



**Kauno technologijos universitetas**

Informatikos fakultetas

# **Lazerio linijos aptikimo algoritmas**

Baigiamasis magistro projektas

---

**Tadas Ivanovas**

Projekto autorius

**doc. dr. Armantas Ostreika**

Vadovas

---

**Kaunas, 2021**



**Kauno technologijos universitetas**

Informatikos fakultetas

## **Lazerio linijos aptikimo algoritmas**

Baigiamasis magistro projektas

Informatika (6211BX007 )

---

**Tadas Ivanovas**

Projekto autorius

**doc. dr. Armantas Ostreika**

Vadovas

**dr. Alfonsas Misevičius**

Recenzentas

---

**Kaunas, 2021**



**Kauno technologijos universitetas**

Informatikos fakultetas

Tadas Ivanovas

## **Lazerio linijos aptikimo algoritmas**

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Tado Ivanovo, baigiamasis projektas tema „Lazerio linijos aptikimo algoritmas“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

Tadas Ivanovas - 

(vardą ir pavardę įrašyti ranka)

(parašas)

Ivanovas, Tadas. Lazero linijos aptikimo algoritmas. Magistro baigiamasis projektas / vadovas doc. Armantas Ostreika; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatika (I100).

Reikšminiai žodžiai: lazero linija, aptikimas, algoritmas.

Kaunas, 2021. 54 p.

## **Santrauka**

Lazero linijos aptikimas yra plačiai naudojamas įvairiose robotikos bei automatikos srityse. Yra įvairių aplikacijų, kur naudojama vaizdo kamera ir lazero linija. Pavyzdžiui, išmanusis neįgaliųjų vežimėlis, 3D objektų skeneris ir įvairūs mobilūs robotai, skirti suvirinimo siūlų nustatymui ar atstumų išmatavimui.

Analizuojant jau sukurtus lazero linijos aptikimo algoritmus, buvo pastebėta, jog daugelis jų yra paremti paprastais spalvų segmentavimo metodais. Tokie būdai nėra adaptyvūs. Nustačius slenkstines ribas lazero linija gali būti aptinkama tik esant statinėms fono sąlygoms. Dėl to šiame darbe yra pasiūlytas naujas lazero linijos aptikimo algoritmas, kuris veikia esant įvairioms fono sąlygoms.

Pasiūlytas algoritmas yra iširtas ir įvertintas naudojant įvairias skirtingų rezoliucijų eksperimentines nuotraukas. Tyrimas atliktas naudojant sukurta programinį įrankį, kuris apskaičiuoja vaizde aptiktos lazero linijos posūkio kampo ir taškų paklaidą. Tyrimo metu taip pat yra vertinama ir algoritmo greitimeika.

Gauti tyrimo rezultatai akcentuoja pasiūlyto algoritmo platų veikimo diapazoną. Daugelyje nuotraukų aptikimas yra sėkmingas, nepaisant turimo vaizdo rezoliucijos ar fono sąlygų. Algoritmo universalumą įrodo ir tai, jog viso tyrimo metu nėra keičiami jokie algoritmo parametrai, dėl to jis reikalauja minimalaus žmogaus įsikišimo. Žinoma, siekiant geresnio algoritmo veikimo, šie parametrai dar gali būti derinami atsižvelgiant į konkrečią aplikaciją.

Ivanovas, Tadas. Laser line detection algorithm. Master's Final Degree Project / supervisor assoc. prof. Armantas Ostreika; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Informatics (I100).

Keywords: laser line, detection, algorithm.

Kaunas, 2021. 54 pages.

### **Summary**

Laser line detection is widely used in various fields of robotics and automation. There are many different applications that use video camera and laser line. For example, smart wheelchair, 3D object scanner of mobile robots that are used for weld line detection or distance measurement.

After the analysis of already proposed laser line detection algorithms it was observed that many of them are based on color segmentation methods. As already known, these methods are not adaptive because static thresholds are suitable only for images with static background. Thresholds must be changed each time according to background conditions. This is why this project proposes new laser line detection algorithm that works in a complex environment.

Proposed algorithm is evaluated by using various photos of different resolutions. Analysis of the algorithm was conducted with created software tool which calculates the error of detected line angle and closest distance between reference line points and detected line points. The processing time of each photo is also taken into consideration during the analysis of the proposed algorithm.

The results of the algorithm experiment highlight the wide detection range of the algorithm. The laser line detection is successful with the most of experimental photos despite different resolution or background conditions. The universality of the algorithm is proved by the fact that none of the algorithm parameters are changed during the experiment. Of course, these parameters can be adjusted considering the application this algorithm is used by.

## Turinys

<b>Turinys</b> .....	<b>6</b>
<b>Įvadas</b> .....	<b>7</b>
<b>1. Lazerio linijos aptikimo metodų apžvalga</b> .....	<b>8</b>
1.1. Kelio linijų aptikimo algoritmas.....	8
1.2. Lazerio linijos aptikimas skirtas suvirinimo linijų nustatymui .....	10
1.3. Kliūčių aptikimas naudojant lazerio liniją.....	12
1.4. Lazerio linijos aptikimas atstumo nustatymui .....	13
1.5. 3D lazerio linijos skeneris .....	16
1.6. Darbe naudojamų metodų apžvalga .....	17
1.6.1. Furjė transformacija.....	17
1.6.2. Hofo transformacija.....	20
1.7. Aukšto dažnio filtrai .....	22
<b>2. Lazerio linijos aptikimo metodologija</b> .....	<b>25</b>
2.1. Dažnių spektro filtravimas .....	25
2.1.1. Aukšto dažnio filtrų tyrimas .....	27
2.2. Linijų paieška .....	31
2.2.1. Branduolio sukimas .....	31
2.2.2. Binarinio vaizdo formavimas .....	33
2.2.3. Binarinio vaizdo apdorojimas.....	34
2.2.4. Linijų aptikimas ir saugojimas binariniame vaizde.....	35
2.3. Lazerio linijos nustatymas .....	38
2.3.1. Linijos centro radimas .....	40
<b>3. Algoritmo tyrimas</b> .....	<b>42</b>
3.1. Tyrimo programinis įrankis .....	42
3.2. Tyrimo eiga .....	46
3.3. Tyrimo rezultatai .....	48
<b>Išvados</b> .....	<b>52</b>
<b>Literatūros sąrašas</b> .....	<b>53</b>

## Įvadas

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 komplikuoatas. 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 ryškia 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 linijos nuskaitymo segmentacija.

Atsižvelgiant į kilusias problemas, yra pasiūlytas sprendimas, kuris, manoma, išspręstų daugelį minėtų uždavinių. Lazerio liniją siūloma aptikti atsižvelgiant į nuotraukos dažninį spektrą, nes ryški lazerio linija vaizde sukelia aukštus dažnius. Toliau linijos aptikimas išfiltruotame vaizde vyksta sukant pailgą linijos aptikimo branduolį. Pasitelkiant tam tikras morfologines operacijas ir Hofs transformaciją, yra nustatoma vaizde matomos linijos pozicija. Galiausiai lazerio linija yra aptinkama remiantis jos pikselių intensyvumo parametrais.

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. Taip pat yra svarbus pasiūlyto algoritmo platus veikimo diapazonas.

**Darbo tikslas** – išanalizuoti jau esamus lazerio linijos aptikimo sprendimus ir pasiūlyti naują algoritmą, kuris veiktų esant sudėtingomis fono sąlygomis.

### **Pagrindiniai darbo uždaviniai:**

- Lazerio linijos metodų aptikimo apžvalga.
- Gauso, idealųjį ir Butterworth'o aukšto dažnio filtrų analizė.
- Pasiūlyti lazerio linijos aptikimo algoritmą.
- Ištirti algoritmo tikslumą ir greitaveiką.

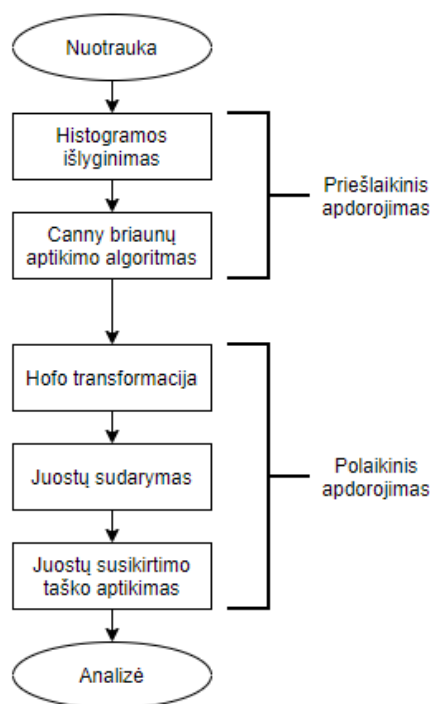
Projekto pradžioje yra apžvelgiama literatūra. Pirmame skyriuje yra pabrėžiamas lazerio linijos aptikimo aktualumas. Yra gausa aplikacijų, kur yra naudojami įvairūs lazerio linijos aptikimo metodai. Kai kurios tų aplikacijų yra glaudžiai susijusios su robotika bei automatika, pavyzdžiui, įvairūs mobilūs robotai. Pastebėta, jog daugelyje tokių aplikacijų lazerio linija aptinkama naudojant paprastus spalvos segmentavimo metodus, kurie, iš esmės, nėra adaptyvūs. Pateiktuose pavyzdžiuose yra matomos ganėtinai paprastas lazerio linijos fonas. Šiame skyriuje taip pat yra aptariami esminiai metodai, kurie yra naudojami pasiūlytam algoritmui. Antrasis skyrius yra skirtas detaliam viso algoritmo apžvalgai. Kiekviena algoritmo dalis yra išanalizuojama pateikiant vaizdinius pavyzdžius. Paskutinis šio darbo skyrius yra skirtas algoritmo tyrimui. Jame yra aprašomas sukurtas testavimo įrankis bei visa eksperimento eiga. Čia taip pat yra pateikiami pavykusių ir nepavykusių eksperimentų pavyzdžiai. Galutinis tyrimo rezultatas yra įvertintas lazerio linijos aptikimo algoritmo tikslumu bei greitaveika.

## 1. 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 aptariamoms tik tos šaltinių dalys, kurios susijusios būtent su linijos aptikimo metodika ir tik minimaliai aptariami panaudos atvejai.

### 1.1. 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], Hoho transformacija [51 p. 4] ir juostų sudarymas), 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 didesnę kontrastą. Tai yra daroma siekiant išryškinti kelio linijų žymėjimus, nes 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 apskaičiuoja binarinį vaizdą, kuris suteikia visą



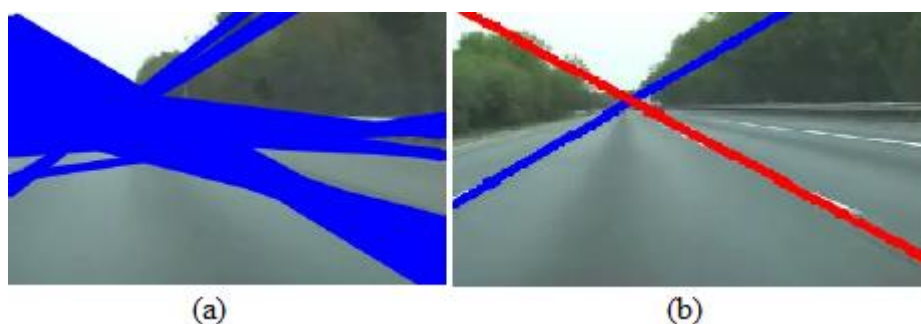
reikiamą informaciją aptinkant linijas (žr. 1.2 pav.). Siekiant sumažinti triukšmą gautame vaizde yra naudojamos morfologinės operacijos: skaitmeninė erozija [44 p. 4] ir skaitmeninis auginimas [42 p. 4]. Kartu yra pasitelkiamas ir vaizdo glotninimas [5].



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 tai, 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.

Hofa 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 iki vieno iš vaizdo kampų ilgis. Tada sugrupavus visas linijas, kurių parametrai panašūs, yra apskaičiuojamas tų parametru 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;
- kelio plotis yra pastovus arba su minimaliais pokyčiais;
- kelio linijos turi sekti griežtas išvaizdos taisykles;
- negali būti ekstremalios oro sąlygos (liūtis, sniegas).

## 1.2. Lazero linijos aptikimas skirtas suvirinimo linijų nustatymui

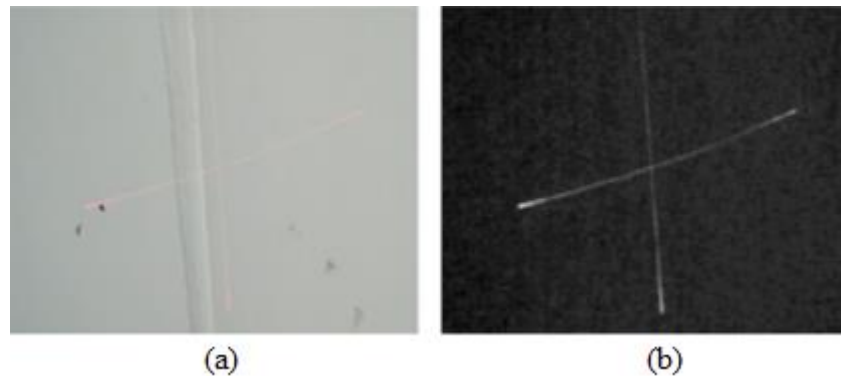
Struktūrizuotos šviesos jutikliai pritraukia vis daugiau dėmesio ir yra plačiai naudojami įvairiose automatizavimo bei robotikos srityse, pavyzdžiui, automatinis virinimas, kokybės kontrolė ar robotų navigacija [6]. Taigi, toliau šiame poskyryje yra kalbama apie struktūrizuotos lazerio šviesos linijų 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:

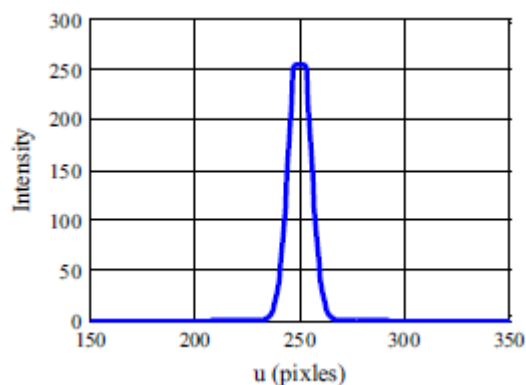
$$I_{i,j} = \max\{|I_{i,j}^R - I_{i,j}^G|, |I_{i,j}^R - I_{i,j}^B|\}, i \in [0, m), j \in [0, n), \quad (1.1)$$

čia  $I_{i,j}$  – pikselio vertė  $i$  eilutėje ir  $j$  nuotraukos stulpelyje, o  $I_{i,j}^R$ ,  $I_{i,j}^G$  ir  $I_{i,j}^B$  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.

