

**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**INFORMATIKOS FAKULTETAS**

**Tadas Ivanovas**

**LAZERIO LINIJOS APTIKIMO ALGORITMAS**

Baigiamasis magistro projektas

|  |  |
| --- | --- |
|  | **Vadovas**  Doc. Armantas Ostreika |

**KAUNAS, 2020**

Turinys

[**KAUNO TECHNOLOGIJOS UNIVERSITETAS** 1](#_Toc70455438)

[**INFORMATIKOS FAKULTETAS** 1](#_Toc70455439)

[**Tadas Ivanovas** 1](#_Toc70455440)

[**LAZERIO LINIJOS APTIKIMO ALGORITMAS** 1](#_Toc70455441)

[Turinys 2](#_Toc70455442)

[Įvadas 3](#_Toc70455443)

[1. Lazerio linijos aptikimo metodų apžvalga 4](#_Toc70455444)

[1.1. Kelio linijų aptikimo algoritmas 4](#_Toc70455445)

[1.2. Lazerio linijos aptikimas skirtas suvirinimo linijų nustatymui 6](#_Toc70455446)

[1.3. Kliūčių aptikimas naudojant lazerio liniją 8](#_Toc70455447)

[1.4. Lazerio linijos aptikimas atstumo nustatymui 9](#_Toc70455448)

[1.5. 3D lazerio linijos skeneris 12](#_Toc70455449)

[1.5.1. Baltos šviesos filtravimas 12](#_Toc70455450)

[1.5.2. Lazerio linijos taškų radimas 13](#_Toc70455451)

[1.6. „Hough“ transformacija 14](#_Toc70455452)

[1.7. Furjė Transformacija 15](#_Toc70455453)

[1.7.1. „Cooley-Tukey“ greitosios Furjė transformacijos algoritmas 17](#_Toc70455454)

[1.8. Aukšto dažnio filtrai 18](#_Toc70455455)

[1.8.1. Idealusis filtras 19](#_Toc70455456)

[1.8.2. „Butterworth“ filtras 19](#_Toc70455457)

[1.8.3. Gauso filtras 20](#_Toc70455458)

[2. Lazerio linijos aptikimo metodologija 22](#_Toc70455459)

[2.1.1. Aukšto dažnio filtrų tyrimas 22](#_Toc70455460)

[2.1.2. Dažnių spektro filtravimas 28](#_Toc70455461)

[2.2. Filtro dydžio parinkimas 30](#_Toc70455462)

[Išvados ir rezultatai 33](#_Toc70455463)

[Literatūros sąrašas 34](#_Toc70455464)

Įvadas

Lazeris – įrenginys, spinduliuojantis koherentišką šviesą. Paprastai lygiagrečiu srautu lazeris skleidžia monochromatinę (vieno bangos ilgio) šviesą. Lazerio linijos aptikimas yra labai svarbus uždavinys robotikos, automatizavimo bei kompiuterinės regos uždavinys. Naudojant lazerio spindulio projektuotą liniją yra kuriamos trimatės objektų rekonstrukcijos, objektų ir kliūčių aptikimo, aplinkos žemėlapio braižymo sistemos.

Šiandien jau yra aibė pasiūlytų, sukurtų ir realizuotų lazerio linijos aptikimo sistemų. Daugelis jų yra skirtos pramonei, atitinkančios aukštus reikalavimus ir sąlyginai brangios. Jos paprastai naudojamos objektų orientacijos, padėties, tūrio ar paviršiaus ploto nustatymui. Nors tokios sistemos ir yra patikimos, jos paprastai būna stacionarios, pritvirtintos virš konvejerių ar kitų konstrukcijų, todėl jų panaudojimas mobilaus roboto aplikacijoje tampa komplikuotas. Dėl to atsiranda pigesnių, paprastesnių, reikalaujančių mažesnių resursų lazerio linijos aptikimo sistemų poreikis.

Tokią sistemą sudarytų paprasta vaizdo kamera ir lazerio linijos įrenginys, kuris projektuoja ryškią liniją ant įvairių paviršių. Tačiau pasitelkus vien paprastą vaizdo kamerą kyla nemažai linijos aptikimo iššūkių, pavyzdžiui, lazerio šviesos sodrumo reiškinys, baltos aplinkos šviesos poveikis ir lazerio nuskaitymo linijos segmentacija.

Daugelis šiandien kompiuterinėje regoje naudojamų lazerio linijos aptikimo algoritmų yra paremti spalvų segmentavimu arba paprastomis slenkstinėmis ribomis, kurios turi būti nustatomos rankiniu būdu. Tokie būdai nėra adaptyvūs, esant sudėtingomis fono sąlygomis arba pakitus aplinkos apšvietai, visas nustatytas slenkstines ribas reikia derinti iš naujo. Taip pat analizuojant literatūrą buvo pastebėta, jog daugelis pateiktų metodų veikia paprastame fone, kai kameros vaizdas yra juodas, o jame ryškiai yra matoma raudona lazerio linija. Apžvelgti algoritmai puikiai veikia tokiomis sąlygomis, bet tai riboja jų pritaikymo galimybes.

Atsižvelgiant į kilusias problemas, buvo pasiūlytas sprendimas, kuris, manoma, išspręstų daugelį minėtų uždavinių. Buvo pasiūlyta liniją aptikti atsižvelgiant į nuotraukos dažninį spektrą. Kadangi lazerio linija vaizde sukelia aukštus dažnius, mums pakanka išfiltruoti žemus dažnius ir galima nebekreipti dėmesio į spalvą. Taip išsprendžiama lazerio šviesos sodrumo problema. Taip pat, dažnių spektro filtravimas yra optimalus metodas, kadangi yra atliekama sąlyginai paprasta operacija – tereikia sudauginti dvi matricas. Toks būdas yra gerokai optimalesnis ir greitesnis už metodus, kurie veikia konvoliucijos principu.

Šiame darbe sprendžiama problema – kaip aptikti lazerio liniją kameros vaizde esant sudėtingomis fono sąlygomis taip pat išsprendžiant problemą, kai dėl nepakankamo kameros jautrumo, raudona lazerio linija per vidurį tampa balta.

**Darbo tikslas** – sukurti algoritmą, kuris aptiktų lazerio liniją kameros vaizde esant sudėtingomis fono sąlygomis.

**Pagrindiniai darbo uždaviniai:**

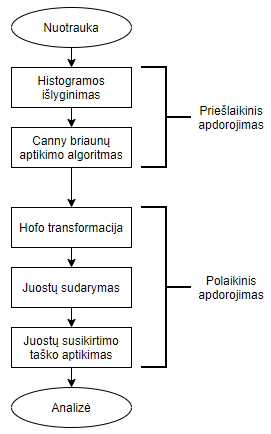
* Išanalizuoti jau esamus metodus, kurie naudojami lazerio linijos aptikimui, atlikti jų tyrimus.
* Sukurti algoritmą lazerio linijos aptikimui remiantis esamais metodais ir eliminuojant jų trūkumus.

# Lazerio linijos aptikimo metodų apžvalga

Šiame skyriuje yra aptariami jau naudojami linijos aptikimo algoritmai, metodai bei konkrečios aplikacijos paremtos jų panaudojimu. Daugelyje šaltinių, kartu su lazerio linijos aptikimo metodika bei algoritmais, yra pateikta informacija ir apie tam tikrą linijos aptikimo panaudojimo atvejį, pavyzdžiui, atstumo nustatymas pagal lazerio liniją. Tačiau plačiau yra aptariamos tik tos šaltinių dalys, kurios susijusios būtent su linijos aptikimo metodika ir tik minimaliai aptariami panaudos atvejai.

## Kelio linijų aptikimo algoritmas

Kelio juostų aptikimas vaidina svarbų vaidmenį išmaniųjų transporto priemonių sistemų srityje. Toliau yra pateikiamas kelio juostų aptikimo algoritmas skirtas aptikti kairę ir dešinę gatvės juostas [1]. Visa metodika susideda iš dviejų pagrindinių dedamųjų dalių: priešlaikinio ir polaikinio apdorojimo (žr. 1.1 pav.). Šiuo atveju dėmesys yra atkreipiamas tik į 4 algoritmo dalis (histogramos išlyginimas [144 p. 2], Canny briaunų aptikimo algoritmas [3], Hofo transformacija [226 p. 4] ir juostų sudarymas [2 p. 1]), kurios išaiškina patį linijų aptikimo veikimo principą.



1.1 pav. Juostų aptikimo algoritmo struktūrinė schema [1]

Pirmasis žingsnis esantis iš karto po vaizdo nuskaitymo yra histogramos išlyginimas. Šis metodas dažniausiai naudojamas siekiant padidinti bendrą vaizdo kontrastą. Tokia kontrasto korekcija yra atliekama perskirstant pikselių intensyvumo reikšmes per visą vaizdo histogramą. Dėl to neryškūs ir sunkiai matomi objektai vaizde įgyja didesnį kontrastą. Tai yra daroma siekiant išryškinti kelio linijų žymėjimus, kadangi dažnu atveju gali nutikti taip, jog pilkšvos linijos susilieja su pilku kelio asfaltu, dėl to tolimesnis vaizdo apdorojimas gali tapti žymiai sudėtingesnis.

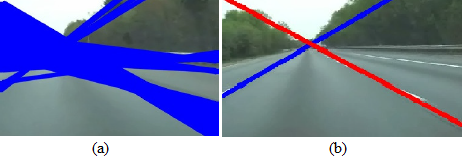
Toliau turimas RGB vaizdas yra paverčiamas į nespalvotą (angl. *grayscale*) vaizdą, kuris vėliau yra apdorojamas Canny briaunų aptikimo algoritmu. Jis grąžina binarinį vaizdą, kuris suteikia visą reikiamą informaciją aptinkant linijas (žr. 1.2 pav.). Siekiant sumažinti triukšmą gautame vaizde yra naudojamos morfologinės operacijos: skaitmeninė erozija[44 p. 4] ir skaitmeninis auginimas [42 p. 4]. Kartu yra pasitelkiamas ir vaizdo glotninimas [5].



1.2 pav. Canny briaunų aptikimo algoritmo veikimo pavyzdys (a) originalus nespalvotas vaizdas (b) Canny algoritmo apdorotas vaizdas su aptiktomis briaunomis [1]

Canny algoritmu gautame binariniame vaizde yra matomos ryškiai išskirtos briaunos (įvairūs kelio linijų žymėjimai). Linijų aptikimui ir identifikavimui toliau yra naudojama Hofo transformacija, kuri gali veikti esant būtent tik binariniam vaizdui. Pagrindinė jos paskirtis yra aptikti tiesias linijas ir kreives. Vienas iš didžiausių Hofo transformacijos privalumų yra tas, jog aptinkant linijas ji gali sujungti atskirus linijos segmentus, kurie priklauso tai pačiai linijai, o kaip jau žinoma, kelio linijų žymėjimas dažnai turi pertrauktas linijas.

