darbas, Czech technical university in Prague, 2017.

[Inc22a] M. Inc. SwingVision: A.I. Scoring, Stats & Line Calling for Tennis. https://swing.tennis/, 2022. Tikrinta 2022-04-15.

[Inc22b] N. T. Inc. HomeCourt. https://www.homecourt.ai/, 2022. Tikrinta 2022-04-15.

[ITT19] I. T. T. F. (ITTF). Table Tennis Review to be implemented at 2019 ITTF World Tour Grand Finals - International Table Tennis Federation. https://www.ittf.com/2019/12/10/table-tennis-review-implemented-2019-ittf-world-tour-grand-finals/, 2019. Tikrinta 2022-04-15.

[JZS+18] Y. Ji, J. Zhang, Z. Shi, M. Liu ir J. Ren. Research on real – time tracking of table tennis ball based on machine learning with low-speed camera. *Systems Science & Control Engineering*, 6:71–79, 2018. DOI: 10.1080/21642583.2018.1450167.

[Kal60] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960-03. ISSN: 0021-9223. DOI: 10.1115/1.3662552. eprint: https://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/82/1/35/5518977/35_1.pdf. URL: https://doi.org/10.1115/1.3662552.

[Kam04] T. Kamei. Method and system for tracking a fast moving object. 2004-11. URL: https://patents.google.com/patent/US6819778.

[KKL+20] V. Kakani, H. Kim, J. Lee, C. Ryu ir M. Kumbham. Automatic Distortion Rectification of Wide-Angle Images Using Outlier Refinement for Streamlining Vision Tasks. *Sensors*, 20:894, 2020-02. DOI: 10.3390/s20030894.

[Let19] G. Letts. Maximum Speed of a Ping Pong Ball. https://www.liveabout.com/ping-pong-ball-maximum-speed-3974874/, 2019. Tikrinta 2019-12-07.

[LH19] H. Lin ir Y. Huang. Ball Trajectory Tracking and Prediction for a Ping-Pong Robot. *2019 9th International Conference on Information Science and Technology (ICIST)*, p. 222–227, 2019. DOI: 10.1109/ICIST.2019.8836894.

[Luk16] N. Lukáš. *Detection of Fast Moving Objects*. Bakalauro darbas, Czech Technical University in Prague, 2016.

[Ma-18] Ma-Dan. YOLOv3-CoreML. `https://github.com/Ma-Dan/YOLOv3-CoreML/`, 2018. Tikrinta 2020-06-06.

[Mao06] J. Mao. *Tracking a tennis ball using image processing techniques*. Magistro darbas, University of Saskatchewan, 2006.

[MOW+07] H. Malm, M. Oskarsson, E. Warrant, P. Clarberg, J. Hasselgren ir C. Lejdfors. Adaptive enhancement and noise reduction in very low light-level video. *2007 IEEE 11th International Conference on Computer Vision*, p. 1–8, 2007.

[MRT13] P. Martinchek, N. Redmon ir I. Thobani. *Low Light Mobile Video Processing*, Stanford University, 2013.

[Old15] K. M. Oldham. *Table tennis event detection and classification*, Loughborough University, 2015-01. HDL: 2134/19626.

[Old19] K. M. Oldham. Table tennis event detection and classification, 2019-08. URL: `https://hdl.handle.net/2134/19626`.

[Rad18] R. Radžiūnas. *Stalo teniso nelegalaus padavimo atpažinimas*. Bakalauro darbas, Vilniaus Universitetas, 2018.

[RF16] J. Redmon ir A. Farhadi. YOLO9000: Better, Faster, Stronger. *CoRR*, abs/1612.08242, 2016. arXiv: 1612.08242. URL: `http://arxiv.org/abs/1612.08242`.

[RF18] J. Redmon ir A. Farhadi. YOLOv3: An Incremental Improvement, 2018. arXiv: 1804.02767.

[RHG+15] S. Ren, K. He, R. Girshick ir J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[RJW96] Z. Rahman, D. J. Jobson ir G. A. Woodell. Multi-scale retinex for color image enhancement. *Proceedings of 3rd IEEE International Conference on Image Processing*, tom. 3, p. 1003–1006, 1996.

[RMM+18] V. Renò, N. Mosca, R. Marani, M. Nitti, T. D'Orazio ir E. Stella. Convolutional Neural Networks Based Ball Detection in Tennis Games. P. 1839–18396, 2018-06. DOI: 10.1109/CVPRW.2018.00228.

[SEG10] D. Schofield, M. Evison ir L. Goodwin. *Influence of Lens Distortion and Perspective Error*. CRC Press, Boca Raton, 2010-03, p. 101–118. ISBN: 9781439811344. DOI: 10.1201/9781439811344.

[SJ17] H. Su ir C. Jung. Low light image enhancement based on two-step noise suppression. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, p. 1977–1981, 2017.

[TDR19] M. Teimouri, M. Delavaran ir M. Rezaei. *A Real-Time Ball Detection Approach Using Convolutional Neural Networks*. Springer, 2019-12, p. 323–336. ISBN: 978-3-030-35698-9. DOI: 10.1007/978-3-030-35699-6_25.

[Too21] O. Toolkit. Powerful and efficient Computer Vision Annotation Tool (CVAT). https://github.com/openvinotoolkit/cvat, 2021. Tikrinta 2021-11-02.

[Tri15] R. Triggs. How far we've come: a look at smartphone performance over the past 7 years. https://www.androidauthority.com/smartphone-performance-improvements-timeline-626109/, 2015. Tikrinta 2022-04-15.

[ZSL+12] X. Zhang, P. Shen, L. Luo, L. Zhang ir J. Song. Enhancement and noise reduction of very low light level images. *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, p. 2034–2037, 2012.