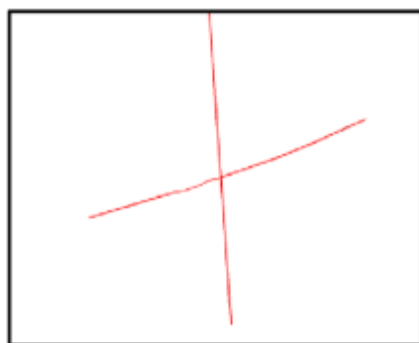


1.5 pav. Pikselių intensyvumas per lazerio linijos plotį [6]

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  $M_i(w)$  eilutėje  $i$  yra apskaičiuojama pagal formulę:

$$M_i(w) = \frac{\sum_{j_{max}^i - (w/2)}^{j_{max}^i + (w/2)} I_{i,j} \cdot j}{\sum_{j_{max}^i - (w/2)}^{j_{max}^i + (w/2)} I_{i,j}}, \quad i \in [0, m), j \in [0, n), \quad (1.2)$$

čia  $j_{max}^i$  – 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.).



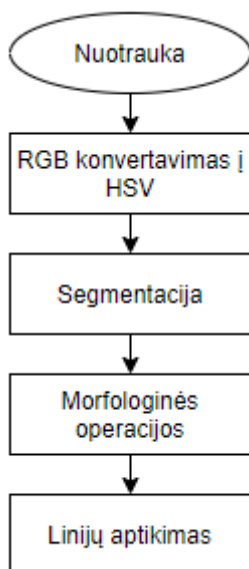
1.6 pav. Nustatyti lazerio linijų centrai [6]

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.

### 1.3. Kliūčių aptikimas naudojant lazerio liniją

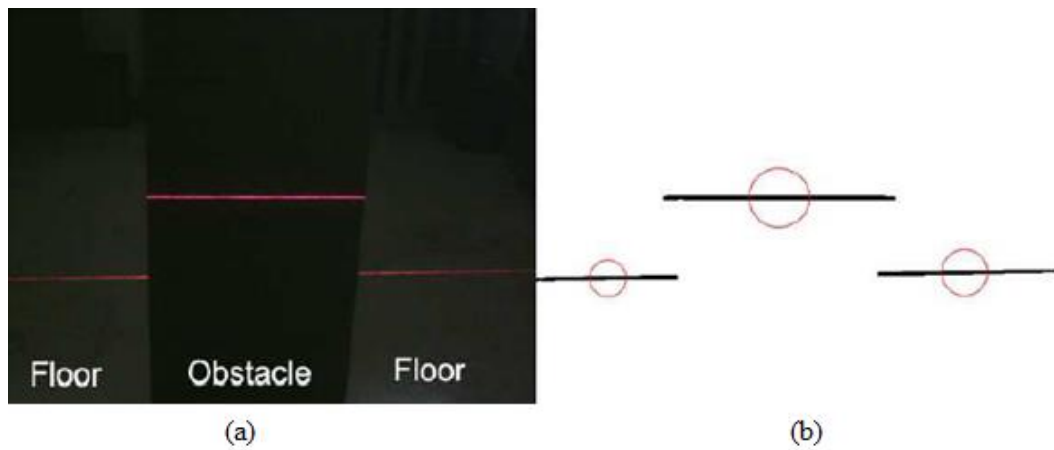
Automatizuoti judantys objektai, tokie kaip, pavyzdžiui, išmanieji neįgaliųjų vežimėliai privalo turėti atstumų nustatymo ir kliūčių aptikimo sistemas. Dėl to toliau yra pateikiamas lazerio linijos aptikimo metodas, kuris yra naudojamas būtent tokiose aplikacijose [9]. Naudojant ant kelio projektuojamą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

Pateiktas algoritmas (žr. 1.7 pav.) naudoja RGB spalvų schemas 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 struktūrinių 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*) [13]. Reiškia, jeigu binariniame vaizde esantis vienetas bet kokia kryptimi (vertikalia, horizontalia ar įstriža) ribojasi su kitu vienetu, tada tas pikselis priklauso tam pačiam linijos segmentui. Po to yra skaičiuojamas kiekvieno segmento centras naudojant masės centrą pagal lygtis:

$$\text{center}_x = \frac{1}{n} \sum_{k=1}^n x_k, \quad (1.3)$$

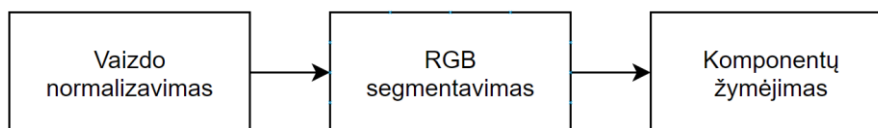
$$\text{center}_y = \frac{1}{n} \sum_{k=1}^n y_k, \quad (1.4)$$

čia  $\text{center}_x$  ir  $\text{center}_y$  yra  $x$  ir  $y$  centro koordinatės,  $n$  – pikselių, kurie priklauso vienam linijos segmentui skaičius, o  $x_k$  ir  $y_k$  – atitinkamai pikselių koordinatės. Galutinis algoritmo rezultatas pavaizduotas (žr. 1.8 pav.).

Taigi, šis algoritmas yra išties ganėtinai paprastas. Pagrindinis lazereo 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 lazereo aptikimui, jei to objekto pikselių reikšmės atitiks nurodytą diapazoną.

#### 1.4. Lazereo linijos aptikimas atstumo nustatymui

Lazereo linijos projekcijos aptikimas šiandien labai plačiai naudojamas įvairiose robotikos bei elektroninių matavimo prietaisų srityse. Aptikimo sistemos kokybė yra priklausoma nuo lazereo 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 [15]. Šiame poskyryje vėlgi yra aptariama tik lazereo linijos aptikimo metodika nekalbant apie jos pritaikymą konkrečiai aplikacijai.

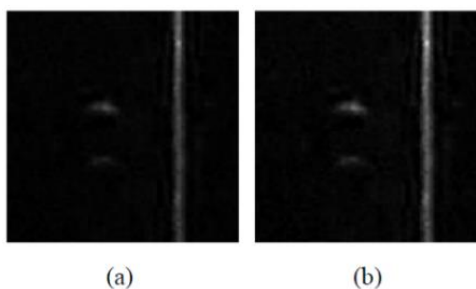


1.9 pav. Lazerio linijos aptikimo algoritmo struktūrinė schema

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 didesnę bendrą kontrastą, taip išryškindamas silpnai matomus objektus (žr. 1.10 pav.). Šiuo atveju tokia pikselių intensyvumo reikšmių korekcija yra atliekama tik su raudonoju kadro kanalu, nes norimos aptikti lazerio linijos spalva yra raudona. Normalizavimas yra apskaičiuojamas pagal formulę:

$$I_{Ni}(R) = \frac{R_i}{\max(R)}, \quad (1.5)$$

čia  $I_{Ni}(R)$  – normalizuota raudonojo kanalo pikselio intensyvumo reikšmė,  $i$  – pikselio indeksas.



1.10 pav. Vaizdo normalizavimo pavyzdys (a) originali nuotrauka

(b) normalizuota nuotrauka [15]

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 nustatius konkrečias slenkstines ribas kiekvienam kanalui. Vienas pikselis laikomas priklausančiu raudonai lazerio linijai, jeigu atitinka išraišką:

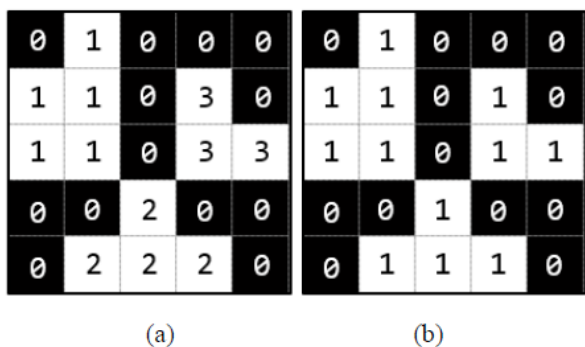
$$I_L = I(R > T_R) \& I(G < T_G) \& I(B < T_B), \quad (1.6)$$

čia  $I_L$  – išskirti iš fono lazerio linijos pikseliai.  $T_R$ ,  $T_G$  ir  $T_B$  yra atitinkamų spalvos kanalų slenkstinės ribos. Dar prieš pradėdant 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.).



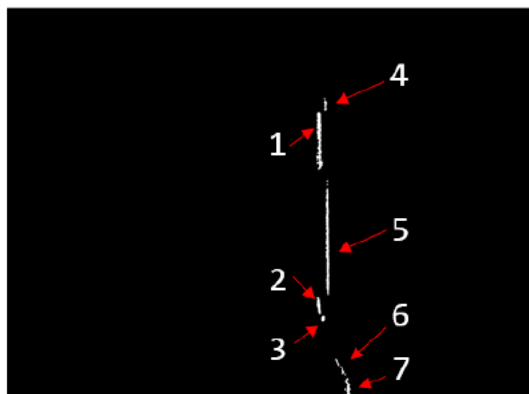
1.11 pav. Spalvų segmentavimo rezultatas (a) originalus vaizdas, (b) binarinis lazerio linijos vaizdas [15]

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 [13], 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 [15]

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 [15]

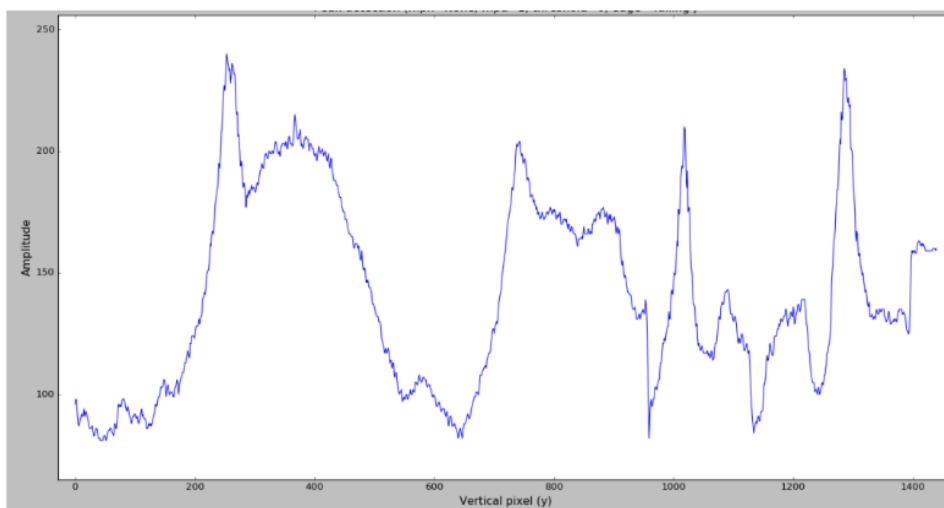


Taigi, šiame poskyryje pateiktas lazerio aptikimo algoritmas iš esmės yra paremtas tik spalvų segmentavimu nustatant slenkstines ribas kiekvienam kanalui RGB vaizde. Toks būdas reikalauja labai tikslaus slenkstinių ribų nustatymo, kuris turi būti koreguojamas atsižvelgiant į tai, kaip ryškiai yra atspindima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

### 1.5. 3D lazerio linijos skeneris

3D lazerio linijos skeneris naudoja mažiausiai vieną lazerio linijos projekciją ir vaizdo kamerą objekto taškų atkūrimui erdveje. Kiekvienas unikalus objekto taškas gali būti apskaičiuojamas naudojant trianguliacijos principą, kai yra žinomas kampas tarp vaizdo kameros ir lazerio liniją projektuojančio prietaiso [16]. Toliau šiame poskyryje yra detalizuojamas tik algoritmas, skirtas lazerio linijos aptikimui, kuris yra naudojamas būtent 3D skenavimo sistemoje.

Paprasčiausiu skenavimo atveju yra naudojama viena pastovaus posūkio kampo lazerio linija. Šiuo atveju, pavyzdžiui, yra naudojama horizontali raudona lazerio linija, kuri yra projektuojama ant objekto. Tada vaizdo kameros pagalba yra užfiksuojamas kadras ir pradedamas priešlaikinis apdorojimas siekiant aptikti lazerio liniją kadre. Kadangi linijos kryptis yra horizontali, norint aptikti visus jos taškus reikia analizuoti kiekvieną kadro pikselių stulpelį atskirai.



1.14 pav. Pikselių stulpelio intensyvumo kitimo grafikas raudonajame kanale [16]

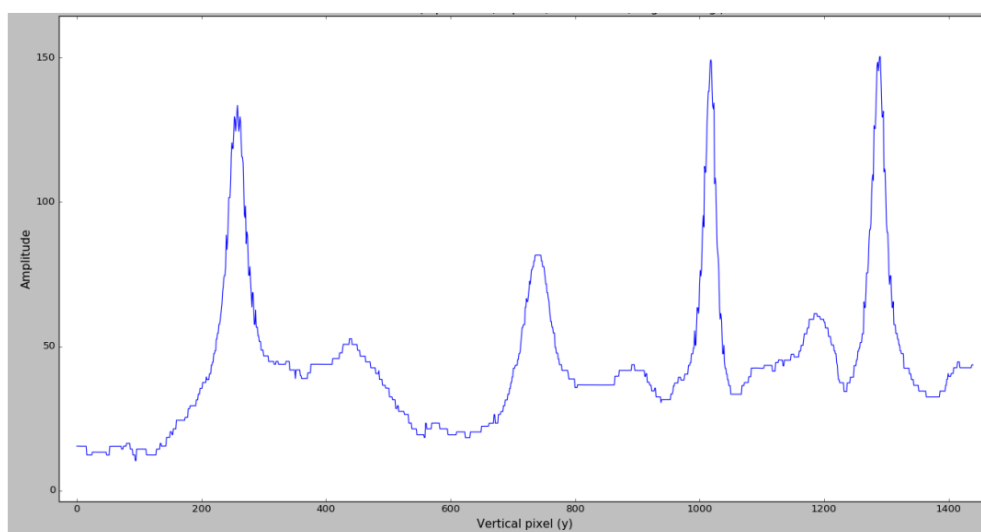
Turint vieno pikselių stulpelio RGB reikšmių duomenis yra atliekamas baltos šviesos filtravimas. Toks apdorojimas yra svarbus siekiant padidinti algoritmo linijos aptikimo kokybę, nes iš aplinkos sklindanti šviesa gali sukelti nepageidaujamų pikselių intensyvumo padidėjimų, kurie grafike atsiranda kaip intensyvumo pikas (žr. 1.14 pav.). Kadangi balta šviesa apytiksliai turi vienodas reikšmes visuose RGB kanaluose, ji yra išfiltruojama iš raudonojo kanalo atėmus žaliojo ir mėlynojo kanalų vidurkį:

$$I_R = I_R - \frac{I_B + I_G}{2}, \quad (1.7)$$

čia  $I_R$ ,  $I_G$  ir  $I_B$  – atitinkamų kanalų pikselių intensyvumo reikšmės.