Hofo transformacija vaizde aptinka ne tik ieškomas 2 linijas (kairiąją ir dešiniąją kelio juostas) bet ir daug pašalinių linijų, kurios kartais tik kerta ieškomas linijas (žr. 1.3 pav. (a)). Dėl to kelio juostų sudarymui ir aptikimui yra naudojami 2 linijų parametrai: posūkio kampas ir linijos statmens ilgis iki vieno iš vaizdo kampų ilgis. Tada sugrupavus visas linijas, kurių parametrai panašūs, yra apskaičiuojamas tų parametrų vidurkis, o galutinis rezultatas atvaizduojamas originaliame vaizde (žr. 1.3 pav. (b)), kur yra pažymimos kairioji ir dešinioji kelio juostos.



1.3 pav. Aptiktos linijos vaizde (a) Hofo transformacijos visos aptiktos linijos (b) išskirtos kairioji ir dešinioji kelio linijos [1]

Taigi, pateiktas algoritmas geba aptikti kairiąją ir dešiniąją kelio juostas esant realiomis lauko sąlygomis. Tačiau jis yra paremtas keliomis prielaidomis, kurios riboja algoritmo veikimo diapazoną:

* kelio linijos yra pakankamai ryškiai matomos nuoseklios;
* kelio plotis yra pastovus arba su minimaliais pokyčiais;
* kelio linijos turi sekti griežtas išvaizdos taisykles;
* negali būti ekstremalios oro sąlygos (liūtis, sniegas).

## Lazerio linijos aptikimas skirtas suvirinimo linijų nustatymui

Struktūrizuotos šviesos jutikliai pritraukia vis daugiau dėmesio ir yra plačiai naudojami įvairiose automatizavimo bei robotikos srityse, pavyzdžiui, automatinis virinimas, kokybės kontrolė ar robotų navigacija [6]. Taigi, toliau šiame poskyryje yra kalbama apie struktūrizuotos lazerio šviesos linijų aptikimą naudojant CCD (angl. *charged-coupled device*) kamerą [7].

Lazerio linijos aptikimo rezultatas daro didžiulę įtaką visiems matavimo rezultatams. Dėl to patikimumas, tikslumas ir aptikimo greitis yra labai svarbūs parametrai visai struktūrizuotos šviesos aptikimo sistemai. Toliau pateiktas aptikimo algoritmas iš esmės yra paremtas vaizdo apšviestumo histograma (angl. *luminance histogram*) ir slenkstinių ribų segmentavimo (angl. *threshold segmentation*) metodu. Šie metodai yra paprasti ir greiti, tačiau labai jautrūs nepageidaujamam triukšmui, ypač esant ryškiai saulės šviesai lauke.

Turint RGB vaizdą, kuriame yra matoma raudona lazerio linija, galima atkreipti dėmesį į tai, jog R kanalas turi žymiai didesnę vertę nei G ir B ties projektuotos linijos vieta, nes, kaip žinoma, raudonas lazeris skleidžia labai ryškią ir monochromatinę šviesą. Remiantis tuo yra pasiūlytas naujas segmentavimo metodas lazerio linijos aptikimui. Pavyzdžiui, turint RGB spalvotą vaizdą, kurio rezoliucija yra 640x480 pikselių ir kurio stulpelių ir eilučių skaičius yra atitinkamai *n* = 640 bei *m* = 480, lazerio linija gali būti apskaičiuojama pagal išraišką:

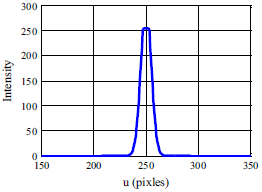
|  |  |
| --- | --- |
| , | (1.1) |

čia *Ii,j* – pikselio vertė *i* eilutėje ir *j* nuotraukos stulpelyje, o , ir yra atitinkamų R, G ir B kanalų pikselių vertės. Segmentavimo rezultatas pateiktas (žr. 1.1 pav.).



1.4 pav. Segmentavimo pavyzdys (a) originalus vaizdas (b) segmentavimo rezultatas [6]

Egzistuoja nemažai algoritmų skirtų lazerio linijos centro lokalizavimui didesniu nei vieno pikselio tikslumu. Pavyzdžiui, maksimalaus intensyvumo radimas per lazerio linijos plotį ir pikselių intensyvumo aproksimacija pagal Gauso skirstinį [8]. Deja, tokie metodai dažnai sukelia nepageidaujamą atsaką į apšviestumo pokyčius ir šešėlius. Verta atsižvelgti ir į tai, jog esant ryškiam lazerio linijos atspindžiui, įvyksta kameros persotinimas, dėl to pikselių intensyvumo reikšmės per linijos plotį yra iškraipytos Gauso skirstinio viršuje (žr. 1.5 pav.). Dėl to tai gali daryti neigiamą poveikį Gauso aproksimacijos skaičiavimui.

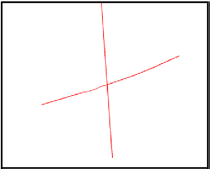


1.5 pav. Pikselių intensyvumas per lazerio linijos plotį [6]

Dėl Gauso skirstinio iškraipymų lokalizuojant lazerio linijos centrą yra naudojamas masės centro algoritmas. Pavyzdžiui, turint vertikalią liniją, masės centras *i* eilutėje yra apskaičiuojamas pasitelkiant mažą paieškos langą aplink intensyviausią linijos vietą. Masės centro stulpelio koordinatė *Mi(w)* eilutėje *i* yra apskaičiuojama pagal formulę:

|  |  |
| --- | --- |
| , , | (1.2) |

čia – maksimalus *i* eilutės intensyvumas *j* stulpelyje, o *w* yra paieškos lango dydis. Lazerio linijos plotis ir ryškumas įtakoja paieškos lango dydį *w*, o jo reikšmė yra 9 pikseliai, kuri buvo nustatyta empiriškai. Tada pagal (1.2) formulę galima nustatyti lazerio linijos centrą (žr. 1.6 pav.).



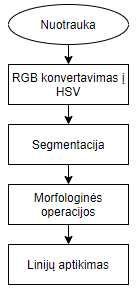
1.6 pav. Nustatyti lazerio linijų centrai [6]

Taigi, tokiu iš esmės paprastu slenkstinių ribų segmentavimo metodu galima aptikti lazerio liniją, kai fonas yra sąlyginai paprastas. Žinoma, kuriant šį algoritmą buvo atsižvelgta ir į jo pritaikymą – suvirinimo linijų aptikimas. Galima manyti, jog esant šiai konkrečiai aplikacijai turbūt dažniausiai bus turimas nesudėtingas fonas – paprastas, vienspalvis, neturintis daug briaunų. Esant šioms prielaidoms algoritmas geba aptikti lazerio linijas.

## Kliūčių aptikimas naudojant lazerio liniją

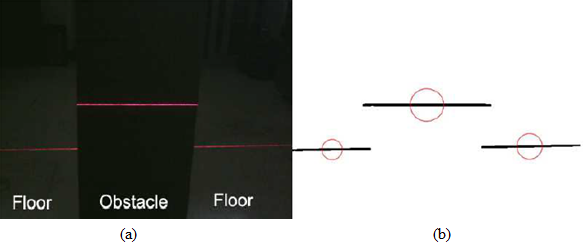
Automatizuoti judantys objektai, tokie kaip, pavyzdžiui, išmanieji neįgaliųjų vežimėliai privalo turėti atstumų nustatymo ir kliūčių aptikimo sistemas. Dėl to toliau yra pateikiamas lazerio linijos aptikimo metodas, kuris yra naudojamas būtent tokiose aplikacijose [9]. Naudojant ant kelio projektuojamą lazerio liniją ir CCD vaizdo kamerą trianguliacijos [10] principu galima nustatyti atstumą iki objekto, ant kurio yra projektuojama lazerio linija. Šiuo atveju bus nagrinėjamas tik lazerio aptikimo algoritmas nekreipiant dėmesio į atstumo nustatymo metodiką.

Pateiktas algoritmas (žr. 1.7 pav.) naudoja RGB spalvų schemos vaizdus, kurių rezoliucija yra 320x240 pikseliai. Priešlaikinio apdorojimo metu vaizdo RGB spalvų schema yra konvertuojama į HSV (angl. *Hue, Saturation, Value*). Tokia konversija pasitelkta dėl labai paprastos priežasties – dominantis aptikimo objektas, šiuo atveju lazerio linija, turi raudoną spalvą. Kaip žinoma, HSV spalvų schemoje H (angl. *Hue*) kanalo vertė nurodo būtent pikselio spalvą, dėl to tolimesniame vaizdo apdorojime spalvų segmentavimas tampa kiek paprastesnis, nei naudojant RGB spalvų schemą. Šią spalvų schemą naudojo M. Mesko‘as ir P. Chmelar‘as aptinkant raudono lazerio linijos projekciją[11, 12].



1.7 pav. Lazerio linijos aptikimo algoritmo struktūrinė schema

Lazerio linijos segmentavimas toliau yra vykdomas paprasčiausiai naudojant statines slenkstines ribas. Formuojamas binarinis vaizdas pagal nustatytus rėžius HSV spalvų schemoje. Apatinė riba yra (0, 70, 70), o viršutinė – (255, 255, 255). Visi pikseliai esantys vaizde, kurie patenka į šiuos rėžius yra pažymimi 1, visi likę – 0. Po to sudarytas binarinis vaizdas yra apdorojamas įvairiomis morfologinėmis operacijomis, tokiomis kaip uždarymas, skaitmeninis auginimas ir skaitmeninė erozija. Šių operacijų struktūriniai elementų dydžiai yra taip pat iš anksto numatyti. Uždarymo operacija yra kartojama 2 kartus su skirtingų dydžių elementais: 2x2 pikseliai ir 3x15 pikseliai. Visos kitos operacijos atliekamos su 2x2 pikselių dydžio struktūriniu elementu.



1.8 pav. Lazerio linijų aptikimo pavyzdys (a) originali nuotrauka (b) aptiktos linijos su pažymėtais centrais [9]

Tada yra pažymimi atskiri linijos segmentai. Tai yra atliekama pagal 8 pikselių kaimynystę (angl. *8-pixel connectivity*). Reiškia, jeigu binariniame vaizde esantis vienetas bet kokia kryptimi (vertikaliai, horizontaliai ar įstrižai) ribojasi su kitu vienetu, tada tas pikselis priklauso tam pačiam linijos segmentui. Po to yra skaičiuojamas kiekvieno segmento centras naudojant masės centrą pagal lygtis:

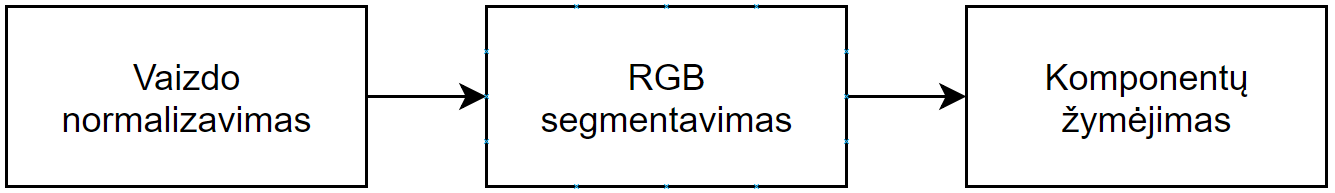
|  |  |
| --- | --- |
| , | (1.3) |
| , | (1.4) |
|  |  |

čia *centerx* ir*centery*yra x ir y centro koordinatės, n – pikselių, kurie priklauso vienam linijos segmentui skaičius, o *xk* ir *yk* – atitinkamai pikselių koordinatės. Galutinis algoritmo rezultatas pavaizduotas (žr. 1.8 pav.).