Atlikus baltos šviesos filtravimą galima pastebėti, jog išanalizavus vieną pikselių stulpelį, grafike yra turimas mažesnis skaičius intensyvumo pikų (žr. 1.15 pav.).



1.15 pav. . Pikselių stulpelio intensyvumo kitimo grafikas raudonajame kanale atlikus baltos šviesos filtravimą [16]

Lazerio linijos taškų radimas iš esmės yra vykdomas ieškant pikselių intensyvumo pikų kiekviename stulpelyje. Tiesiog paprasta statinė slenkstinė riba netinka pikų radimui, nes skenuojant yra nevienodos apšvietimo sąlygos skirtingose skenuojamo objekto vietose. Siekiant lazerio linijos taškų aptikimą padaryti patikimesniu, yra pritaikomas bėgančio vidurkio metodas (angl. *running average*). Reiškia, skenuojant pikselių stulpelį, kiekvienam pikseliui atskirai yra suskaičiuojamas intensyvumo vidurkis kartu su prieš jį esančiais pikseliais. Toks apskaičiuotas vidurkis kiekviename taške yra naudojamas kaip to taško slenkstinė riba (angl. *threshold*), prie kurios taip pat dar yra pridedama ir statinė, empiriškai nustatyta kita slenkstinė riba. Visi pikseliai, kurie turi didesnę intensyvumą nei apskaičiuota slenkstinė riba yra laikomi lazerio linijos taškais.

Taigi, toks lazerio linijos aptikimo algoritmas iš esmės yra glaudžiai pririštas prie linijos krypties. Norint, pavyzdžiui, skenuoti vertikalia kryptimi, tektų keisti algoritmo veikimą ir pikselius analizuoti kiekvienoje eilutėje, o ne stulpelyje. Kiek sudėtingiau būtų, jeigu lazerio projekcija turėtų įstrižą kryptį, tačiau, žinoma, reikia atsižvelgti į tai, jog šis algoritmas yra naudojamas objektų skenavimui kur lazerio linijos kryptis dažniausiai yra nekintama.

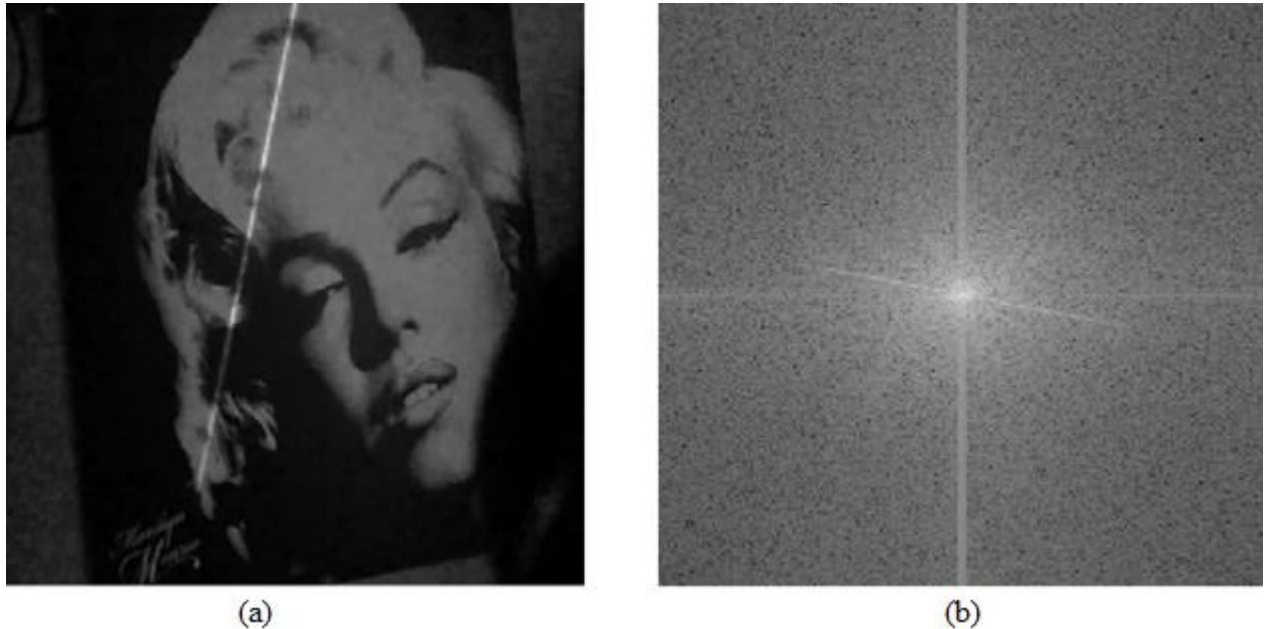
## 1.6. Darbe naudojamų metodų apžvalga

Šiame poskyryje yra aptariami jau esami algoritmai bei metodai, kurie naudojami šiame darbe. Kadangi kitame skyriuje pateiktas algoritmas iš esmės yra paremtas šiais metodais, dėl to yra labai svarbu suprasti esminį jų veikimo principą. Čia yra apžvelgiama Furjė transformacija, Hoho transformacija bei aukštų dažnių filtrai.

### 1.6.1. Furjė transformacija

Furjė transformacija yra svarbi vaizdo apdorojimo priemonė, naudojama vaizdui suskaidyti į jo sinuso ir kosinuso komponentus. Ji naudojama įvairiose srityse, tokiose kaip vaizdo analizė, vaizdo filtravimas, vaizdo rekonstravimas ir vaizdo glaudinimas. Transformacijos išvestis atspindi vaizdą Furjė arba dažnio srityje, o įvesties vaizdas yra erdvinės srities ekvivalentas [17]. Furjė domeno

paveikslėlyje kiekvienas taškas žymi tam tikrą erdvinio domeno vaizdo dažnį. Paprasčiau tariant, funkcijos ar signalo Furjė transformacija, šiuo atveju vaizdo ar nuotraukos, yra kompleksiniais skaičiais įvertinama dažnio funkcija, kurios dydis ar kitaip tariant, absoliuti reikšmė, parodo to specifinio dažnio kiekį, esantį pradinėje funkcijoje (vaizde erdviniam domene). Šios operacijos pavyzdys pateiktas žemiau (žr. 1.16 pav.).



1.16 pav. Nuotraukos Furjė transformacija (a) originali nuotrauka (b) atvaizduotas nuotraukos dažnių spektras

Šiame darbe iš esmės yra kalbama apie diskrečius signalus (skaitmeninius vaizdus), todėl tolesnė analizė apsiriboja diskrečiąja Furjė transformacija (angl. *DFT* – Discrete Fourier Transform). DFT yra diskretinė arba imtinė Furjė transformacijos forma. Joje nėra visų įmanomų vaizde 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 erdviniam, tiek dažnių srityje yra vienodas.  $N$  dydžio kvadratiniam atvaizdai dvimatis DFT apskaičiuojamas taip:

$$F(k, l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m, n) e^{-i2\pi(\frac{km}{N} + \frac{ln}{N})}, \quad (1.8)$$

čia  $f(m, n)$  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 rezultata.

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

$$f(k, l) = \frac{1}{N^2} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} F(m, n) e^{i2\pi(\frac{km}{N} + \frac{ln}{N})}, \quad (1.9)$$

čia  $f(m, n)$  ir  $F(k, l)$  turi tas pačias reikšmes, kaip ir (1.8) lygtyje. Lygtyje (1.9) reikia atkreipti dėmesį į  $\frac{1}{N^2}$  trupmeną, kuri naudojama atvirkštinėje Furjė transformacijoje. Kartais ji gali būti naudojama ir tiesioginėje transformacijoje, tačiau niekada abiejose.

Remiantis (1.8) ir (1.9) lygtimis, darosi akivaizdu, jog dvimatį signalą konvertuoti iš erdvinio domeno į dažninį reikalauja pakankamai daug skaičiuojamosios galios. Dėl to įprastai aplikacijose yra naudojama FFT (angl. *Fast Fourier Transform*). Tai yra optimizuota DFT, o pats populiariausias jos algoritmas yra Cooley-Tukey greitosios Furjė transformacijos algoritmas [14]. Tai yra „skaldyk ir valdyk“ tipo algoritmas, kuris skaičiavimo problemą padalija į daug mažesnių problemų, kurios skaičiavimo požiūriu yra daug pigesnės. Jau yra žinoma, jog DFT yra apibūdinama lygtimi:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}nk}, \quad (1.10)$$

čia  $X_k$  yra signalas dažnių srityje,  $x_n$  – konvertuojamas signalas,  $N$  – signalo dydis, o  $k$  yra sveikasis skaičius nuo 0 iki  $N-1$ . Pačiu paprasčiausiu atveju šis FFT algoritmas įgyja skaičiavimo greičio pranašumą, kai signalas yra dalijamas į du atskirus masyvus. Viename jų yra signalo reikšmės, kurios turi lyginę indekso reikšmę, o kitame – nelyginę. Tokiu atveju (1.10) lygtį galima perrašyti taip:

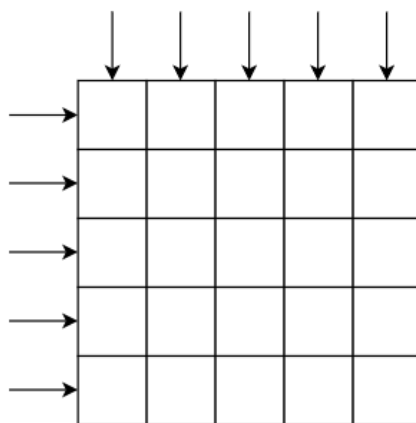
$$X_k = \sum_{m=0}^{\frac{N}{2}-1} x_{2m} e^{-\frac{2\pi i}{N}(2m)k} + \sum_{m=0}^{\frac{N}{2}-1} x_{2m+1} e^{-\frac{2\pi i}{N}(2m+1)k}, \quad (1.11)$$

Tada atlikus nesudėtingus tarpinius prastinimus, sutrakus bendrąjį daugiklį iš (1.11) išraiškos, DFT galima aprašyti taip:

$$\begin{aligned} X_k &= E_k + e^{-\frac{2\pi i}{N}k} O_k, \\ X_{k+\frac{N}{2}} &= E_k - e^{-\frac{2\pi i}{N}k} O_k, \end{aligned} \quad (1.12)$$

čia  $E_k$  lyginių indekso verčių DFT, o  $O_k$  – nelyginių. Pirmiausia algoritmas apskaičiuoja lyginių ir nelyginių indeksuotų verčių diskrečiąją Furjė transformaciją. Tai yra svarbiausia dalis, kur algoritmas įgauna savo greitį. Signalas rekursyviai dalijamas į dvi mažesnes DFT kol signalo tampa nebeįmanoma padalinti į dvi lygias dalis.. Tokiu būdu yra išvengiama daug papildomų skaičiavimų, kadangi mažesnių transformacijų rezultatai yra naudojami daug kartų apskaičiuoti visą transformaciją.

Remiantis (1.12) formule, pateiktas algoritmas tinka tik vienmačiam signalui skaičiuoti. Nepaisant to, tai nėra sunku atlikti naudojant dvimatį signalą, šiuo atveju – vaizdą ar nuotrauką. Vaizdo dažnio spektras apskaičiuojamas pirmiausia atliekant DFT viena kryptimi, o po to kita (žr. 1.17 pav.).



1.17 pav. Dvimačio signalo FFT skaičiavimas

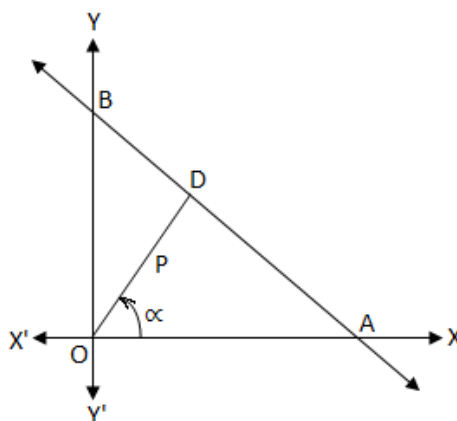
Vieną pikselių eilutę ar stulpelį galima laikyti vienmačiu signalu, kurio diskrečiąją Furjė transformaciją galima suskaičiuoti naudojant „Cooley-Tukey“ algoritmą pagal (1.12) išraišką.. Iš pradžių tai daroma su kiekviena eilute ir po to su kiekviena stulpeliu, o rezultatas susumuojamas į vieną dažnio spektrą (žr. 1.16 pav. (b)).

### 1.6.2. Hofo transformacija

Hofo transformacija yra metodas, kuris yra naudojamas tam tikrų formų ir požymių išskyrimui vaizde. Kadangi yra reikalavimas, jog norimos formos savybės būtų apibrėžtos tam tikra parametru erdve, klasikinė Hofo transformacija dažniausiai naudojama nustatant įprastas kreives, tokias kaip linijos, apskritimai ar elipsės. Šiuo atveju yra kalbama apie Hofo transformacijos naudojimą linijos aptikimui. Pagrindinis principas grindžiamas normaliaja linijos forma:

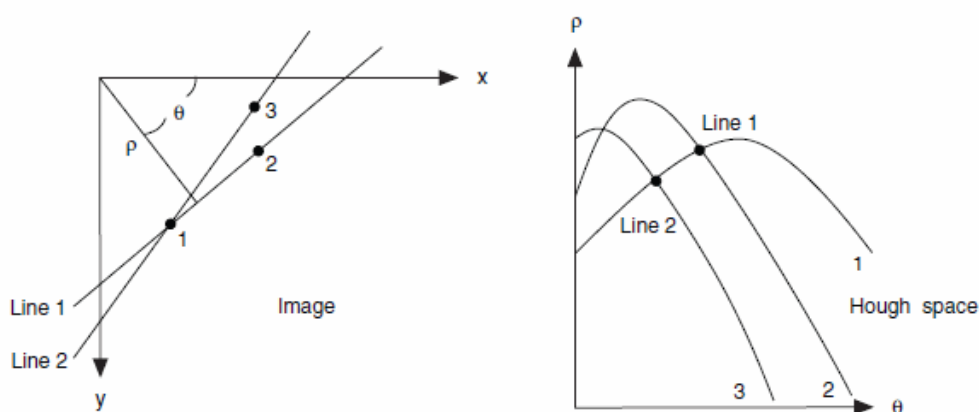
$$P = x \cos(\alpha) + y \sin(\alpha), \quad (1.13)$$

,kur  $P$  yra atraminės linijos ilgis, o  $\alpha$  yra kampas tarp atraminės linijos ir pradinės ašies (žr. 1.18 pav.). Atraminė linija visada yra statmena linijai, kurią norima apibrėžti pagal (1.13) išraišką.



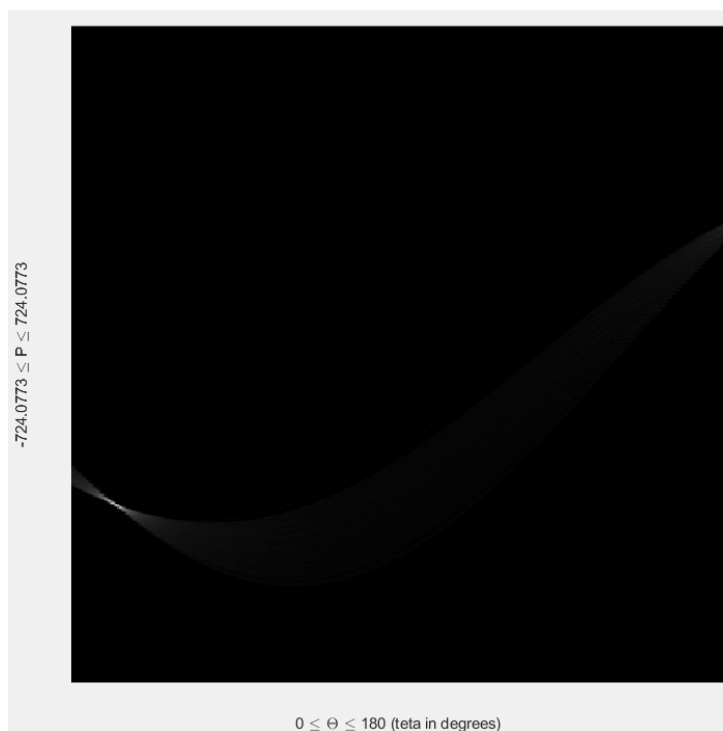
1.18 pav. Normalioji linijos forma

Įvestis į šį algoritmą privalo būti binarinis vaizdas. Hofo transformacija tada apdoroja kiekvieną vienetu pažymėtą pikselį binariniame vaizde. Kiekvienas jų atitinkamai pagal (1.13) formulę yra konvertuojamas į reikšmę Hofo erdvėje (žr. 1.19 pav.). Tokiu būdu kiekvienam pikseliui, kurio reikšmė lygi vienetui binariniame vaizde, yra apskaičiuojama pozicija Hofo erdvėje. Šiuo atveju jos vertikalioji ašis apibrėžia  $\rho$  (atraminės linijos ilgis), o horizontalioji –  $\theta$  (kampas tarp atraminės linijos ir koordinatinių pradžių). Taigi, kuo daugiau taškų binariniame vaizde sudaro tiesią liniją, tuo daugiau taškų atitinka tą pačią poziciją Hofo erdvėje.



1.19 pav. Konvertavimas iš vaizdo erdvės į Hofo erdvę [18]

Iš esmės kiekvienas tos pačios linijos taškas binariniame vaizde atitinka vieną ir tą patį tašką Hofo erdvėje. Taigi, jeigu linija egzistuoja binariniame vaizde, ji yra atvaizduojama kaip taškas (lokalus maksimumas) Hofo erdvėje (žr. 1.20 pav.). Pagal to taško poziciją galima nusakyti linijos parametrus  $\rho$  ir  $\theta$ , o pagal juos galima nubrėžti liniją originaliame vaizde remiantis (1.13) išraiška.



1.20 pav. Hofo erdvė

Toliau linija yra aptinkama balsavimo metodu. Kuo daugiau taškų sudaro vieną liniją nuotraukoje, tuo ryškesnis taškas yra matomas Hoho erdvėje, kuris apibūdina tą liniją. Hoho transformacija naudoja slenkstinės ribos parametą, kuris nusako kiek mažiausiai taškų turi sudaryti liniją, jog ją būtų galima laikyti linija. Jei tas taškas Hoho yra ryškesnis už nustatytą slenkstinę ribą, tada reiškia, jog ta linija yra aptikta vaizde.

### 1.7. Aukšto dažnio filtrai

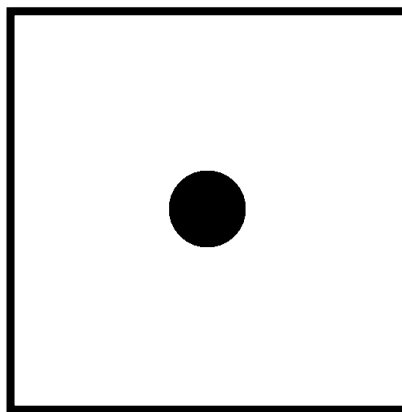
Aukšto dažnio filtras leidžia pro jį praeiti tik aukštiesiems dažniams signale. Aukšti dažniai nuotraukoje yra tos vietos, kur yra didelis skirtumas tarp gretimų pikselių intensyvumo reikšmių. Pavyzdžiui, įvairių objektų kraštai, projektuojama lazerio šviesa ar balto popieriaus lapo, gulinčio ant juodo stalo, kraštas. Žemi dažniai nuotraukoje gali būti traktuojami tiesiog kaip fonas, kadangi ten pikselių reikšmės yra tolygios. Kaip jau žinoma, kiekvienas taškas vaizdo dažnių spektre atitinka tam tikrą dažnį, o kuo to taško pozicija yra toliau nuo spektro centro, tuo aukštesnį dažnį jis apibrėžia, o tai reiškia, jog spektro centre esantys taškai apibrėžia žemus dažnius. Taigi, šių filtrų veikimo principas iš esmės yra paremtas reikšmių, esančių dažnio spektro centre, pašalinimu iš vaizdo signalo.

Toks nuotraukos dažnių srities filtravimas iš esmės yra ganėtinai optimalus metodas. Furjė transformacijos atlikimas ir jos gauto rezultato sandauga su sudarytu aukšto dažnio filtru yra kur kas, skaičiavimo požiūriu, pigesnė ir mažiau skaičiuojamosios galios reikalaujanti operacija nei, pavyzdžiui, atliekant įvairias konvoliucijos principu paremtas operacijas erdviniame domene. Taigi, toliau šiame poskyryje yra aprašyti darbe naudojami trys nuotraukų aukšto dažnio filtrai: idealusis, Gauso ir Butterworth'o.

Idealiojo aukšto dažnio filtro veikimo principas yra ganėtinai paprastas. Jis turi tik vieną parametą  $D_0$ , kuris reiškia filtro spindulį [19]. Idealusis filtras iš esmės veikia kaip paprasta slenkstinė riba nuotraukos dažnių spektre, o jo išraiška yra:

$$H(u, v) = \begin{cases} 0 & \text{if } D(u, v) \leq D_0 \\ 1 & \text{if } D(u, v) > D_0 \end{cases} \quad (1.14)$$

čia  $D(u, v)$  yra atstumas tarp esamo pikselio ir pikselio, kuris yra dažnio spektro centre,  $H(u, v)$  – atitinkamai filtro pikselių binarinės vertės. Taip suformuotas aukšto dažnio filtras iš esmės yra tiesiog binarinis vaizdas, kurio centre yra matomas juodas apskritimas (žr. 1.21 pav.), turintis pikselių vertes lygias nuliui. Filtro dydis privalo būti toks pat kaip ir filtruojamojo dažnio spektro, kad būtų galima kiekvieną filtro reikšmę sudauginti su kiekviena dažnių spektro reikšme. Tada nesunku suprasti, jog sudauginus tokį filtrą su dažnio spektru (žr. 1.16 pav. (b)), centre esančios reikšmės (žemi dažniai) tampa lygios nuliui, o visos kitos (aukšti dažniai) išlieka nepakitusios. Tokiu būdu iš nuotraukos yra pašalinami visi žemi dažniai.

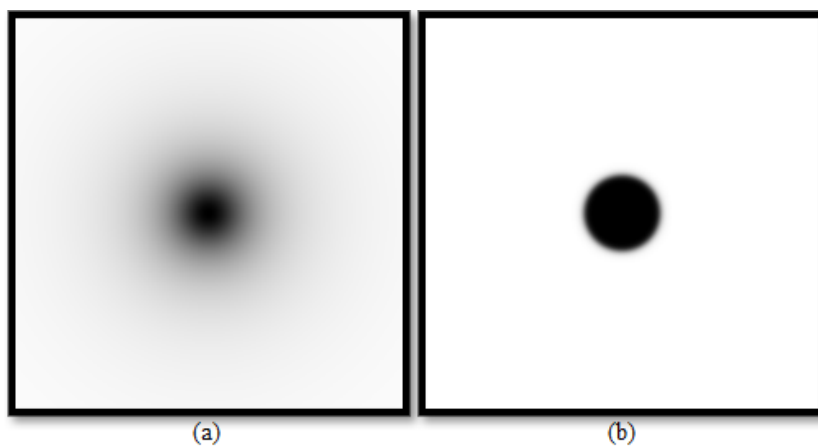


1.21 pav. Idealusis aukšto dažnio filtras

Butterworth'o aukšto dažnio filtras (žr. 1.22 pav.) neturi tokios ryškios briaunos kaip idealusis filtras, dėl to skirtumas tarp išfiltruotų ir praleistų dažnių yra tolydesnis. Jis turi platesnes derinimo galimybes, kadangi be filtro spindulio  $D_0$  egzistuoja dar vienas parametras  $n$ , kuris reiškia filtro eilę:

$$H(u, v) = \frac{1}{1 + \left(\frac{D(u, v)}{D_0}\right)^{2n}}, \quad (1.15)$$

čia  $D(u, v)$  ir  $H(u, v)$  reiškia tą patį kaip ir (1.14) išraiškoje.

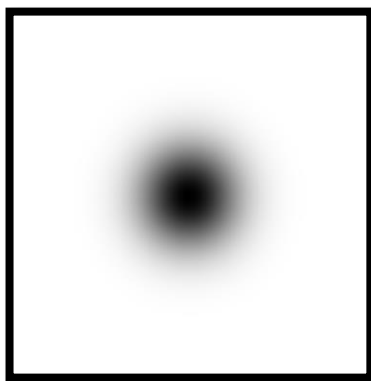


1.22 pav. Butterworth'o aukšto dažnio filtras (a)  $n = 1$  (b)  $n = 10$

Gauso aukšto dažnio filtras turi taip pat glotnų kraštą (žr. 1.23 pav.), tačiau jo funkcija yra skirtinga:

$$H(u, v) = e^{-\frac{D^2(u, v)}{2D_0^2}}, \quad (1.16)$$

čia  $D(u, v)$  ir  $H(u, v)$  reiškia tą patį kaip ir (1.14) išraiškoje.



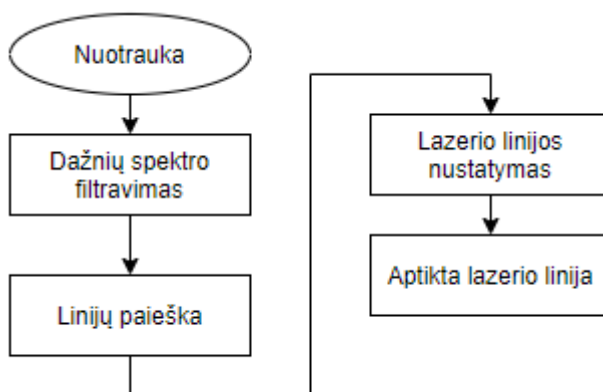
1.23 pav. Gauso aukšto dažnio filtras

Taigi, visi trys filtrai turi skirtingas funkcijas, todėl naudojant tokio pačio ilgio spindulį visi filtrai veikia skirtingai. Idealusis filtras nustato konkrečią ribą, iki kurios žemi dažniai yra filtruojami, dėl to išfiltruotame vaizde gali būti sunku aptikti objektų kraštus, nes nėra kartu filtruojamas perėjimas iš aukšto dažnio į žemą. Kita vertus Gauso ir Butterworth'o filtrai turi glotnų kraštą, dėl to jais išfiltruoti vaizdai turi aštresnes briaunas, nes dažniai yra filtruojami sulig briaunų perėjimais vaizde.



## 2. Lazero linijos aptikimo metodologija

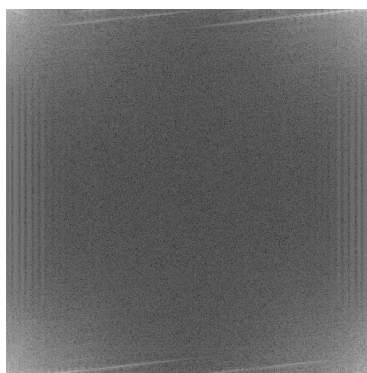
Šiame skyriuje yra pasiūlyta nauja lazero linijos aptikimo metodologija, kuri iš esmės susideda iš trijų pagrindinių dalių: dažnių spektro filtravimo, linijų paieškos ir lazero linijos nustatymo (žr. 2.1 pav.). Toliau esančiuose poskyriuose kiekviena iš šių dalių yra analizuojama atskirai..



2.1 pav. Algoritmo struktūrinė schema

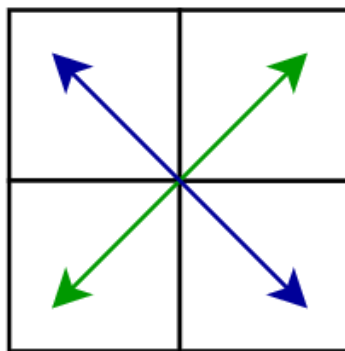
### 2.1. Dažnių spektro filtravimas

Šis poskyris yra skirtas detalesnei vaizdo dažnių spektro filtravimo apžvalgai. Atlikus Furjė transformaciją yra gaunamas vaizdas, kuriame yra atvaizduotas originalios nuotraukos dažnių spektras (žr. 2.2 pav.). Originali šio vaizdo forma dažnius atvaizduoja kitaip – žemi dažniai yra koncentruoti nuotraukos kampuose.



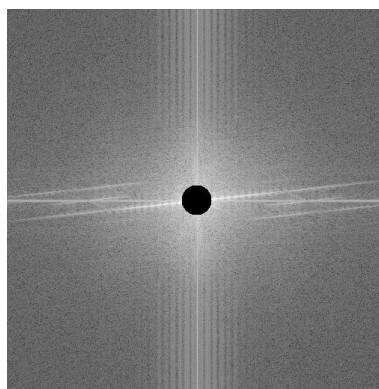
2.2 pav. Pradinė dažnių spektro forma

Filtruoti tokį vaizdą yra sudėtinga, nes žemi dažniai yra išmėtyti skirtingose vaizdo vietose. Tam, kad filtravimas būtų paprastesnis ir greitesnis įstrižai yra sukeičiami dažnių spektro ketvirčiai (žr. 2.3 pav.). Tada visi žemi dažniai tampa koncentruoti vienoje vietoje – spektro centre. Po tokio ketvirčių sukeitimo dažnių reikšmės tampa pasiskirsčiusios pagal dėsnį – kuo reikšmė yra toliau nuo spektro centro, tuo aukštesnį dažnį ji apibūdina.