Taigi, šis algoritmas yra išties ganėtinai paprastas. Pagrindinis lazerio linijos aptikimo principas iš esmės yra paremtas tiesiog slenkstinių ribų segmentavimo metodu taip iš vaizdo išskiriant raudoną spalvą pagal pakankamai platų pikselių intensyvumo diapazoną (nuo (0, 70, 70) iki (255, 255, 255) pagal HSV spalvų schemą). Remiantis tuo, galima daryti prielaidą, jog bet koks raudonai ryškesnis objektas vaizde gali daryti neigiamą įtaką lazerio aptikimui, jei to objekto pikselių reikšmės atitiks nurodytą diapazoną.

## Lazerio linijos aptikimas atstumo nustatymui

Lazerio linijos projekcijos aptikimas šiandien labai plačiai naudojamas įvairiose robotikos bei elektroninių matavimo prietaisų srityse. Aptikimo sistemos kokybė yra priklausoma nuo lazerio linijos projekcijos ant įvairių objektų aptikimo tikslumo. Toliau pateiktas algoritmas yra paremtas RGB spalvų segmentavimu ir atskirų linijos komponentų žymėjimu (žr. 1.9 pav.). Būtent šio algoritmo pagrindu yra sukurta atstumų nustatymo sistema veikianti kartu su autonominiu robotu, kuris gali tirti nepažįstamas vietas ir kurti tų vietų žemėlapius [13]. Šiame poskyryje vėlgi yra aptariama tik lazerio linijos aptikimo metodika nekalbant apie jos pritaikymą konkrečiai aplikacijai.

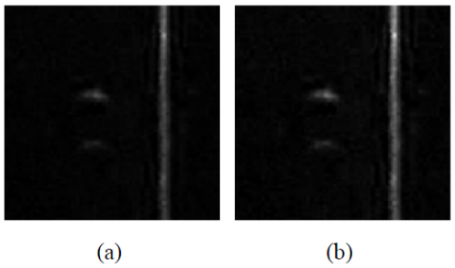


1.9 pav. Lazerio linijos aptikimo algoritmo struktūrinė schema

Priešlaikinio apdorojimo (angl. *pre-processing*) metu yra labai svarbu atlikti vaizdo normalizavimą. Šis paprastas būdas leidžia koreguoti vaizdo pikselių reikšmes taip, jog visas kadras įgyja didesnį bendrą kontrastą, taip išryškindamas silpnai matomus objektus (žr. 1.10 pav.). Šiuo atveju tokia pikselių intensyvumo reikšmių korekcija yra atliekama tik su raudonuoju kadro kanalu, nes norimos aptikti lazerio linijos spalva yra raudona. Normalizavimas yra apskaičiuojamas pagal formulę:

|  |  |
| --- | --- |
| , | (1.5) |

čia *INi(R)* – normalizuota raudonojo kanalo pikselio intensyvumo reikšmė, *i –* pikselio indeksas.



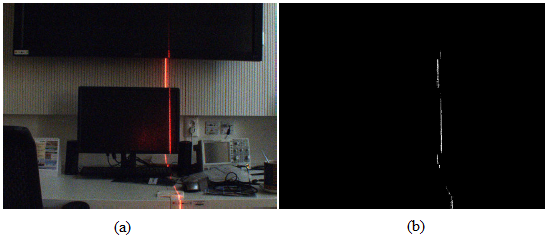
1.10 pav. Vaizdo normalizavimo pavyzdys (a) originali nuotrauka

(b) normalizuota nuotrauka [13]

Turint jau normalizuotą vaizdą toliau yra atliekamas spalvų segmentavimas. Tai yra pati svarbiausia šio algoritmo dalis, nes būtent segmentavimo būdu yra atrenkami visi pikseliai kadre, kurie priklauso lazerio linijai. Kadangi monochromatinė raudona lazerio linijos šviesa didžiausias pikselių intensyvumo reikšmes sukelia būtent raudonajame vaizdo kanale, logiška, jog būtent raudonasis kanalas atspindi pagrindines lazerio savybes. Likę du – žaliasis ir mėlynasis kanalai reikalingi baltos spalvos fone įvertinimui. Taigi, lazerio linijai priklausančių pikselių išskyrimas iš fono yra atliekamas nustačius konkrečias slenkstines ribas kiekvienam kanalui. Vienas pikselis laikomas priklausančiu raudonai lazerio linijai, jeigu atitinka išraišką:

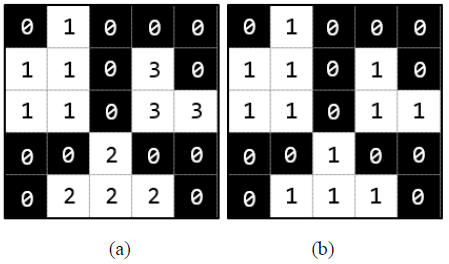
|  |  |
| --- | --- |
| , | (1.6) |

čia *IL* – išskirti iš fono lazerio linijos pikseliai. TR, TG ir TB yra atitinkamų spalvos kanalų slenkstinės ribos. Dar prieš pradedant spalvų segmentavimą visos trys slenkstinės ribos kiekvienam vaizdo kanalui turi būti parinktos rankiniu būdu atsižvelgiant į fono sąlygas. Tinkamai suderinus šiuos parametrus, yra gaunamas binarinis lazerio linijos vaizdas (žr. 1.11 pav.).



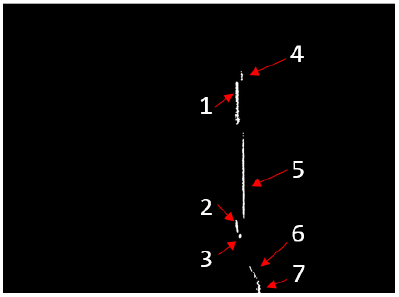
1.11 pav. Spalvų segmentavimo rezultatas (a) originalus vaizdas, (b) binarinis lazerio linijos vaizdas [13]

Tada turimas binarinis vaizdas (žr. 1.11 pav. (b)) yra apdorojamas morfologinėmis operacijomis, kurios pašalina smulkius lazerio linijos pertrūkimus ir užpildo skyles. Po to yra atliekamas skirtingų linijos segmentų žymėjimas pagal pikselių tarpusavio pozicijos sąryšį, kuris gali būti nusakomas dviem būdais (žr. 1.12 pav.).



1.12 pav. Sužymėti komponentai (a) keturių pikselių sąryšio būdas (b) aštuonių pikselių sąryšio būdas [13]

Keturių pikselių sąryšio būdas (angl. 4–*pixel connectivity*) (žr. 1.12 pav. (a)) sujungia atskirus pikselius į viena komponentą, jeigu šie liečiasi tik vertikalia ar horizontalia kryptimis, o aštuonių pikselių sąryšio būdas (angl. 8–*pixel connectivity*) (žr. 1.12 pav. (b))prie to pačio prideda ir pikselius, kurie ribojasi įstrižai. Šiam algoritmui yra naudojamas pirmasis būdas. Atskirų lazerio linijos komponentų žymėjimas yra naudingas tuo, jog galima aptikti ir nustatyti atstumus iki skirtingų objektų, ant kurių yra projektuojama lazerio linija.

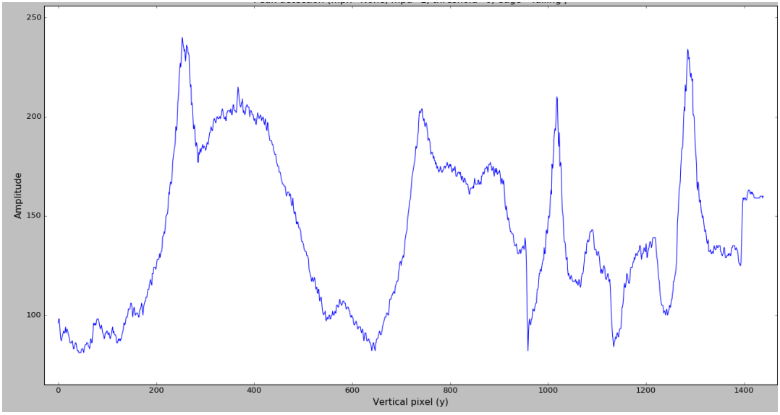


1.13 pav. Sužymėti lazerio linijos komponentai [13]

Taigi, šiame poskyryje pateiktas lazerio aptikimo algoritmas iš esmės yra paremtas tik spalvų segmentavimu nustatant slenkstines ribas kiekvienam kanalui RGB vaizde. Toks būdas reikalauja labai tikslaus slenkstinių ribų nustatymo, kuris turi būti koreguojamas atsižvelgiant į tai, kaip ryškiai yra atspindima lazerio šviesa. Taip pat turi būti garantuota, jog turimam vaizde nebus daugiau pikselių, apart lazerio linijos, kurie atitinka (1.6) išraišką.

## 3D lazerio linijos skeneris

Norint aptikti lazerio liniją yra skaičiuojami pikselių intensyvumo lygiai ir ieškomi jų pikai, aukščiausios reikšmės. Kintanti aplinkos apšvieta gali sukelti aptinkant lazerio liniją, nes nepageidaujama šviesa taip pat gali sukelti pikselių intensyvumo pikus, kurie nėra lazerio linijos dalis.



1.14 pav. Vieno nuotraukos stulpelio R kanalo pikselių intensyvumo kitimo grafikas [7]

Siekiant padidinti lazerio linijos aptikimo patikimumą ir sumažinti aplinkos keliamus pikselių intensyvumo pikus, galim pasinaudoti faktu, jog raudona lazerio šviesa ryškiausiai bus atspindima raudoname kanale. Tokiu atveju, siekiant aptikti liniją, algoritmui galima naudoti tik vieno kanalo reikšmes.