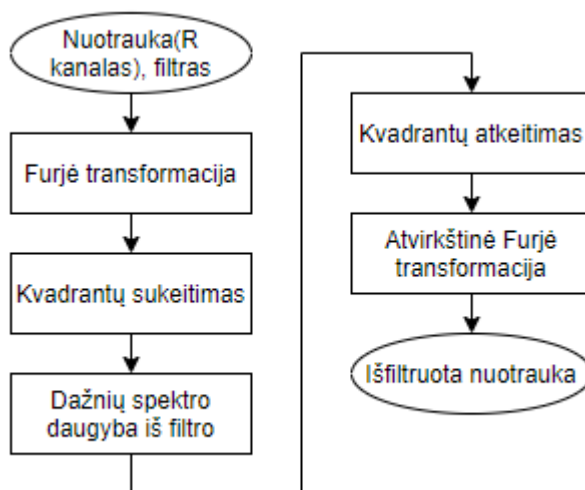
2.3 pav. Dažnių spektro ketvirčių sukeitimas

Toliau vyksta iš esmės labai paprastas filtravimo procesas. Kadangi žemi dažniai yra koncentruoti spektro centre, dabar yra galimas aukšto dažnio filtro taikymas (žr. 2.4 pav.). Turimo filtro ir dažnių spektro dydžiai turi būti vienodi, tam kad būtų galima kiekvieną spektro reikšmę padauginti iš kiekvienos filtro pikselio reikšmės (žr. 2.4 pav.). Tokia dviejų matricų daugybos operacija yra greita, optimali ir nereikalaujanti daug skaičiuojamosios galios.



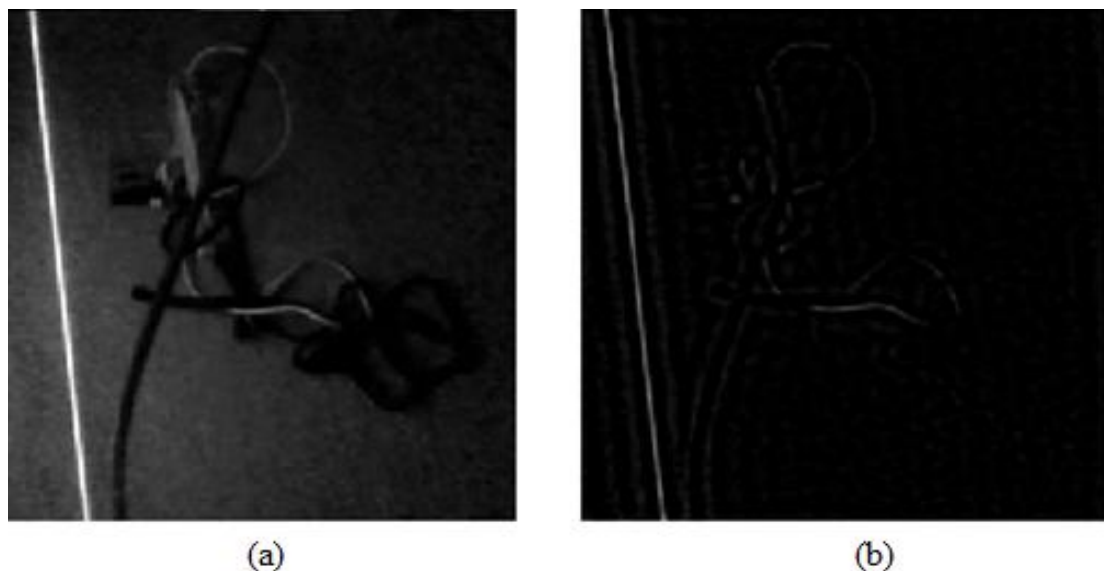
2.4 pav. Dažnių spektras su aukšto dažnio filtru

Kadangi, šiuo atveju, filtro viduryje reikšmės yra lygios nuliui arba artimos nuliui, atlikus daugybos veiksmą, žemų dažnių reikšmės taip pat yra paverčiamos į nulį arba skaičių, artimą nuliui. Tada dažnių spektras yra konvertuojamas atgal į erdvinį domeną (žr. 2.5 pav.).



2.5 pav. Dažnių spektro filtravimo procesas

Šiame darbe yra ieškoma raudona lazerio linija vaizde, dėl to filtravimo proceso įvestis yra raudonasis nuotraukos kanalas (žr. 2.6 pav. (a)) bei iš anksto suformuotas aukšto dažnio filtras su nustatytu spinduliu. Po to Furjė transformacijos pagalba yra suskaičiuojamas vaizdo dažnių spektras, kurio ketvirčiai yra sukeičiami įstrižai. Tada spektras yra padauginamas iš turimo filtro, o spektro ketvirčiai atkeičiami atgal į pradinę poziciją. Ketvirčių atkeitimas yra svarbus žingsnis prieš pradedant konversiją į erdvinį domeną, nes dažnių pozicijos spektre buvo pakeistos. Pasitelkus atvirkštinę Furjė transformaciją vaizdas yra atgal konvertuojamas į erdvinį domeną (žr. 2.6 pav. (b)).



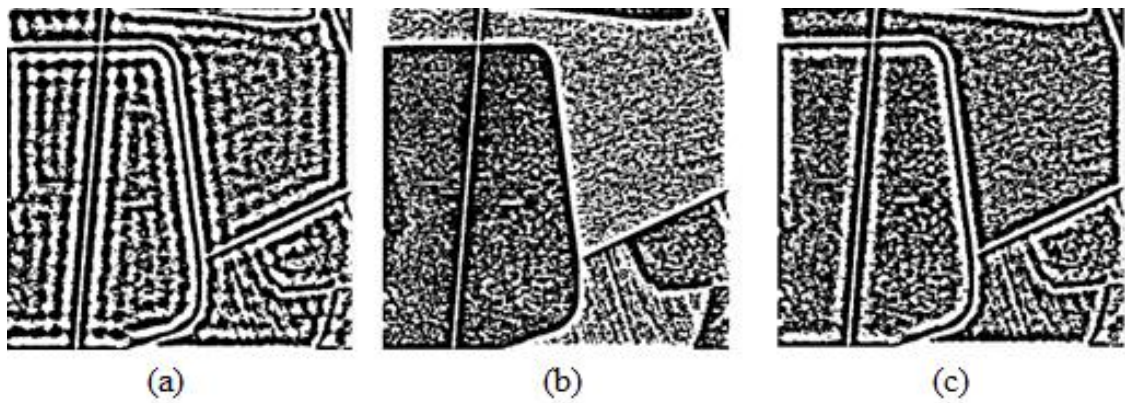
2.6 pav. Dažnių filtravimo pavyzdys (a) vaizdas prieš filtravimą (b) vaizdas po filtravimo

Pateiktame pavyzdyje (žr. 2.6 pav.) yra pavaizduotas dažnių spektro filtravimo rezultatas. Vaizde prieš filtravimą yra matoma ryški linija kairėje pusėje. Tai yra raudona lazerio linija, tačiau kadangi filtravimui yra naudojamas vien raudonasis kanalas, visas vaizdas yra nespalvotas. Toks ryškus objektas vaizde sukelia aukštus dažnius spektre, dėl to atlikus filtravimą yra pašalinama daug nereikalingo triukšmo. Matoma, jog iš nuotraukos yra pašalintas fonas, tačiau linija išlieka taip pat ryški.

### 2.1.1. Aukšto dažnio filtrų tyrimas

Lazerio linijos aptikimo kokybė yra tiesiogiai susijusi su priešlaikinio apdorojimo metu vykdomu dažnio spektro filtravimu. Skirtingi filtrai gražina skirtingus rezultatus, dėl to yra labai svarbu nustatyti, kuris filtras yra tinkamiausias lazerio linijos aptikimo uždaviniui spręsti.

Vizualiai įvertinus trijų aukšto dažnio filtrų rezultatus (žr. 2.7 pav.) galima pastebėti, jog nepaisant vienodo filtrų dydžio, kiekvieno jų filtravimo rezultatas yra skirtingas. Pagrindinis tokio filtravimo tikslas yra pašalinti iš vaizdo kiek įmanoma daugiau triukšmo paliekant tik reikalingą informaciją. Šiuo atveju reikalinga informacija yra visi pikseliai, kurie priklauso lazerio linijai, o triukšmas – visas likęs fonas, iš kurio bandoma išskirti liniją. Taigi, toliau šiame skyrelyje yra aprašomas bandymas, kurio metu nustatoma, kuris filtras yra tinkamiausias lazerio linijos aptikimui.



2.7 pav. Dažnio spektro filtravimo rezultatai su skirtingais filtrais (a) idealusis filtras (b) Gauso filtras (c) Butterworth'o filtras

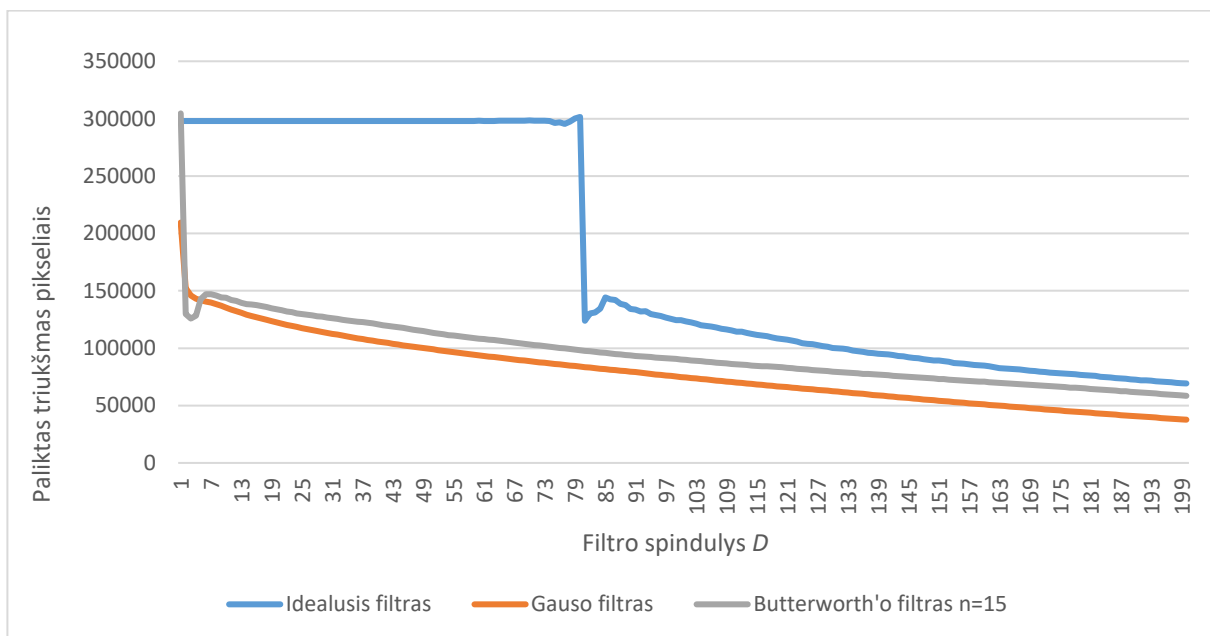
Bandymui yra parenkamos trys eksperimentinės nuotraukos, kuriose yra matoma lazerio linija. Kiekvienoje jų lazerio linija yra pažymima ryškiai žalia spalva (žr. 2.8 pav.). Žymėjimo pikselių RGB reikšmės yra  $(0, 255, 0)$ . Prieš žymėjimą yra patikrinama ar nuotraukoje nėra daugiau pikselių, kurie turi tokias pat reikšmes. Taip yra užtikrinami kiek įmanoma tikslesni bandymų rezultatai.



2.8 pav. Eksperimentinės nuotraukos pavyzdys (a) originali nuotrauka (b) pažymėta nuotrauka

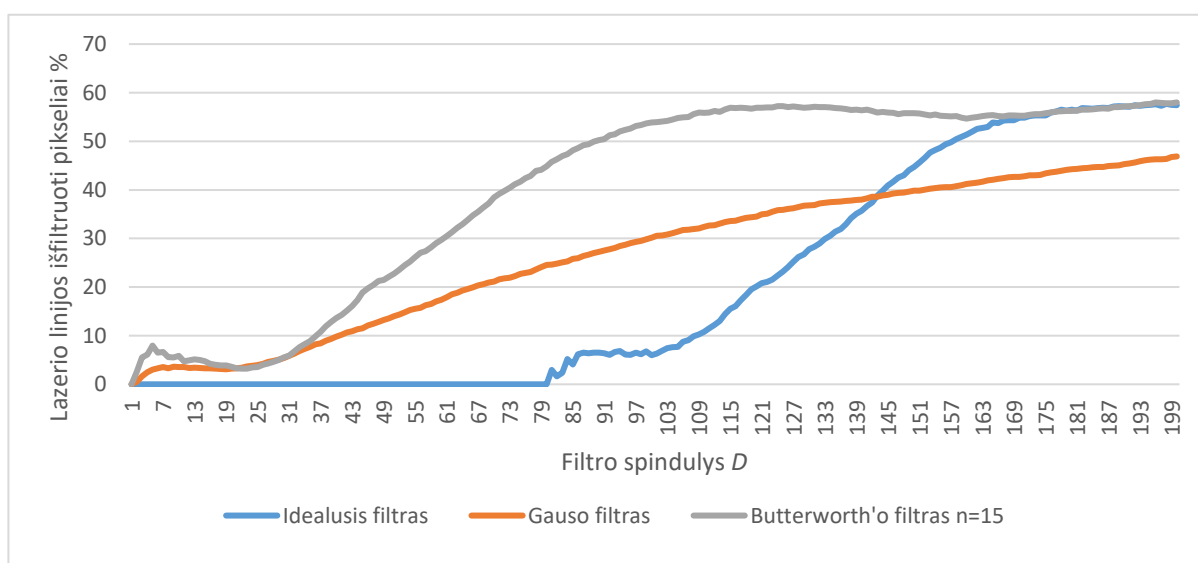
Visi pikseliai, kurie yra pažymėti visiškai ryškiai žalia spalva yra laikomi lazerio linijai priklausančiais pikseliais, o visi kiti – nereikalingas triukšmas. Remiantis tuo yra atliekami bandymai su kiekviena iš eksperimentinių nuotraukų. Kiekvienos jų dažnių spektras yra filtruojamas su skirtingais aukšto dažnio filtrais (idealiuoju, Gauso ir Butterworth'o). Su kiekvienu filtru ir kiekviena nuotrauka yra atliekama po 200 iteracijų, o kiekvienos jų metu filtro spindulys  $D$  yra didinamas vienetu. Butterworth'o filtrui yra nustatomas vis skirtingas statinis eilės numeris  $n$  kiekvienai nuotraukai. Po kiekvienos iteracijos yra skaičiuojama, kiek procentais filtras išfiltravo reikalingos informacijos (pikselių, kurie priklauso lazerio linijai) ir kiek vaizde yra palikta triukšmo pikseliais (visi kiti pikseliai, kurie nepriklauso lazerio linijai).