### Baltos šviesos filtravimas

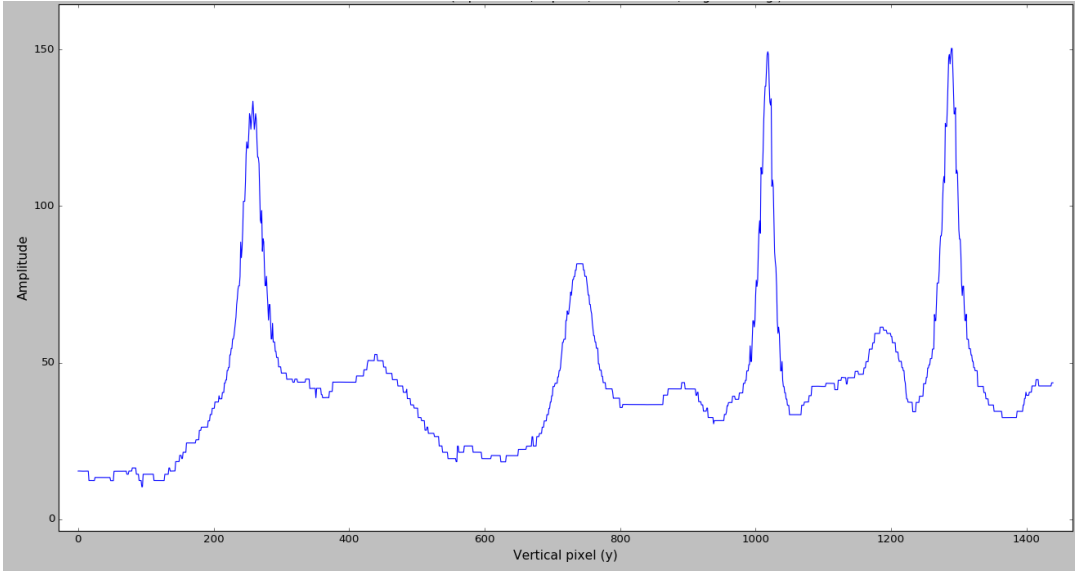
Kadangi balta šviesa apytiksliai vienodai atsispindi visuose trijuose (R, G, B) kanaluose, ji taip pat gali sukelti nepageidaujamus pikselių intensyvumo pikus raudoname kanale. Siekiant išfiltruoti baltą šviesą iš kadro, yra naudojamas žalio ir mėlyno kanalo vidurkis:

|  |  |
| --- | --- |
|  | (1.3) |

*IR, IG ir IB –* atitinkamų kanalų pikselių intensyvumo reikšmės.

Iš raudono kanalo intensyvumo reikšmės yra atimamas mėlyno ir žalio kanalų intensyvumų vidurkis. Tokiu būdu šviesa, kuri sklinda iš nepageidaujamų aplinkos šviesos šaltinių yra išfiltruojama. Pasėkoje to, yra gaunami ryškesni intensyvumų pikai, kurie įtakoja lazerio linijos aptikimo patikimumą.

Grafike (1.9 pav.) yra pateiktas to pačio stulpelio nuotraukos pikselių intensyvumo kitimas iš raudono kanalo atėmus žalio ir mėlyno kanalo vidurkį.



1.15 pav. Vieno nuotraukos stulpelio pikselių intensyvumo kitimo grafikas iš raudono kanalo reikšmių atėmus mėlyno ir žalio kanalų reikšmių vidurkius [7]

Iš (1.8 pav.) ir (1.9 pav.) pateiktų grafikų, galima matyti, jog dalis buvusių pikų yra išfiltruojama, o likę pikai tampa labiau ryškūs viso grafiko atžvilgiu.

### Lazerio linijos taškų radimas

Ieškant lazerio linijos taškų, kiekvienas nuotraukos pikselių stulpelis yra skanuojamas paeiliui. Siekiant surasti taškus, turi būti randami pikselių intensyvumo kitimo pikai. Yra skirtingų būdų, norint aptikti pikus, tačiau algoritmas turi būti pakankamai optimalus, nes tai smarkiai įtakos aptikimo algoritmo veikimo trukmę.

Šiam algoritmui anksčiau buvo taikomas statinės slenkstinės ribos[8] metodas. Kadangi skanavimo metu aplinkos apšvieta yra kintama, šis metodas nėra tinkamas patikimam pikų radimo algoritmui įgyvendinti.

Norint pakelti pikų radimo patikimumą, buvo pritaikytas sudėtinis vidurkis (angl. running average) [9], kurio dėka, kintantys fono pikselių intensyvumo lygiai yra kompensuojami. Kiekviename skanuojamo nuotraukos stulpelio taške yra skaičiuojamas sudėtinis vidurkis. Norint aptikti pikus, yra nustatoma slenkstinė riba (angl. threshold), prie kurios taip pat yra pridedama kita statinė slenkstinė riba. Tada visos reikšmės, kurios yra aukščiau nustatytos ribos yra laikomos lazerio linijos taškais.

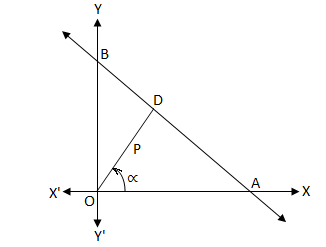
## „Hough“ transformacija

„Hough“ transformacija yra metodas, kuris gali būti naudojamas norint išskirti tam tikros formos požymius vaizde [18]. Kadangi yra reikalavimas, kad norimos savybės būtų nurodytos tam tikra parametrų forma, klasikinė „Hough“ transformacija dažniausiai naudojama nustatant įprastas kreives, tokias kaip linijos, apskritimai, elipsės.

Šiuo atveju mes kalbėsime apie „Hough“ transformacijos naudojimą linijos aptikimui. Pagrindinis principas grindžiamas normaliąja linijos forma:

|  |  |
| --- | --- |
|  | (2.1.7) |

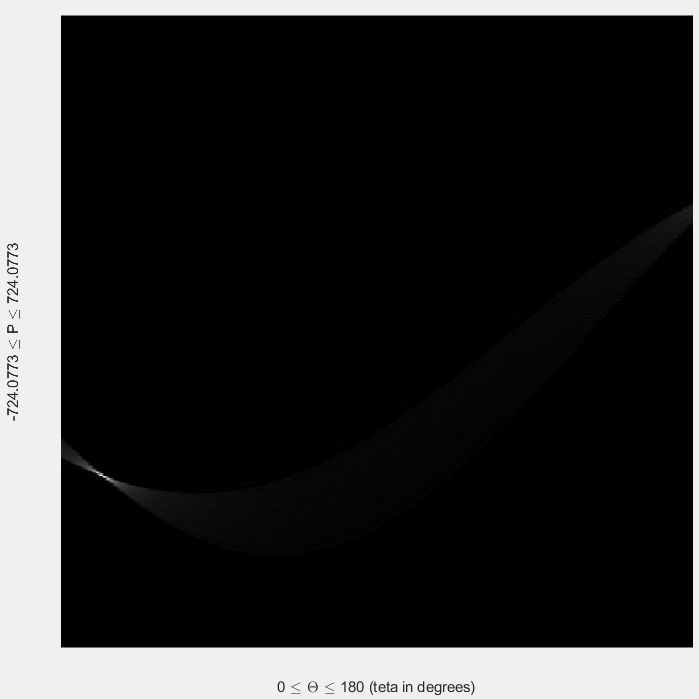
,kur *p* yra atraminės linijos ilgis, o yra kampas tarp atraminės linijos ir pradinės ašies. Tai galima pamatyti (2.19 pav.). Atramos linija visada yra statmena linijai, kurią norime apibrėžti.



2.19 pav. Normalioji linijos forma

Įvestis į šį algoritmą turi būti dvejetainis vaizdas. Jį nesunkiai galima gauti išfiltruotai nuotraukai pritaikius paprastos slenkstinės ribos metodą. Tada algoritmas iš dvejetainio vaizdo su taškais piešia taškus „Hough“ erdvėje.

„Hough“ erdvė yra matematinė erdvė, kurios ašys yra *p* ir , atitinkamai pagal (2.10) išraišką. Tai yra du parametrai, apibrėžiantys liniją skirtinga forma, kaip matome (2.20 pav.). Šioje matematinėje erdvėje linija yra apibrėžiama kaip taškas. Apskaičiuotos „Hough“ erdvės pavyzdys yra pateiktas (2.20 pav.).



2.20 pav. „Hough“ matematinė erdvė

Linija yra aptinkama balsavimo metodu. Kuo daugiau taškų sudaro vieną liniją nuotraukoje, tuo ryškesnis taškas bus „Hough“ erdvėje, kuris apibūdina tą liniją. Matoma (2.20 pav.) vienas ryškus lokalus maksimumas, kuris apibūdina liniją. Pasirinkus norimą balsų skaičių, tai yra taškų skaičius, kuris pasako kiek mažiausiai taškų turi sudaryti liniją, jog tai būtų laikoma linija yra sprendžiama ar tas lokalus maksimumas gali būti nuotraukoje aptikta linija.

Pagal ašis galima nustatyti du to taško parametrus: *p* ir . Turint šiuos parametrus iš esmės turime ir linijos lygtį, kurią nubrėžti nuotraukoje.

## Furjė Transformacija

Furjė transformacija yra svarbi vaizdo apdorojimo priemonė, naudojama vaizdui suskaidyti į jo sinuso ir kosinuso komponentus. Transformacijos išvestis atspindi vaizdą Furjė arba dažnio srityje, o įvesties vaizdas yra erdvinės srities ekvivalentas. Furjė domeno paveikslėlyje kiekvienas taškas žymi tam tikrą erdvinio domeno vaizdo dažnį. Paprasčiau tariant, funkcijos ar signalo Furjė transformacija, šiuo atveju vaizdo, nuotraukos, yra kompleksiniais skaičiais įvertinama dažnio funkcija, kurios dydis ar kitaip tariant, absoliuti reikšmė, parodo to specifinio dažnio kiekį, esantį pradinėje funkcijoje (vaizde erdviniame domene). Šios operacijos pavyzdys pateiktas žemiau (žr. 1.1 pav.).



2.1 pav. Furjė transformacijos pavyzdys **a)** originali nuotrauka, **b)** nuotraukos dažnio spektras

Furjė transformacija naudojama įvairiose srityse, tokiose kaip vaizdo analizė, vaizdo filtravimas, vaizdo rekonstravimas ir vaizdo glaudinimas. Šiuo atveju ji yra naudojama filtruojant žemus dažnius iš nuotraukos. Tai reiškia, kad mes stengiamės pašalinti taškus, kurie nepriklauso linijai, ir išlaikyti kuo daugiau taškų, kurie priklauso linijai, kurią bandome aptikti.

Dažnio spektras (2.1 pav.) rodo, kad mūsų paveikslėlyje yra aukštų dažnių. Kuo pikselio pozicija yra tolimesnė nuo centro pikselio spektre, tuo aukštesnį dažnį tas pikselis atvaizduoja. Be to, įstriža balta linija, matoma spektre, rodo lazerio liniją erdvės domene. Ji turi vieną labai svarbų bruožą - jo kampas yra statmenas tai linijai erdvinėje srityje.

Šioje situacijoje mes kalbame apie diskrečius signalus (skaitmeninius vaizdus), todėl tolesnė diskusija apsiriboja diskrečiąja Furjė transformacija (DFT). DFT yra diskretinė arba imtinė Furjė transformacijos forma. Joje nėra visų įmanomų paveikslėlyje aptiktų dažnių, bet tik tam tikrų dažnių rinkinys, kurio pakanka apibūdinti erdvinės srities vaizdą. Svarbu paminėti, kad dažnių skaičius yra lygiai toks pat kaip ir pikselių skaičius originaliame paveikslėlyje, todėl vaizdų dydis tiek erdviniame, tiek dažnių srityje yra vienodas. N dydžio kvadratiniam atvaizdui dvimatis DFT apskaičiuojamas taip:

|  |  |  |
| --- | --- | --- |
|  |  | (2.1.8) |

,kur f (m, n) yra vaizdas erdvinėje srityje, atitinkantis kiekvieną tašką F (k, l) Furjė erdvėje. Paprasčiau tariant, šią lygtį galima apibūdinti taip: kiekvienas Furjė erdvės taškas gaunamas padauginus vaizdą erdvinėje srityje su duota bazine funkcija ir susumavus rezultatą.