Remiantis pateiktu grafiku (žr. 2.9 pav.) galima teigti, jog Gauso filtras, lyginant su idealiuoju ir Butterworth'o filtrais, vaizde palieka mažiausiai triukšmo nepaisant esamo filtro dydžio. Tačiau esant labai mažam filtro spinduliui (apie 5) grafike galima pastebėti, jog Butterworth'o filtras palieka mažiau triukšmo nei Gauso filtras. Visgi tokie mažų dydžių filtrai šiame darbe yra nenaudojami, dėl to į tai galima nekreipti dėmesio ir vadovautis vien matomu dėsningumu.



2.9 pav. Paliktas triukšmas pikseliais (pirma nuotrauka)

Analizuojant grafiką (žr. 2.10 pav.) galima pastebėti, jog Gauso ir idealiojo filtrų grafikai susikerta ties tašku, kur filtro spindulys  $D$  yra lygus 144. Esant mažesniai spinduliui atrodo, jog idealusis filtras pašalina mažiausią dalį lazerio linijos duomenų, tačiau ties didesniu spinduliu Gauso filtras tampa geresniu pasirinkimu. Taip pat verta atkreipti dėmesį į tai, jog Gauso filtro grafikas yra pats tolydžiausias, dėl to darosi paprasčiau nustatyti jo poveikį visai sistemai, nes priklausomybė atrodo beveik tiesinė.



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

Iš esmės visų bandymų metu buvo gauti panašūs rezultatai, kaip ir pateiktuose grafikuose. Dėl to siekiant gauti viso bandymo įvertinimą, kiekvienos nuotraukos gautų rezultatų vidurkiai yra pateikti lentelėje (žr. 2.1 lentelė).

2.1 lentelė Visų bandymų rezultatų vidurkiai

	Idealusis filtras	Gauso filtras	Butterworth'o 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. Jis vidutiniškai vaizde palieka apie 58 tūkstančius pikselių triukšmo, lyginant su kitais filtrais – 77 ir 64 tūkstančiai. Gauso filtras taip pat daro mažiausią įtaką lazerio linijos pikseliams. Jis vidutiniškai išfiltruoja tik apie 11% pikselių, kurie priklauso lazerio linijai, lyginant su kitais filtrais – 35% ir 30%.

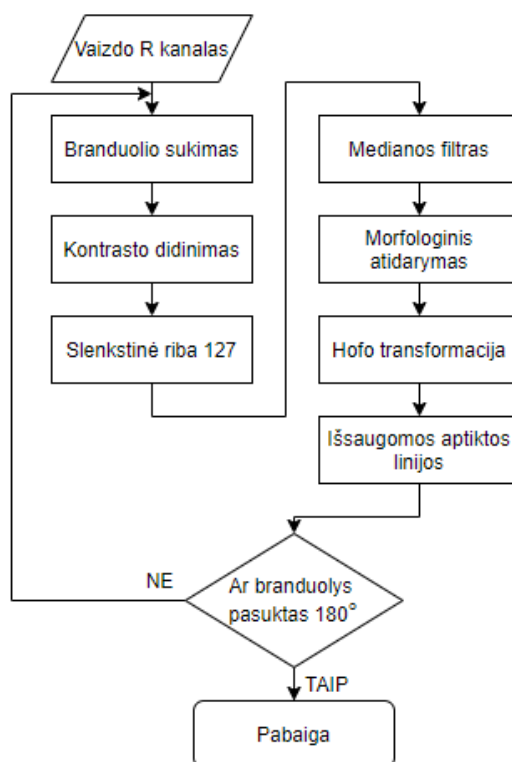
Remiantis gautais bandymų rezultatais yra nuspręsta, jog optimaliausias filtras šiam darbui yra Gauso filtras. Jo poveikis sistemai yra lengvai nuspėjamas ir derinimas yra pakankamai paprastas, nes yra tik vienas parametras – filtro spindulys  $D$ . Kai tuo tarpu Butterworth'o filtras turi papildomą parametą  $n$  (filtro eilės numeris). Taigi, toliau šiame darbe naudojamo aukšto dažnio Gauso filtro generavimo kodas pateiktas (žr. 2.11 pav.). Čia *sqrt* ir *exp* funkcijos atitinkamai reiškia šaknies ir eksponentės funkcijas.

```
filter[height, width]
center_x = width/2
center_y = height/2
for (i=0; i<width; i++)
    for (j=0; j<height; j++)
        distance=sqrt((center_x - i)^2 + (center_y - j)^2)
        filter[j,i] = 1 - exp(-distance^2/(2*rad^2))
    end
end
end
```

2.11 pav. Gauso filtro generavimo kodas



## 2.2. Linijų paieška



2.12 pav. Linijų paieškos algoritmas

Šiame poskyryje yra aprašomas naudojamas algoritmas linijų paieškai. Jis iš esmės yra paremtas vaizdo konvoliucijos branduolio sukimu ir Hofo transformacija (žr. 2.12 pav.). Vienas kadras yra apdorojamas vis su skirtingu kampu pasuktu branduoliu. Šios iteracijos vyksta tol, kol branduolys galiausiai yra apsuktamas  $180^\circ$ . Atlikus konvoliuciją su pasuktu branduoliu, po to yra atliekamas kontrasto didinimas ir formuojamas binarinis vaizdas paprastos slenkstinės ribos metodu. Šiuo atveju yra nustatyta slenkstinė riba 127. Tai yra tiesiog pusė maksimalaus galimos pikselio intensyvumo reikšmės – 255. Daroma prielaida, jog vaizde lazerio linija turi būti pakankamai ryškiai matoma.

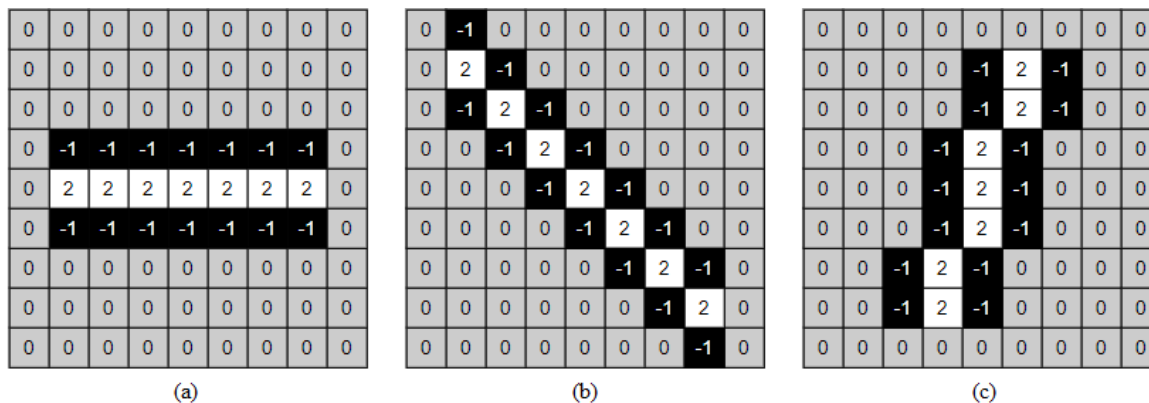
Toliau vyksta binarinio vaizdo apdorojimas. Siekiant pašalinti triukšmą iš vaizdo yra naudojami medianos filtras ir morfologinis atidarymas. Galiausiai toks binarinis vaizdas yra apdorojamas Hofo transformacija, kuri aptinka visas linijas vaizde ir išsaugo jų tam tikrus parametrus tolimesniam algoritmo veikimui.

### 2.2.1. Branduolio sukimas

Linijų paieška pradedama vykdyti nuo vaizdo, kuris yra išfiltruotas su Gauso aukšto dažnio filtru. Po to yra vykdomas to vaizdo filtravimas konvoliucijos principu pagal iš anksto sudarytą linijos aptikimo branduolį. Branduolio dydis yra 1% turimo vaizdo skersmens. Šis dydis yra suapvalinamas iki artimiausio nelyginio skaičiaus. Branduolio dydis privalo būti nelyginis skaičius dėl paprastesnio algoritmo įgyvendinimo. Tokiu būdu, nesvarbu kokių kampu yra pasuktas branduolys, jo centriniame pikselyje visada bus linijos centras.

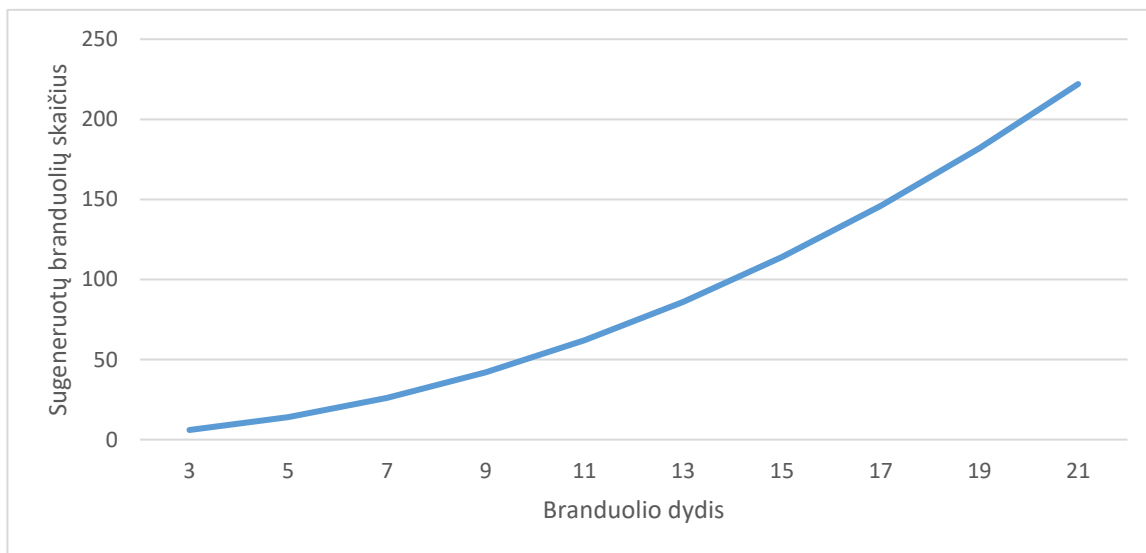
Branduolio sukimas vyksta paprastu principu. Iš pradžių yra sudaromas vertikalus branduolys (žr. 2.13 pav. (a)), po to yra generuojami visi įmanomi branduolio pikselių išsidėstymo variantai taip, kad

pikseliai turintys reikšmę 2, vertikaliai, horizontaliai arba skersai liestūsi su kitu pikseliu, turinčiu reikšmę 2. Atitinkamai kiekvienas dvejetas turi ir po du pikselius, kurių reikšmė lygi (-1). Branduolio sukimo kampas yra skaičiuojamas nuo horizontalios padėties pagal laikrodžio rodyklę. Visi tokiu būdu sugeneruoti branduoliai yra saugomi programos vidinėje atmintyje.



2.13 pav. Pasukti branduoliai (a) 0° kampu (b) 45° kampu (c) 105° kampu

Sugeneruotų branduolių skaičius yra tiesiogiai priklausomas nuo pačio branduolio dydžio (žr. 2.14 pav.), kadangi didesnis branduolys turi daugiau pikselių, o tai reiškia, jog egzistuoja daugiau įmanomų skirtingų pikselių išsidėstymo variantų. Esant dideliame branduolio dydžiui yra sugeneruojama sąlyginai daug pasuktų branduolių, pavyzdžiui, jei nustatomas branduolio dydis yra 21, tai algoritmas sugeneruoja 222 skirtingų kampų branduolius. Ilgainiui vis didinant branduolio dydį darosi labai neoptimalu atlikti konvoliucijos operaciją su kiekvienu iš branduolių.

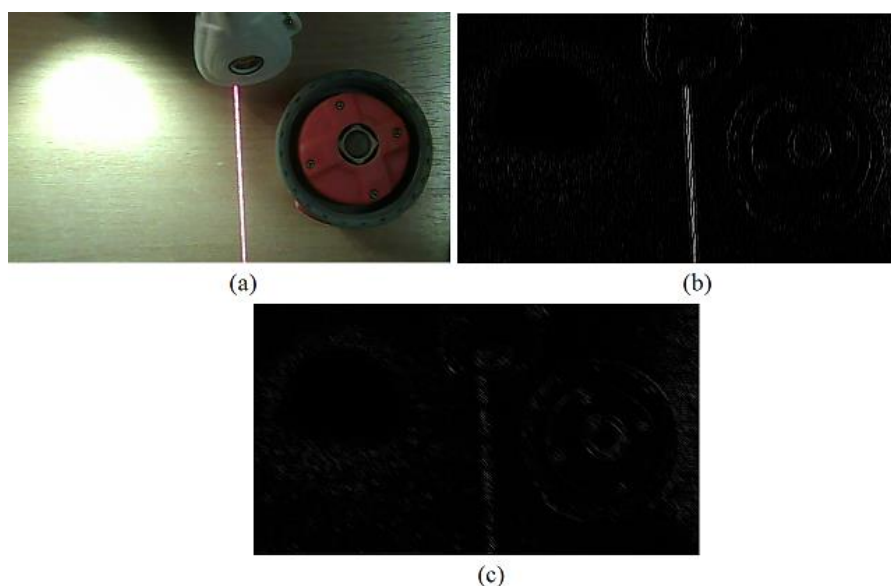


2.14 pav. Sugeneruotų branduolių skaičiaus priklausomybė nuo branduolio dydžio

Branduolio dydis proporcingai didėja nuo vaizdo dydžio, esant dideliame branduoliui tenka atlikti daug konvoliucijos operacijų. Dėl to algoritmas pašalina kai kuriuos sugeneruotus tarpinius branduolius, jeigu jų yra sugeneruota daugiau nei 45. Pavyzdžiui, jeigu branduolio dydis yra parenkamas 13, tada yra sugeneruojami 86 pasukti branduoliai. Tokiu atveju galima iš branduolių rinkinio išmesti kas antrą branduolį, tada jų liks 43. Šiam algoritmui visiškai pakanka tokios



branduolio kampo sukimo rezoliucijos, nes toliau šiame darbe naudojama Hoho transformacija naudoja tik  $1^\circ$  kampo rezoliuciją.

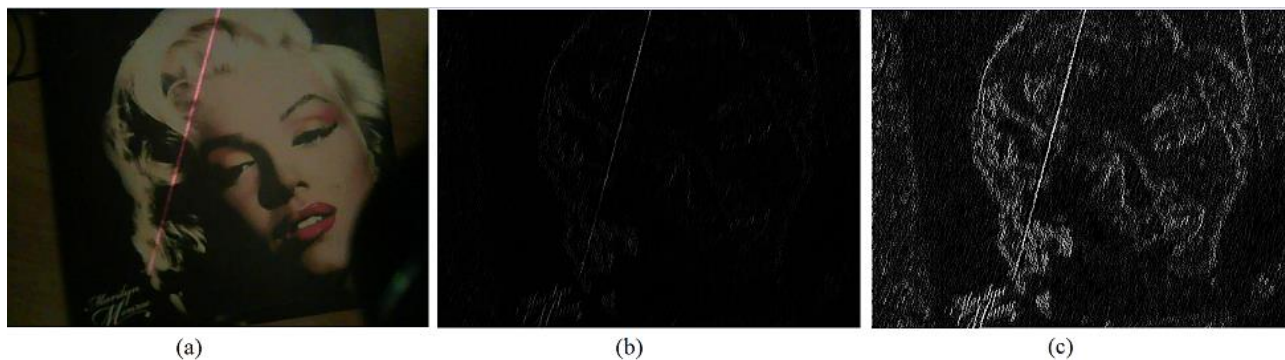


2.15 pav. Filtravimas konvoliucijos principu (a) originali nuotrauka (b) filtravimas su  $90^\circ$  kampu pasuktu branduoliu (c) filtravimas su  $45^\circ$  kampu pasuktu branduoliu

Taigi, šiame žingsnyje algoritmas iš esmės naudoja paprasčiausią linijos aptikimo branduolį, pasuktą įvairiais kampais. Taip yra daroma, nes iš anksto nėra žinoma, kokia yra lazerio linijos vaizde orientacija, o išbandžius visus branduolius visais kampais, mažiausiai su vienu branduoliu tikrai bus gautas pakankamai ryškus rezultatas (žr. 2.15 pav. (b)). Jeigu branduolio kampas yra panašus į lazerio linijos kampą, linija po konvoliucijos išryškėja, kitais atvejais, ji tampa blanki ir tolesniuose algoritmo žingsniuose ji nėra aptinkama (žr. 2.15 pav. (c)).

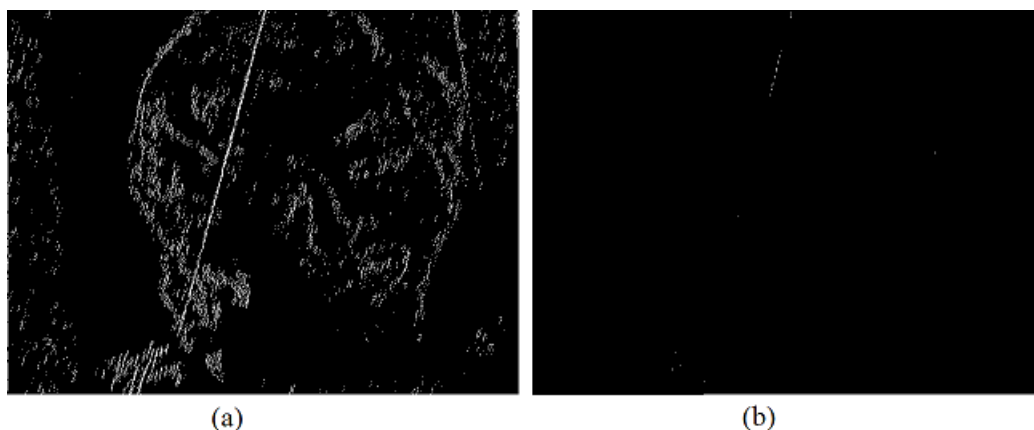
### 2.2.2. Binarinio vaizdo formavimas

Atlikus konvoliuciją su vienu iš sugeneruotų branduolių toliau yra formuojamas binarinis vaizdas. Šiame žingsnyje labai svarbus yra išfiltruoto vaizdo kontrasto didinimas. Būtent jis suteikia visam algoritmui daugiau lankstumo ir dinamiškumo. Tam tikrais atvejais, kai lazerio linija nėra labai ryškiai atspindima (žr. 2.16 pav. (a)), gali būti sunku suformuoti binarinį vaizdą neprarandant lazerio linijos.



2.16 pav. Kontrasto didinimas (a) originali nuotrauka (b) konvoliucijos principu išfiltruota nuotrauka (c) nuotrauka su padidintu kontrastu

Kontrasto didinimas padeda ne tik aptikti silpniau atspindimą lazerio liniją, bet ir suteikia galimybę naudoti paprastą slenkstinę ribą neprarandant reikalingų duomenų (žr. 2.17 pav.). Jeigu vaizdas su padidintu kontrastu turi matomą lazerio liniją, galima nesudėtingai išfiltruoti visus žemesnio intensyvumo pikselius. Šiuo atveju yra nustatoma statinė slenkstinė riba 127. Tai taip pat supaprastina algoritmo veikimo principą, nes nebereikia slenkstinės ribos derinti atskirai prie kiekvieno vaizdo. Tokiu būdu yra gaunamas patenkinamas filtravimo rezultatas nepaisant kaip stipriai yra atspindima lazerio linija.

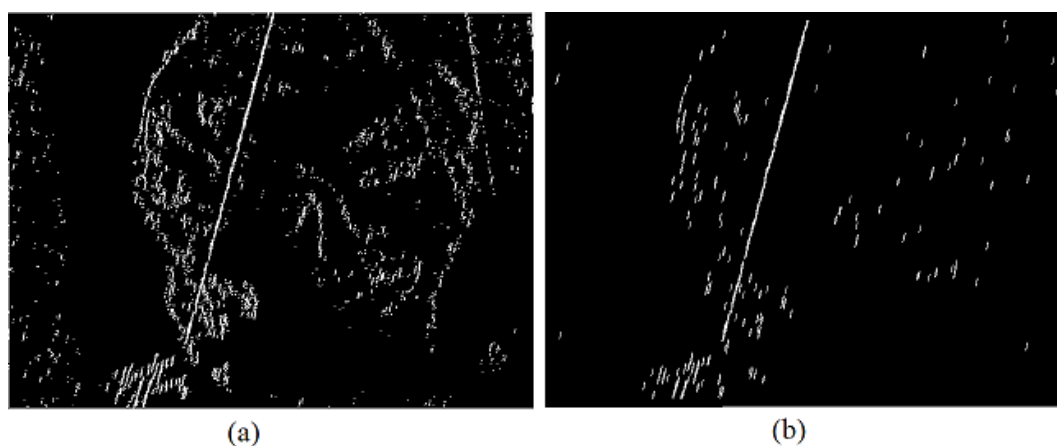


2.17 pav. Binarinio vaizdo pagal statinę slenkstinę ribą (a) su kontrasto didinimu (b) be kontrasto didinimo

Padidinus kontrastą ir panaudojus paprastą slenkstinę ribą yra išsaugoma didelė dalis reikalingų duomenų. Žinoma, toks kontrasto koregavimas turi ir neigiamą efektą – vaizde taip pat yra paliekama kur kas daugiau nereikalingo triukšmo. Dėl to norint jį pašalinti toliau yra atliekamos morfologinės operacijos bei papildomas binarinio vaizdo filtravimas, kuris yra aprašytas kitame skyrelyje.

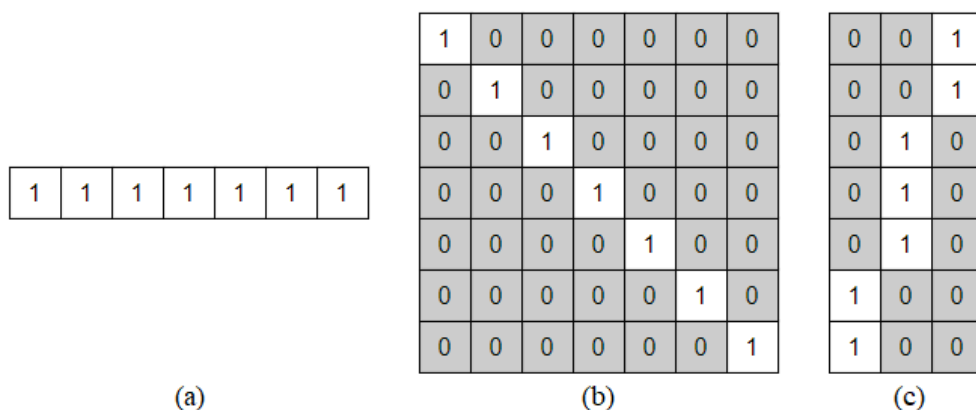
### 2.2.3. Binarinio vaizdo apdorojimas

Toliau siekiant pašalinti kiek įmanoma daugiau triukšmo iš nuotraukos yra atliekamas binarinio vaizdo filtravimas pasitelkiant medianos filtrą. Jis yra veiksmingas naudojant mažų taškų pašalinimui, tačiau jo dydis turi labai stiprų poveikį galutiniam filtravimo vaizdui. Jeigu medianos filtro dydis yra parenkamas per didelis, gali būti išfiltruota visiškai visa lazerio linija. Dėl to šiam darbe yra naudojamas minimalaus dydžio (2x2) medianos filtras, kuris mažai įtakoja lazerio liniją, tačiau pašalina dalį nereikalingo triukšmo (žr. 2.18 pav.). Vaizdas prieš pritaikant medianos filtrą (žr. 2.17 pav.).



2.18 pav. Binarinio vaizdo filtravimas (a) medianos filtras (b) morfologinis atidarymas

Po filtravimo su medianos filtru yra atliekamas morfologinis atidarymas. Tai yra morfologinės erozijos ir morfologinio išplėtimo operacijų kombinacija pagal tą patį struktūrinį elementą. Struktūrinis elementas yra generuojamas ties kiekviena konvoliucijos branduolio iteracija pagal posūkio kampą (žr. 2.19 pav.). Iš esmės struktūrinis elementas yra sudaromas pagal tą patį branduolį iš jo tiesiog pašalinus neigiamas reikšmes ir eilutes bei stulpelius, kuriuose yra tik nuliai. Tokiu būdu struktūrinis elementas turi tokį patį kampą kaip ir naudojamas branduolys.



2.19 pav. Morfologinio atidarymo struktūriniai elementai (a) 0° (b) 45° (c) 105°

Morfologinis atidarymas iš esmės binariniame vaizde pašalina visus mažus objektus, tačiau išsaugo didelių objektų formą ir dydį pagal struktūrinį elementą. Taigi, pavyzdžiui, jeigu vaizde yra 105° kampu pasukta lazerio linija (žr. 2.18 pav. (a)) ir tas vaizdas yra apdorojamas su tokiu pat kampu pasuktu konvoliucijos branduoliu (žr. 2.13 pav. (c)), o po to atliekamas morfologinis atidarymas pagal tokio pat kampo struktūrinį elementą (žr. 2.19 pav. (c)), galiausiai yra gaunamas binarinis vaizdas su ryškiai matoma linija (žr. 2.18 pav. (c)). Kadangi struktūrinio elemento kampas sutampa su linijos kampu, linija vaizde yra išsaugoma, o didelė dalis triukšmo yra nufiltruojama. Toliau toks binarinis vaizdas yra apdorojamas Hofo transformacija, apie tai plačiau yra rašoma kitame skyrelyje.

#### 2.2.4. Linijų aptikimas ir saugojimas binariniame vaizde

Hofa transformacija yra esminis šio algoritmo žingsnis. Apdorotame ir išfiltruotame binariniame vaizde toliau yra ieškoma taškų, kurie sudaro linijas. Kaip jau žinoma, atlikus Hofa transformaciją yra apskaičiuojama Hofa erdvė, kurioje lokalūs maksimumai reiškia egzistuojančias linijas binariniame vaizde.

Toliau linijų aptikimas Hofa erdvėje vyksta pagal pagrindinius 2 parametrus: minimalus linijos ilgis ir maksimalus linijos pertrūkimo ilgis. Minimalus linijos ilgis reiškia koks turi būti linijos ilgis, jog ją būtų galima laikyti linija. Jeigu linija turi pertrūkimų, tai maksimalus pertrūkimo ilgis reiškia, jeigu pertrūkimo ilgis neviršija nustatytos ribos, tai pertrūkimas yra ignoruojamas ir atskirti linijos segmentai yra laikomi kaip viena linija. Šiuo atveju šie du parametrai taip pat priklauso nuo vaizdo skersmens. Minimalus linijos ilgis yra 20% vaizdo skersmens, o maksimalus pertrūkimo ilgis - 5%.

Kiekviena aptikta linija ties kiekviena branduolio sukimo iteracija yra saugoma masyve. Iš pradžių yra apskaičiuojamas linijos kampas pagal du jos galų taškus (žr. 2.20 pav.). Svarbu atkreipti dėmesį į tai, jog kampas yra skaičiuojamas nuo horizontalios padėties pagal laikrodžio rodyklę. Tokia taisyklė yra priimta visame algoritme.

```

function points_to_angle(x1, x2, y1, y2)
    slope = (y2 - y1) / (x2 - x1)
    slope_angle = arctan(slope)

    if slope_angle < 0:
        slope_angle = 180 - slope_angle

    return slope_angle

```

2.20 pav. Linijos kampo apskaičiavimo kodas

Apskaičiavus aptiktos linijos kampą yra ieškomas tam kampui statmenas konvoliucijos branduolys (žr. 2.21 pav.). Kadangi visi pasukti branduoliai yra sugeneruoti dar algoritmo pradžioje, iš visų brandulių masyvo belieka išsirinkti reikiamą branduolį pagal kampą.

```

function degrees_to_kernel(kernels, kernels_count, degrees)
    //How many degrees is rotated from horizontal position
    if degrees > 180:
        degrees = degrees - 180

    rotation_resolution = 180/kernels_count

    kernel_deg = 0
    kernel_deg_before = 0
    kernel_before = kernels[0]
    for each kernel in kernels:
        if degrees > kernel_deg:
            //Check which kernel is closer to provided degrees
            if |degrees - kernel_deg_before| < |degrees - kernel_deg|:
                return kernel_before
            else
                return kernel
            if
        kernel_deg_before = kernel_deg
        kernel_before = kernel
        kernel_deg = kernel_deg + rotation_resolution

```

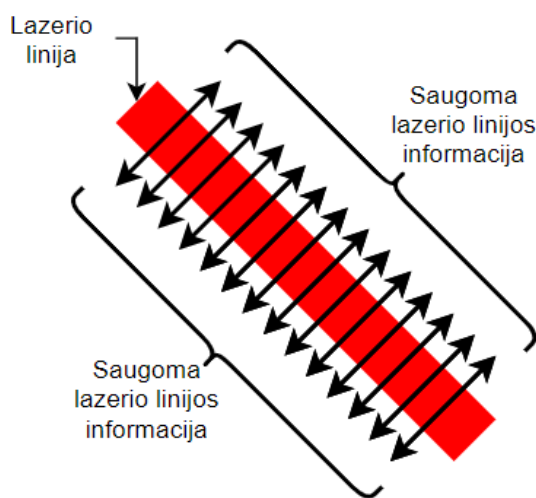
2.21 pav. Branduolio parinkimo pagal kampą kodas

Kaip jau žinoma, sugeneruotų branduolių skaičius priklauso nuo pačio branduolio dydžio. Kuo didesnis branduolys, tuo daugiau yra sugeneruojama pasuktų branduolių ir gaunama didesne branduolio kampo sukimo rezoliucija. Kitaip tariant, tai yra tam tikras kampo kvantavimo vienetas, kuris reiškia kaip tiksliai sugeneruoti branduoliai apibūdina posūkio kampą:

$$r = \frac{180}{n}, \quad (2.1)$$

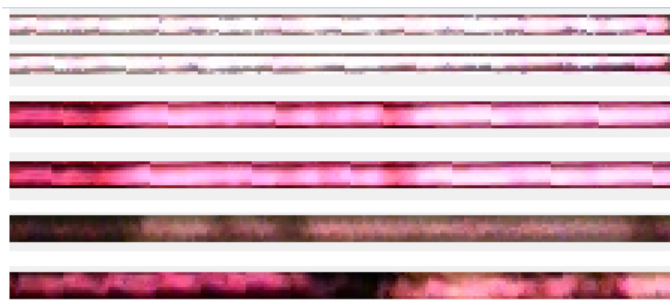
čia  $r$  yra vieno branduolio kvantavimo vienetas laipsniais, o  $n$  – sugeneruotų branduolių skaičius. Iš esmės, kuo daugiau turima branduolių, tuo mažesnis yra gaunamas  $r$  ir dėl to kiekvienas branduolys tiksliau apibūdina kampą.