Panašiu būdu Furjė vaizdą galima atgal konvertuoti į erdvinį domeną. Atvirkščioji diskrečioji Furjė transformacija yra aprašoma taip:

|  |  |  |
| --- | --- | --- |
|  |  | (2.1.9) |

,kur f (m, n) ir F (k, l) turi tas pačias reikšmes, kaip aprašyta (2.1) lygtyje.

Remiantis (2.1) ir (2.2) lygtimis, akivaizdu, kad šios išraiškos, skaičiavimo požiūriu, yra tikrai brangios, jų skaičiavimas reikalauja daug resursų ir šiais laikais jos daugiau nenaudojamos. Toliau yra pateikiamas greitesnis, ir optimalesnis algoritmas, kuris išsprendžia šią našumo problemą.

### „Cooley-Tukey“ greitosios Furjė transformacijos algoritmas

„Cooley-Tukey“ algoritmas yra labiausiai paplitęs greitosios Furjė transformacijos algoritmas. Greitoji Furjė transformacija (FFT) - yra greitesnioji metodika, skirta apskaičiuoti diskretinę Furjė transformaciją (DFT), kuri savo ruožtu yra tolydžiosios Furjė transformacijos diskretinė versija, iš kurios, iš tikrųjų, yra kilusios visos kitos jos versijos.

„Cooley-Tukey“ algoritmas yra skaldyk ir valdyk [14] tipo algoritmas. Jis padalija problemą į daug mažesnių problemų, kurios skaičiavimo požiūriu yra daug pigesnės. Taigi, ką jis daro, jis suskaido diskretinę Furjė transformaciją į mažesnes DFT transformacijas ir tik tada apskaičiuoja pilną DFT.

„Radix-2“ yra paprasčiausia „Cooley-Tukey“ algoritmo forma. Jis padalija N dydžio DFT į du vienodai mažesnius N / 2 dydžio DFT. Iš čia kilęs pavadinimas „radix-2“.

Diskretinę Furjė transformaciją apibūdina lygtis:

|  |  |  |
| --- | --- | --- |
|  |  | (2.1.10) |

,kur k yra sveikas skaičius nuo *0* iki *N - 1*.

Kitas žingsnis - susumuoti lyginių skaičių *n = 2m* ir nelyginių *n = 2m + 1* indeksų vertės:

|  |  |
| --- | --- |
|  | (2.1.11) |

Dabar, jei mes sutrauksime bendrąjį daugiklį, (2.4) lygtį galima perrašyti taip:

|  |  |
| --- | --- |
|  | (2.1.12) |

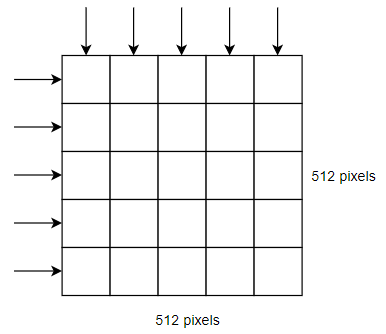
,kur *Ek* yra lygios indeksavimo vertės, o *Ok* - nelyginės indeksuotos vertės.

O dabar galime išreikšti DFT perrašydami *Xk* kaip:

|  |  |
| --- | --- |
|  | (2.1.13) |

Pirmiausia algoritmas apskaičiuoja lygių ir nelyginių indeksuotų verčių diskrečiąją Furjė transformaciją. Tai yra svarbiausia dalis, kur algoritmas įgauna savo greitį. Signalas rekursyviai padalijamas į dvi mažesnes DFT, kol signalo tampa nebeįmanoma suskaldyti. Tokiu būdu yra išvengiama daug papildomų skaičiavimų, kadangi mažesnių transformacijų rezultatai yra naudojami daug kartų apskaičiuoti visą transformaciją.

Pagal (2.6) formulę, pateiktas algoritmas tinka tik vienmačiam signalui skaičiuoti. Nepaisant to, tai nėra sunku atlikti naudojant dvimatį signalą, šiuo atveju – vaizdą, nuotrauką. Vaizdo dažnio spektras apskaičiuojamas pirmiausia jį atliekant viena kryptimi, o vėliau kita kryptimi, kaip pavaizduota (2.2 pav.).



pav. 2.2 Greitosios Furjė Transformacijos taikymas su nuotrauka

Vieną pikselių eilę ar stulpelį galima įsivaizduoti kaip vienmatį signalą, kurio diskrečiąją Furjė transformaciją galima suskaičiuoti naudojant „Cooley-Tukey“ algoritmą. Iš pradžių tai daroma su kiekviena eilute, po to su kiekviena stulpeliu, o rezultatas susumuojamas į vieną dažnio spektrą. Vizualizuotas dažnių spektras pavaizduotas (2.1 pav.). Žinoma, dėl tokio algoritmo pobūdžio, originali jo realizacija turi ribojimą – signalo dydis privalo būti skaičiaus 2 kartotinis. Modernios bibliotekos turi savų realizacijų, kuriose ši problema yra išspręsta ir šiame darbe į tai nekreipiame dėmesio.

## Aukšto dažnio filtrai

Aukšto dažnio filtras yra filtras, pro jį leidžiantis praeiti tik aukštiems dažniams. Aukšti dažniai nuotraukoje yra tos vietos, kur yra labai didelis skirtumas tarp gretimų pikselių intensyvumų reikšmių [15]. Pavyzdžiui, įvairių objektų kraštai, balto popieriaus lapo, kuris guli ant juodo stalo, kraštas.

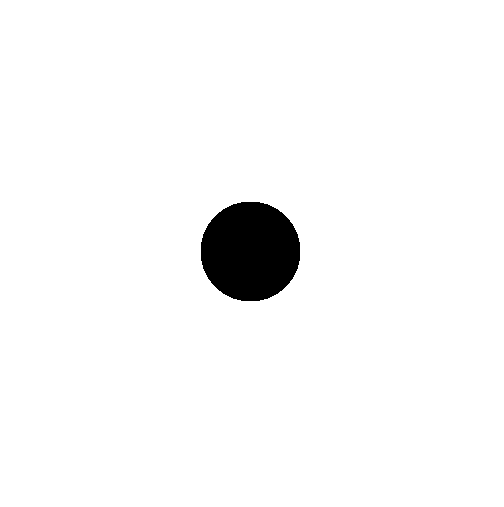
Filtravimo dažnio srityje priežastis dažniausiai yra ta, kad žymiai greičiau yra atlikti dvimatę Furjė transformaciją ir jos gautą rezultatą padauginti iš turimo filtro, nei atlikti vaizdo, erdvinio domeno konvoliuciją. Tai ypač svarbu didėjant filtro dydžiui [16]. Viena iš daugelio galimų priežasčių, kodėl filtruoti dažnių sritį yra gera idėja, yra ta, kad ji skaičiavimo požiūriu yra pigesnė. Šiame skyriuje kalbėsime apie įvairius aukšto dažnio filtrus, kurie gali būti naudojami dažnio spektrui filtruoti.

### Idealusis filtras

Idealiojo aukšto dažnio filtro idėja yra labai paprasta. Jis turi tik vieną parametrą *D0*, kuris yra filtro spindulys, ir naudojama kaip paprasta slenkstinė riba nuotraukos dažnių spektre. Šį filtrą galima aprašyti tokia išraiška:

|  |  |
| --- | --- |
|  | (2.1.14) |

, kur *D(u,v)* yra atstumas tarp esamo pikselio ir pikselio, kuris yra dažnio spektro centre.



2.3 pav. Idealusis aukšto dažnio filtras

Iš esmės, įgyvendinus (2.7) išraišką ir darant iteracijas su kiekvienu nuotraukos pikseliu, mes gauname dvejetainį vaizdą (2.3 pav.), kuris veikia kaip filtras. Tada galime jį lengvai padauginti iš savo dažnių spektro. Juodas apskritimas, esantis centre, turi reikšmes lygias 0, visi kiti pikseliai – 1.

Kadangi dažnių spektras yra pertvarkomas taip, jog centre turima žemi dažniai, tai padauginus šį filtrą iš viso dažnių spektro, yra išmetami visi žemi dažniai iš nuotraukos.

### „Butterworth“ filtras

„Butterworth“ aukšto dažnio filtras yra lankstesnis ir dinamiškesnis filtras. Į jo funkciją įvedamas naujas parametras *n*, tai yra filtro eilė [17]. Šio parametro pagalba galima valdyti filtro ryškumą, tai yra, kiek filtro kraštai bus išlieti (pav. 2.4). Filtras yra aprašomas pagal funkciją:

|  |  |
| --- | --- |
|  | (2.1.15) |

,kur *D(u,v)* yra atstumas tarp pikselio esančio centre ir dabartinio pikselio dažnių spektre, o parametras *n* nusako filtro eilę.



2.4 pav. „Butterworth“ aukšto dažnio filtras

Aiškiai matoma, kad „Butterworth“ filtras, priešingai nei idealusis filtras, neturi aštrių briaunų. Šis filtras taip pat filtruoja pereinamuosius taškus, kai keičiasi intensyvumas. Todėl išfiltruotas vaizdas erdviniame domene turės ryškesnius kraštus.

### Gauso filtras

Gauso aukšto dažnio filtras yra tarsi tolygesnė „Butterworth“ filtro versija. Atskirtis tarp praleistų ir išfiltruotų dažnių yra labai neryški, dėl to yra gaunamos labai ryškūs kraštai ir briaunos erdviniame domene po filtravimo, nes daugelis pereinamųjų pikselių yra tiesiog išfiltruojama:

|  |  |
| --- | --- |
|  | (2.1.16) |

,kur *D(u,v)* yra atstumas tarp pikselio esančio centre ir dabartinio pikselio dažnių spektre, o parametras *D0* nusako filtro spindulį.



2.5 pav. Gauso aukšto dažnio filtras

# Lazerio linijos aptikimo metodologija

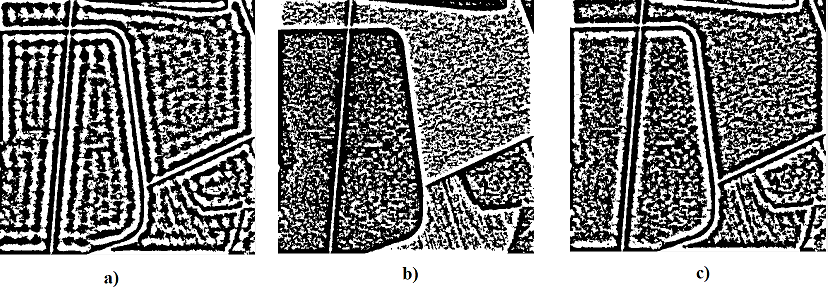
Šioje dalyje pateikiami metodų aprašymai ir įžvalgos, naudojamos aptikti raudoną lazerio liniją vaizde. Lazerio linijos prietaisas skleidžia koherentinę koncentruotą šviesą. Kadangi skleidžiama šviesa yra koncentruota, ji sukuria labai aukštus dažnius vaizde, ypač kai linija yra atspindima nuo šviesių paviršių. Tai yra pagrindinė savybė, kuri yra naudojama aptikti liniją vaizde. Norėdami konvertuoti vaizdą iš erdvinės srities į dažnio sritį, yra naudojama Furjė transformacija [12]. Buvo naudojamas Cooley-Tukey greitosios Furjė transformacijos algoritmas [13], siekiant išvengti didelių skaičiavimo sąnaudų. Tada apskaičiuotas vaizdo dažnio spektras buvo filtruojamas naudojant aukšto dažnio filtrą. Buvo analizuojami trys skirtingi tų aukšto dažnio filtrų tipai:

* Idealusis filtras.
* Butterworth filtras.
* Gauso filtras.

Ypatybių aptikimo metodas, vadinamas Hough transformacija, buvo naudojamas aptikti visą liniją. Yra daugybė skirtingų implementacijų ir modifikacijų, tačiau čia analizuojama tik pati paprasčiausia linijos aptikimo implementacija.

### Aukšto dažnio filtrų tyrimas

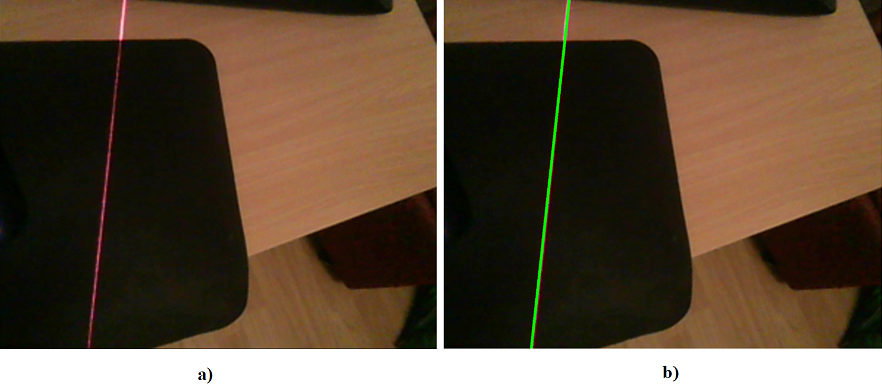
Lazerio linijos aptikimo kokybė smarkiai priklauso nuo prieš tai atlikto dažnių spektro filtravimo kokybės. Pagal dabartinę sudaryta metodologiją, dažnių filtravimas priklauso nuo aukšto dažnio filtro tipo ir dydžio.



2.6 pav. Dažnio spektro filtravimo rezultatai su skirtingais filtrais a) idealusis filtras b) Gauso filtras c) „Butterworth“ filtras

Pagal (2.6 pav.) pateiktus rezultatus vizualiai matome, jog „Butterworth“ ir Gauso filtrai grąžina sąlyginai panašius rezultatus, išsiskiria tik idealusis filtras. Filtruojant dažnių spektrą šiuo filtru matome, jog rezultate atsiranda tam tikrų „bangavimų“, nėra taip ryškiai išskiriami vaizdo kontūrai. Akivaizdu, jog idealusis filtras yra mažiausiai naudingas šiuo atveju, yra paliekama daug nereikalingo triukšmo. Siekiant išsiaiškinti kiekvieno filtro poveikį linijos aptikimo kokybei buvo atliktas bandymas.

Bandymui buvo parinktos trys eksperimentinės nuotraukos. Vienos iš jų pavyzdys pateiktas (2.7 pav.) nuotraukoje. Čia matome originalią nuotrauką su pakankamai ryškia lazerio linija (a) bei pažymėtą nuotrauką (b). Pastarojoje lazerio linija yra pažymėta visiškai žalia spalva (RGB reikšmės lygios 0, 255, 0). Tokią pažymėtą liniją yra nesunku aptikti, nei vienoje eksperimentinėje nuotraukoje pikselių, kurie turėtų lygiai tokias pat reikšmes nėra.



2.7 pav. Eksperimentinės nuotraukos pavyzdys a) originali nuotrauka b) pažymėta nuotrauka

Visi pikseliai, kurie yra pažymėti visiškai žalia spalva yra laikomi lazerio linijai priklausančiai pikseliais, visi kiti – nereikalingas triukšmas. Remdamiesi tuo, atliekame bandymus su kiekviena iš trijų nuotraukų. Kiekvienos nuotraukos dažnių spektrą filtruojame su trimis skirtingais aukštų dažnių filtrais. Ši iteracija kartojama tris kartus vis didinant filtro dydį (spindulį *D*). „Butterworth“ filtrui nustatome statinį eilės numerį *n*=3.

Po kiekvienos iteracijos yra skaičiuojama, kiek procentais kiekvienas filtras išfiltravo mums naudingos duomenis, tai yra pikselius, kurie priklauso lazerio linijai bei paliktą triukšmą – pikselius, kurie nepriklauso lazerio linijai.

Lentelė 2.1 Filtravimo rezultatai su pirma eksperimentine nuotrauka *D*=20, *n*=3

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Idealusis filtras** | **Gauso filtras** | **„Butterworth“ filtras** |
| Išfiltruoti linijos pikseliai (%) | 1.457 | 1.457 | 1.249 |
| Paliktas triukšmas (pikseliai) | 117878 | 108398 | 108724 |

Lentelė 2.2 Filtravimo rezultatai su pirma eksperimentine nuotrauka *D*=50, *n*=3

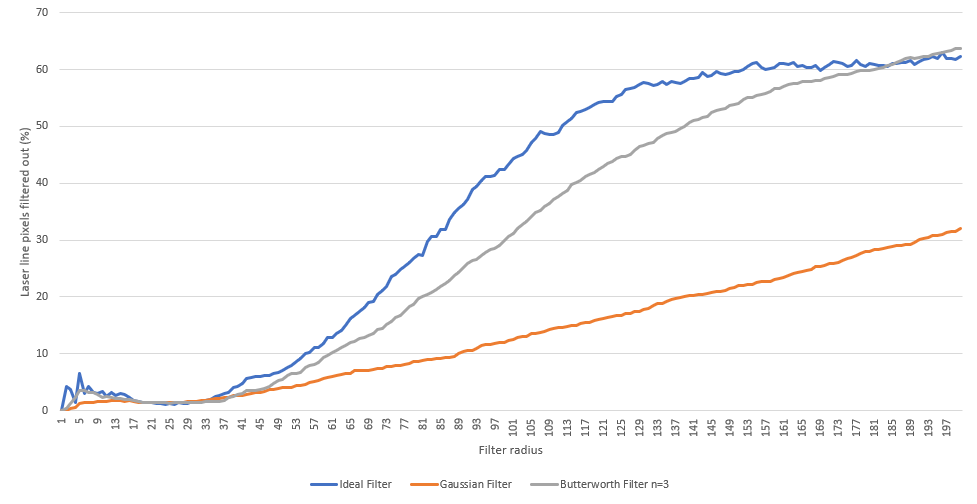
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Idealusis filtras** | **Gauso filtras** | **„Butterworth“ filtras** |
| Išfiltruoti linijos pikseliai (%) | 8.899 | 2.715 | 6.335 |
| Paliktas triukšmas (pikseliai) | 104359 | 83165 | 93803 |

Lentelė 2.3 Filtravimo rezultatai su pirma eksperimentine nuotrauka *D*=80, *n*=3

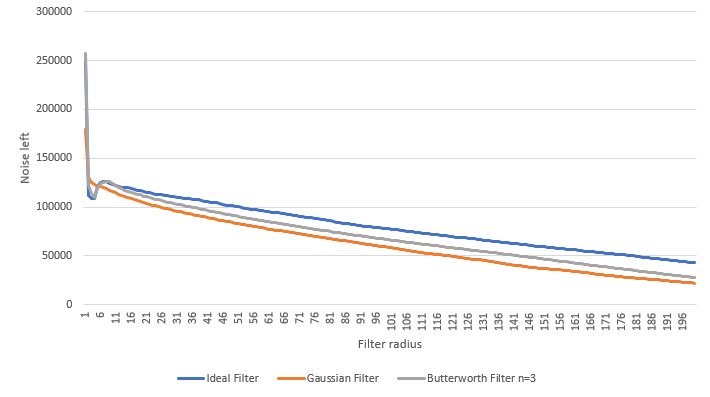
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Idealusis filtras** | **Gauso filtras** | **„Butterworth“ filtras** |
| Išfiltruoti linijos pikseliai (%) | 7.113 | 4.149 | 5.504 |
| Paliktas triukšmas (pikseliai) | 100566 | 83639 | 91032 |

Pagal (2.1, 2.2, 2.3 lentelės) pateiktus duomenis matome, jog idealiojo filtro rezultatas yra prasčiausias. Visais atvejais šis filtras išfiltravo daugiausia mums reikalingų duomenų ir paliko didžiausią triukšmo kiekį. „Butterworth“ filtras yra kiek sunkiau palyginamas su kitais dviem, kadangi šis turi papildomą parametrą – eilės numerį *n*.

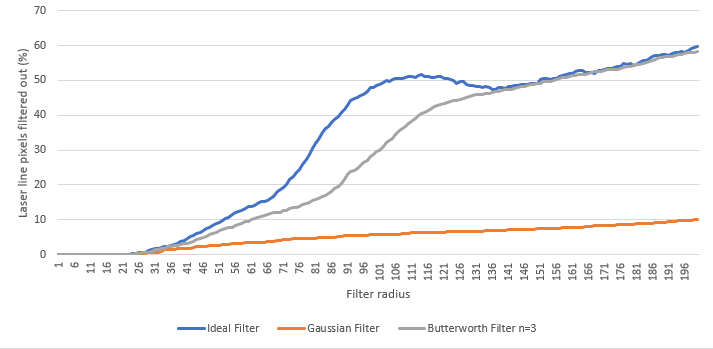
Lentelėse pateikti duomenys apibūdina kiekvieno filtro įtaką tik prie tam tikrų filtro dydžio reikšmių. Siekiant detaliau nustatyti kaip filtro dydis įtakoja filtravimo kokybę, atlikome sekantį bandymą. Jo metu buvo paimtos tos pačios trys eksperimentinės nuotraukos, tačiau dabar atliekame daugiau iteracijų. Kiekvienos iteracijos metu filtro dydį *D* keičiame nuo 1 iki 200 ir skaičiuojame tuos pačius parametrus.



2.8 pav. Išfiltruoti lazerio linijos pikseliai procentais (pirma nuotrauka)



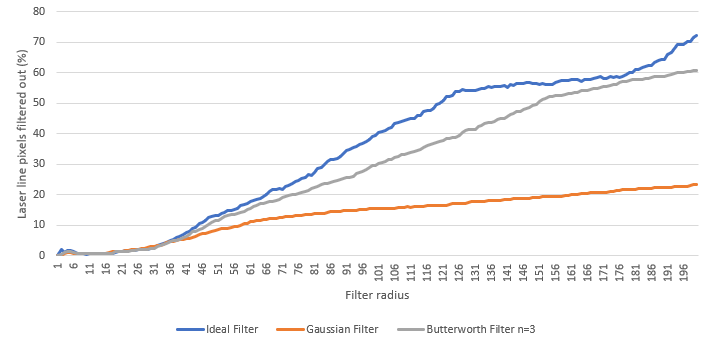
2.9 pav. Paliktas triukšmas pikseliais (pirma nuotrauka)



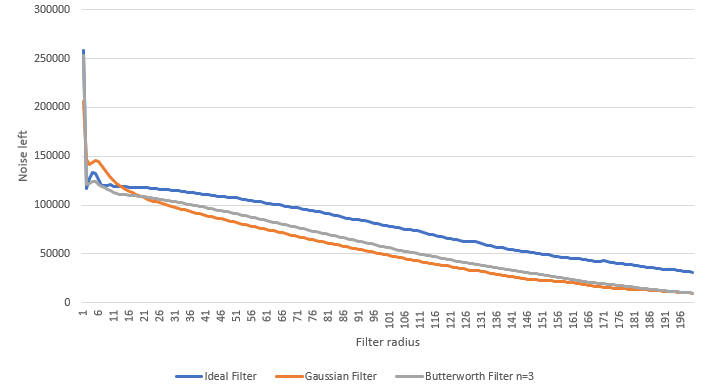
2.10 pav. Išfiltruoti lazerio linijos pikseliai procentais (antra nuotrauka)



2.11 pav. Paliktas triukšmas pikseliais (antra nuotrauka)



2.12 pav. Išfiltruoti lazerio linijos pikseliai procentais (trečia nuotrauka)



2.13 pav. Paliktas triukšmas pikseliais (trečia nuotrauka)

Pagal (2.8 pav. – 2.13 pav.) matome, jog ties kiekvienu bandymu Gauso filtras išfiltruoja mažiausiai lazerio linijos taškų ir palieka mažiausiai triukšmo, o idealusis filtras atvirkščiai – išfiltruojama daugiausia lazerio linijos taškų ir paliekama daugiausia triukšmo. Toliau (2.4 lentelė) pateikta visų bandymų rezultatų vidurkiai.

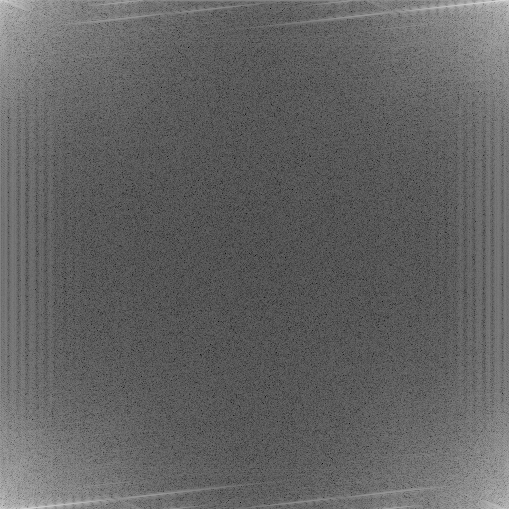
Lentelė 2.4 Visų tyrimų rezultatų vidurkis

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Idealusis filtras** | **Gauso filtras** | **„Butterworth“ filtras** |
| Išfiltruoti linijos pikseliai (%) | 34.773 | 10.709 | 29.698 |
| Paliktas triukšmas (pikseliai) | 77448 | 58144 | 64146 |

Remiantis gautais rezultatais galima teigti, jog iš bandytų trijų filtrų tinkamiausias lazerio linijos aptikimo uždaviniui spręsti yra Gauso filtras.

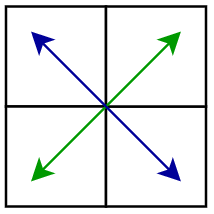
### Dažnių spektro filtravimas

Atlikus Furjė transformaciją yra gaunama nuotrauka, konvertuota į dažnių domeną. Originalioje formoje, ši nuotrauka žemus dažnius atspindi kampuose. Tai reiškia, jog nuotraukos kampuose esančios reikšmės, pikseliai apibūdina žemus dažnius (pav. 2.14).



2.14 pav. Pradinė dažnio spektro forma

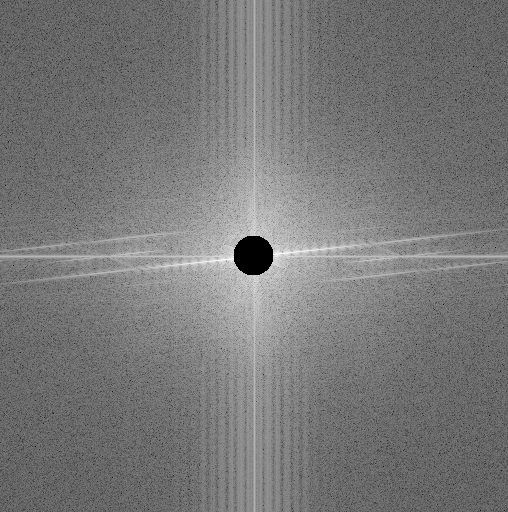
Filtruoti tokį vaizdą darosi sudėtinga, kadangi žemi dažniai yra išmėtyti skirtingose nuotraukos vietose. Tam, kad filtravimas būtų paprastesnis ir greitesnis, yra įstrižai sukeičiami nuotraukos kvadrantai (pav. 2.15).



2.15 pav. Keičiami spektro kvadrantai

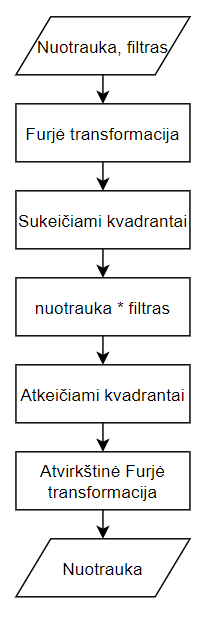
Tai yra daroma tam, jog visi žemi dažniai būtų sutelkti vienoje spektro vietoje – centre. Po šios operacijos, reikšmės esančios toliau nuo centro, turi aukštesnį dažnį, o arčiau esančios – žemesnį. Turint tokį vaizdą, galima jau galima taikyti šiame skyriuje aprašytus aukšto dažnio filtrus.

Dažnių spektro filtravimas yra ganėtinai paprasta procedūra, apie tai jau buvo užsiminta šiame skyriuje. Paprasčiausiai, yra paimamas dažnių spektras, kaip matrica, ir ji yra padauginama iš turimo filtro. Šios procedūros pavyzdys yra pateiktas (pav. 2.16).



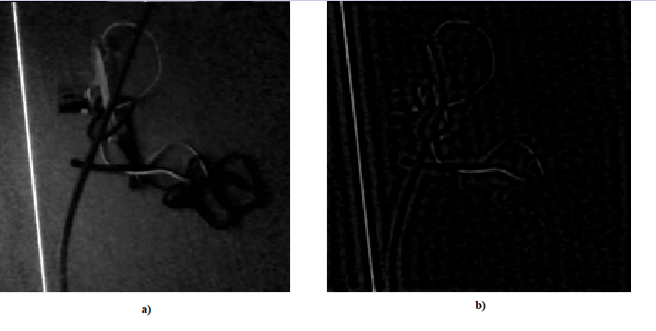
2.16 pav. Dažnių spektras su aukšto dažnio filtru

Kadangi, šiuo atveju, filtro viduryje reikšmės yra lygios nuliui, arba artimos nuliui, atlikus daugybos veiksmą, žemi dažnių reikšmės taip pat yra paverčiamos į artimas nuliui. Visas filtravimo procesas yra pavaizduotas (pav. 2.17).



2.17 pav. Dažnių spektro filtravimo procesas

Kaip įvestį, yra turima vieno kanalo nuotrauka bei iš anksto suformuotas aukšto dažnio filtras. Nuotraukos ir filtro dydžiai turi būti vienodi. Furjė transformaciją turimą nuotrauką konvertuoja į dažnių spektrą. Prieš filtravimą spektro kvadrantai privalo būti sukeisti įstrižai, tada turimas spektras yra sudauginamas su filtru. Tada spektras turi būti grąžinamas į pradinę formą, kvadrantai yra atkeičiami atgal į savo pradines padėtis, o atvirkštinės Furjė transformacijos pagalba yra gaunama nuotrauka erdviniame domene, kurioje yra išfiltruoti žemi dažniai.



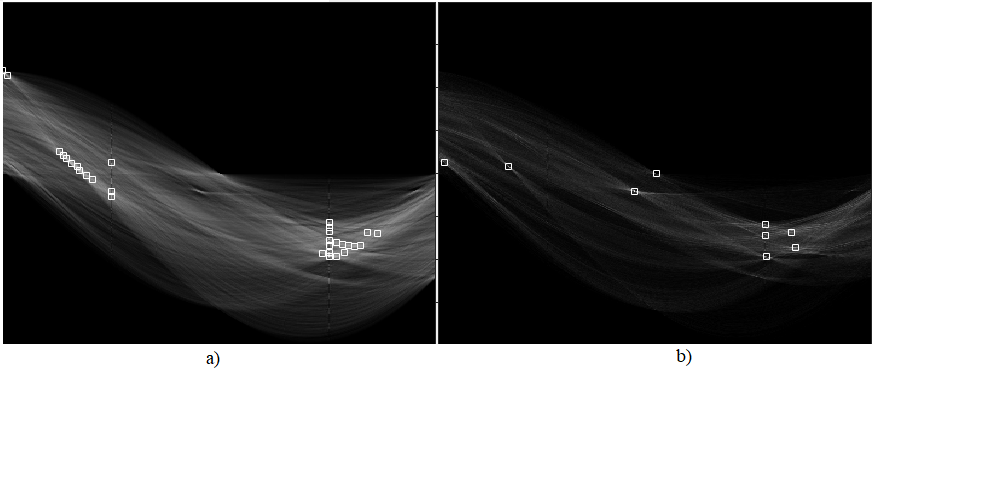
2.18 pav. Aukštų dažnių filtravimo pavyzdys a) vaizdas prieš filtravimą b) vaizdas po filtravimo

Pateiktame pavyzdyje (pav. 2.18 b) yra matoma nuotrauka su išfiltruotais žemais dažniais. Kadangi linija, esanti kairėje, yra ryšiai atspindima, ji lygiai taip pat ryškiai matoma ir po filtravimo proceso. Šiam pavyzdžiui buvo naudojamas raudonas nuotraukos kanalas.

## Filtro dydžio parinkimas

Atsižvelgiant į 2.2.4 skyrelyje pateiktus tyrimo rezultatus (apibendrinti rezultatai pateikti 2.4 lentelėje) tikslingiausia yra naudoti būtent Gauso aukštų dažnių filtrą. Šis filtras turi vieną parametrą, kurį reikia parinkti atsižvelgiant į kadro savybes (fono sudėtingumas, apšvieta, lazerio linijos ryškumas), tai yra filtro dydis – filtro spindulys. Paprastai nustatytas vienas filtro dydis gali tikti esant vienoms sąlygoms, tačiau netiks esant kitoms sąlygoms. Sprendžiant šią problemą filtro dydžio nustatymui yra naudojamas „Hough“ transformacijos gautas rezultatas.

Linijos „Hough“ transformacijos vaizde (pav. 20) paprastai yra ieškomos pagal lokalų maksimumą. Tai yra kai aptinkama didelė reikšmė matricoje o aplink ją esančios reikšmės vis mažėja. Taip yra aptinkami transformacijos pikai. Šie yra pažymėti baltais kvadratais (2.21 pav.).



2.22.1 pav. „Hough“ transformacijos aptikti pikai a)filtro dydis – 10 b)filtro dydis – 100

Jei nuotraukoje turime pakankamai ryškią liniją ji paprastai išryškėja „Hough“ transformacijos vaizde vis didinant aukšto dažnio filtro dydį. Tuo pačiu didelė dalis nereikalingo triukšmo yra nufiltruojama. Taigi, tampa pakankamai aišku, jog filtro dydį reikia parinkti pakankamai didelį, jog būtų nufiltruota didelė dalis triukšmo, tačiau ne per didelį, jog neprarastume mus dominančių duomenų – lazerio linijai priklausančių pikselių.

Taigi, pagal (2.22 pav.) pateiktą grafiką matome, jog vis didinant filtro dydį pikų skaičius „Hough“ transformacijoje vis mažėja kol galiausiai nusistovi. Iš to galime daryti išvadą, jog nuotraukoje tikrai yra matyti ryškių linijų.

2.22 pav. Pikų skaičiaus „Hough“ transformacijoje priklausomybė nuo filtro dydžio.

Kadangi pikų skaičius didinant filtro dydį galiausiai nusistovi, yra parenkamas toks filtro dydis, kuris atitinka nusistovėjimo pradžią. Tai yra, jei didinant filtro dydį pikų skaičius nesikeičia parenkame tą filtro dydį, ties kuriuo buvo pastebėta nusistovėjimo pradžia.

Išvados ir rezultatai

Atlikus 2.2 skyrelyje aprašytą aukštų dažnių filtrų bandymus buvo išanalizuoti 3 aukštų dažnių filtrai: Gauso, „Butterworth“ ir idealusis. Atsižvelgiant į 2.4 lentelėje pateiktus duomenis galima daryti išvadą, jog Gauso filtras yra labiausiai tinkamas iš visų trijų sprendžiant lazerio linijos aptikimo problemą. Jis vidutiniškai išfiltruoja mažiausią procentinę dalį mus dominančios linijos taškų (10.709%), kai tuo tarpu idealusis ir „Butterworth“ filtrai - atitinkamai 34.773% ir 29.698%. Gauso filtras taip pat vidutiniškai palieka mažiausią kiekį triukšmo nuotraukoje (58144 pikseliai) lyginant su idealiuoju ir „Butterworth“ fitlrais (77448 ir 64746 pikseliai).

Bandant nustatyti aukšto dažnio filtro dydį pagal „Hough“ transformacijos rezultatą buvo remiamasi (2.22 pav.) pateiktu grafiku. Dažnu atveju galimų linijų skaičius esant pakankamai dideliam filtro dydžiui galiausiai nusistovi ir pagal tai parinkus filtro dydį, kuris atitinka nusistovėjimo pradžią grafike, pavyksta aptikti liniją. Kol kas negalime šio bandymo rezultatų įvertinti skaitinėmis reikšmėmis.

Taigi, liniją šiuo atveju pavyksta aptikti, tačiau dar reikia tobulinti algoritmus. Jei linija yra suskaidyta į kelis segmentus, tokiu atveju aptinkame tik vieną patį ryškiausią linijos segmentą. Be to, jei linijos storis yra didesnis, mano pateiktas būdas aptinka tik vieną linijos pusę, bet ne linijos vidurį. Tęsiant toliau darbus bus bandoma aptikti ir lygiagrečiai einančią kita linijos pusę ir taip apskaičiuoti linijos vidurį.

Literatūros sąrašas

1. Chan Yee Low, Zamzuri H, Mazlan SA. Simple robust road lane detection algorithm. *ICIAS*. Jun 2014:1-4. <https://ieeexplore.ieee.org/document/6869550>. doi: 10.1109/ICIAS.2014.6869550.
2. R. C. Gonzalez and R. E. Woods, Digital Image Processing, Third Edition, 2008. <http://sdeuoc.ac.in/sites/default/files/sde_videos/Digital%20Image%20Processing%203rd%20ed.%20-%20R.%20Gonzalez%2C%20R.%20Woods-ilovepdf-compressed.pdf>
3. John Canny, A computational Approach to Edge Detection; http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf
4. Algimantas Juozapavičius, Tadas Meškauskas; Vaizdų ir signalų analizė ir apdorojimas; https://www.ebooks.ktu.lt/eb/451/vaizdu-apdorojimas/
5. Different Approaches to Blurring Digital Images and Their Effect on Facial Detection by Erich-MatthewPulfer <https://scholarworks.uark.edu/cgi/viewcontent.cgi?article=1067&context=csceuht>
6. A novellaservisionsensorforweldlinedetection on wall-climbingrobot Liguo Zhang,WeiKe,QixiangYe 1, JianbinJiao n <https://www.sciencedirect.com/science/article/pii/S0030399214000061>

1. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6742594>; A Review of the Pinned Photodiode for CCD and CMOS Image Sensors Eric R. Fossum, Fellow, IEEE, and Donald B. Hondongwa, Student Member, IEEE

1. <https://www.researchgate.net/publication/252062037_A_Simple_Algorithm_for_Fitting_a_Gaussian_Function_DSP_Tips_and_Tricks>; Hongwei Guo Shanghai University 60 PUBLICATIONS   1,064 CITATI

1. <https://www.researchgate.net/profile/Muhammad-Fauzi-6/publication/309114385_Fast_obstacle_distance_estimation_using_laser_line_imaging_technique_for_smart_wheelchair/links/59e789e5458515c3630f92de/Fast-obstacle-distance-estimation-using-laser-line-imaging-technique-for-smart-wheelchair.pdf>; WHEELCHAIR

1. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6815509>; A Method for Image Processing and Distance Measuring Based on Laser Distance Triangulation

1. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7129023>; The Laser Color detection for 3D Range Scanning Using Gaussian Mixture Model P. Chmelar

1. <https://www.researchgate.net/publication/350124875_Laser_spot_detection>; Laser Spot Detection M.Mesko

1. <https://publications.waset.org/9996622/the-laser-line-detection-for-autonomous-mapping-based-on-color-segmentation>; The Laser Line Detection for Autonomous Mapping Based On Color Segmentation
2. Yining Deng, B. S. Manjunath, Hyundoo Shin. Color Image Segmentation [žiūrėta 2020m. sausio 12d.] Internetinė prieiga: <https://pdfs.semanticscholar.org/be59/65b3659f1846b417939b80e8d1a081505fa6.pdf>
3. Contrast Adjustment And Image Normalization [žiūrėta 2020m. sausio 12d.] Internetinė prieiga: <https://www.giassa.net/?page_id=472>
4. P.Kasparaitis. Skaitmeninis vaizdų apdorojimas. Morfologinis apdorojimas ir skeletizavimas [žiūrėta 2020m. sausio 14d.] Internetinė prieiga: <https://klevas.mif.vu.lt/~pijus/SVA/skelmorf.pdf>
5. R. Fisher; S. Perkins; A. Walker; E. Wolfart. [Connected Component Labeling](http://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm) [žiūrėta 2020m. sausio 14d.] Internetinė prieiga: <http://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm>
6. Defining Connectivity [žiūrėta 2020m. sausio 14d.] Internetinė prieiga: <http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracig_Abeer_George_Ghuneim/connect.html>
7. Arnold Bijman. A low cost 3D laser-line scanner for facial acquisition [žiūrėta 2020m. sausio 15d.] Internetinė prieiga: <https://essay.utwente.nl/72064/1/Bijman_BA_EEMCS.pdf>
8. Image Processing. Thresholding [žiūrėta 2020m. sausio 15d.] Internetinė prieiga: <https://mmeysenburg.github.io/image-processing/07-thresholding/>
9. Moving Average. [žiūrėta 2020m. sausio 15d.]. Internetinė prieiga: <https://en.wikiversity.org/wiki/Moving_Average>
10. ValtenFx Electronics Innovation, Obstacle Detection Using Laser And Image Processing [žiūrėta 2020m. sausio 16d.]. Internetinė prieiga: http://valentfx.com/obstacle-detection-using-laser-and-image-processing/
11. David Jacobs. Image Gradients [žiūrėta 2020m. sausio 15d.], Internetinė prieiga: <http://www.cs.umd.edu/~djacobs/CMSC426/ImageGradients.pdf>
12. R. Fisher, S. Perkins, A Walker, E. Wolfart, “*Fourier Transform”,* 2003, Hypermedia Image Processing Reference. [žiūrėta 2020-06-02]. Internetinė prieiga: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm?fbclid=IwAR13t9VTIgHMP28giCfS5GDb-8clbLk97yMXQsaJveVj_cySkk330gvnh-I>
13. “Fast Fourier transform – FFT”, *Cooley-Tukey technique,* Article. 10, “A simple, pedagogical radix-2 algorithm in C++”. [žiūrėta 2020-06-02]. Internetinė prieiga: <http://www.librow.com/articles/article-10>
14. T. Cormen, D. Balkcom, Khan Academy, *Divide and conquer algorithms*, Article, [žiūrėta 2020-06-02]. Internetinė prieiga: <https://www.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/divide-and-conquer-algorithms>
15. C. Chen, “*Digital Image Processing using Fourier Transform in Python*”. [žiūrėta 2020-06-02]. Internetinė prieiga: <https://medium.com/@hicraigchen/digital-image-processing-using-fourier-transform-in-python-bcb49424fd82>
16. P. Bourke, “*Image Filtering in the Frequency Domain”,* 1998. [žiūrėta 2020-06-02]. Internetinė prieiga: <http://paulbourke.net/miscellaneous/imagefilter/>
17. A.Dogra, P. Bhalla, “*Image Sharpening By Gaussian And Butterworth High Pass Filter*”, 2014, Biomedical & Pharmacology Journal. [žiūrėta 2020-06-02]. Internetinė prieiga: <https://biomedpharmajournal.org/vol7no2/image-sharpening-by-gaussian-and-butterworth-high-pass-filter/>
18. R. Fisher, S. Perkins, A Walker, E. Wolfart, “*Hough Transform”,* 2003, Hypermedia Image Processing Reference. [žiūrėta 2020-06-02]. Internetinė prieiga: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm>

1. Utaminingrum F, Fitriyah H, Wihandika RC, Fauzi MA, Syauqy D, Maulana R. Fast obstacle distance estimation using laser line imaging technique for smart wheelchair. *International journal of electrical and computer engineering (Malacca, Malacca)*. 2016;6(4):1602. <https://search.proquest.com/docview/1821424986>. doi: 10.11591/ijece.v6i4.9798.

2. Chan Yee Low, Zamzuri H, Mazlan SA. Simple robust road lane detection algorithm. *ICIAS*. Jun 2014:1-4. <https://ieeexplore.ieee.org/document/6869550>. doi: 10.1109/ICIAS.2014.6869550.

stylefix