Iš esmės aptikta linija binariniame vaizde yra toje pačioje pozicijoje kaip ir originaliame RGB vaizde. Dėl to toliau pagal linijos poziciją binariniame vaizde yra surenkami visi tos linijos RGB pikseliai iš originalaus vaizdo. Suradus statmeną linijai branduolį, pagal jį yra formuojamas lygiai tokio pat kampo struktūrinis elementas, kuris buvo naudojamas ir morfologinei atidarymo operacijai. Tada šie struktūriniai elementai yra dedami ant originalaus vaizdo per visą linijos ilgį taip, jog linijos centras sutaptų su struktūrinio elemento centru (žr. 2.22 pav.). Tada masyve yra išsaugomi originalaus RGB vaizdo pikseliai, kurių pozicija sutampa su struktūrinio elemento binariniais vienetais. Kitaip tariant, yra išsaugomas visas originalus linijos vaizdas (žr. 2.23 pav.).



2.22 pav. Lazerio linijos išsaugojimas

Toks linijos saugojimo būdas yra geras tuo, jog yra nesvarbus linijos posūkio kampas vaizde. Optimalu yra ir tai, jog iš anksto sugeneruoti branduoliai yra naudojami ne vieną kartą per visą algoritmą, o branduolio dydį visada galima pakeisti ir taip priderinti prie lazerio linijos pločio. Žinoma, tokiu būdu saugant linijas yra neišvengiama ir tai, jog kartu su linija bus išsaugoma ir dalis fono, kuris yra tik nereikalingas triukšmas, tačiau to yra sudėtinga išvengti, nes linijos plotis nėra iš anksto žinomas. Taigi, tolimesni algoritmo žingsniai yra skirti būtent išsaugotų linijų analizei. Tarp visų jų dažnai pasitaiko ir kitų linijų, kurios nėra lazerio linijos, tačiau jei lazerio linija vaizde yra ryškiai matoma, tarp išsaugotų linijų bus ir ieškoma lazerio linija.



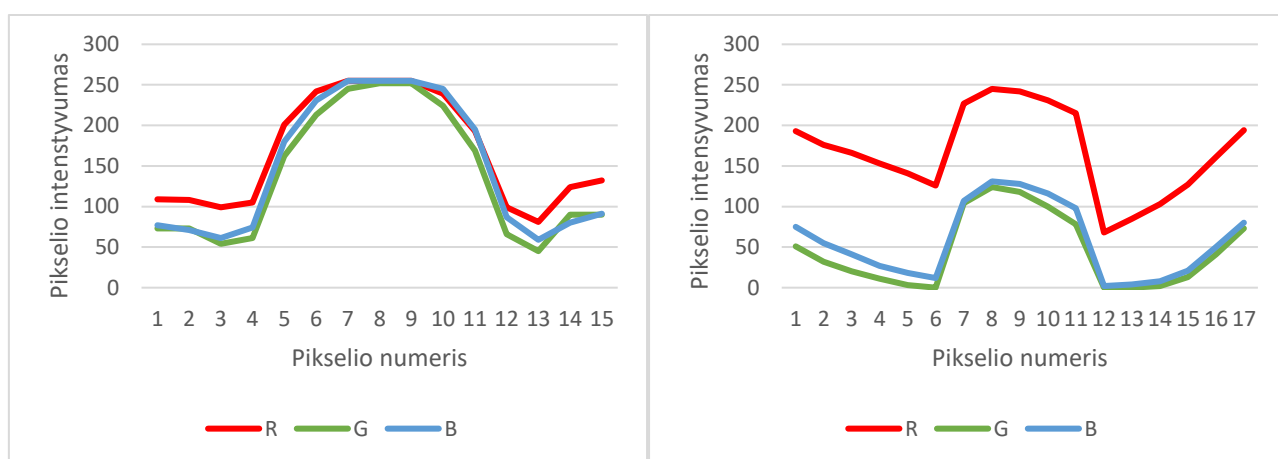
2.23 pav. Išsaugotų lazerio linijų pavyzdžiai

### 2.3. Lazero linijos nustatymas

Baigus visas iteracijas su pasuktais konvoliucijos branduoliais toliau yra analizuojama kiekviena iteracijų metu aptikta linija. Tada yra daromos iteracijos su kiekviena aptikta ir programos vidinėje atmintyje išsaugota linija. Pagrindinis šio žingsnio tikslas yra nustatyti, kuri iš visų aptiktų linijų yra būtent raudona lazerio linija.

Šiuo atveju galima pastebėti du skirtingus lazerio linijos projektavimo ant paviršiaus atvejus. Pirmasis, kai lazerio linija nuo paviršiaus yra stipriai atspindima, tada ji per vidurį tampa balta. Šis reiškinys yra vadinamas kameros persotinimu (angl. *oversturation*) ir tai nutinka dėl nepakankamo kameros jautrumo. Antrasis atvejis yra tada, kai raudona lazerio linijos ir kameros vaizde išlieka raudona. Abu šiuos atvejus galima pastebėti (žr. 2.23 pav.).

Iš esmės lazerio linijos nustatymas yra paremtas RGB vaizdo skirtingų kanalų tarpusavio santykiu. Jeigu kamera nėra persotinta, tada per visą lazerio linijos plotį raudonasis vaizdo kanalas yra intensyvesnis už žaliajį ir mėlynąjį kanalus (žr. 2.24 pav. (b)). Tačiau kitu atveju, kai lazerio linija tampa balta, pikselių intensyvumas pasiekia maksimalią (255) reikšmę visuose trijuose kanaluose (žr. 2.24 pav. (a)). Tada nebelieka tokio pat dėsningo kaip pirmuoju atveju. Tačiau galima pastebėti, jog projektuojamas lazeris ant paviršiaus šiek tiek išsisklaido ir ties kraštais jo intensyvumas sumažėja. Dėl to kraštuose raudonasis kanalas turi didesnę intensyvumą lyginant su kitais dviem.



2.24 pav. Pikselių intensyvumo kitimo grafikai per lazerio linijos plotį (a) kai lazerio linija kameros vaizde yra balta (b) kai lazerio linija kameros vaizde yra raudona

Norint nustatyti lazerio liniją yra reikalingas tam tikras kiekvienos linijos skaitinis įvertis. Skaičiuojant šį įvertį iš pradžių lazerio linija per plotį yra suskaidoma į 3 dalis: dvi kraštines dalis ir centrą. Kiekvienos dalies plotis priklauso nuo išsaugotos linijos pločio, o išsaugotas linijos plotis priklauso nuo linijos saugojimui skirto struktūrinio elemento dydžio, kuris šiuo atveju yra 2% vaizdo skermens. Norint tinkamai suskirstyti liniją svarbu išlaikyti kiekvienos dalies proporcingumą ir tai atlikti taip, jog abiejų kraštinių dalių ilgai būtų tokie patys, tačiau ne didesni už centrinės dalies ilgį. Tai yra svarbu todėl, nes būtent kraštinėse lazerio linijos dalyse raudonasis kanalas visada bus intensyvesnis už centrinę, nepaisant kameros persotinimo.



```

function line_partition(laser_width)
    sides_len = laser_width/3
    center_len = laser_width - sides_len*2
    if center_len > sides_len:
        center_len = center_len - 2
        sides_len = sides_len + 1
    return center_len, sides_len

```

2.25 pav. Linijos dalių ilgių nustatymo kodas

Jeigu linijos plotis yra 7 pikseliai, tada kraštinių dalių ilgiai bus po 3 pikselius, o centrinės dalies ilgis – 1 pikselis. Jeigu linijos plotis yra 9 pikseliai, tada linija yra suskirstoma į 3 lygias dalis po 3 pikselius. Pagrindinė esmė, jog kraštinės dalys visada yra ilgesnės arba lygios centrinei daliai, nes lazerio linijos kraštai visada turi tą patį požymį – raudonasis kanalas yra intensyvesnis už kitus du kanalus.

Toliau yra skaičiuojamas kiekvienos linijos dalies raudonojo, žaliojo ir mėlynojo kanalų vidurkiai. Tada pagal juos atskirai yra įvertinama kiekviena iš linijos dalių. Linijos kraštinių dalių įverčiai yra apskaičiuojami taip:

$$S = \left( M_R - \frac{M_G + M_B}{2} \right) \cdot 3, \quad (2.2)$$

čia  $S$  – kraštinės dalies skaitinis įvertis, o  $M_R$ ,  $M_G$  ir  $M_B$  yra atitinkamų RGB kanalų kraštinių dalių vidurkiai. Skaičiuojant šį įvertį iš esmės iš raudonojo kanalo yra atimamas žaliojo ir mėlynojo kanalo vidurkis. Tai parodo kiek raudonasis kanalas yra intensyvesnis už likusius du. Paprasčiau tariant, kiek raudoni yra lazerio linijos kraštai. Lygiai taip pat svarbu yra viską padauginti iš konstantos 3, nes kraštinėms dalims siekiama suteikti didesnę reikšmę skaičiuojant bendrą visos linijos įvertį. Žinoma, kartais nutinka taip, jog linijos kraštai yra visiškai raudoni, padidėjęs intensyvumas yra matomas tik raudonajame kanale, dėl to žaliojo ir mėlynojo kanalo suma tampa lygi nuliui, o dalyba iš nulio yra negalima. Tokiu atveju įvertis yra prilyginamas tiesiog raudonojo kanalo vidurkiui padaugintam iš trijų (žr. 2.26 pav.).

```

function side_part_coefficient(Mr, Mg, Mb)
    S = 0
    if (Mg + Mb) == 0:
        S = Mr * 3
    else
        S = (Mr - (Mg + Mb)/2) * 3
    return S

```

2.26 pav. Kraštinių dalių įverčio skaičiavimo algoritmas

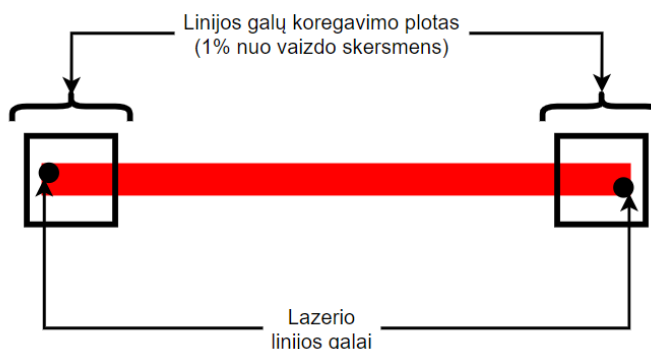
Toliau yra skaičiuojamas centrinės linijos dalies skaitinis įvertis pagal formulę:

$$C = \frac{M_R + M_G + M_B}{3}, \quad (2.3)$$

čia  $C$  yra centrinės dalies skaitinis įvertis, o  $M_R$ ,  $M_G$  bei  $M_B$  reiškia tą patį kaip ir (2.2) formulėje. Jau yra žinoma, jog lazerio vidurys gali būti raudonas arba, kameros persotinimo atveju, baltas. Esant baltai linijai visi trys RGB kanalai siekia maksimalias reikšmes ir raudonojo kanalo intensyvumas iš esmės yra lygus žaliojo ir mėlynojo kanalų intensyvumams. Dėl to (2.2) formulė tokiu atveju netinka. Dėl to linijos centrinę dalį paprasčiausia yra įvertinti pagal bendrą jos ryškumą, tai yra tiesiog visų trijų kanalų vidurkis. Tada bendras linijos įvertis yra skaičiuojamas tiesiog sudedant abiejų kraštinių dalių ir centrinės dalies įverčius. Lazerio linija yra laikoma ta linija, kurios bendras skaitinis įvertis yra didžiausias.

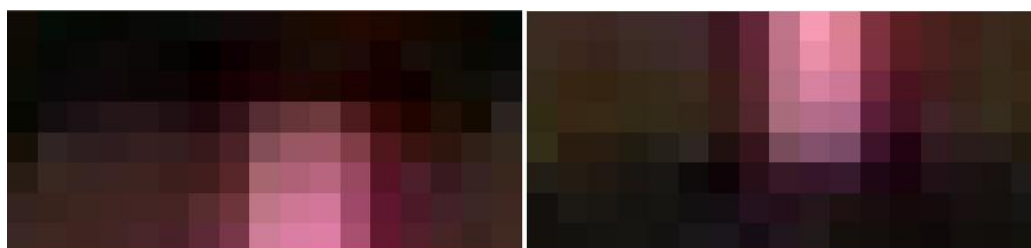
### 2.3.1. Linijos centro radimas

Aptikta lazerio linija turi 2 apskaičiuotus jos galus. Jeigu lazerio linija vaizde yra platesnė, dažnai nutinka taip, jos iš esmės vienas arba abu taškai atsiduria ties linijos kraštais. Siekiant pagerinti algoritmo tikslumą toliau yra ieškomas lazerio linijos centras.



2.27 pav. Lazerio linijos centro patikslinimas

Kaip jau žinoma, lazerio linijos centras visada bus ryškiausia vieta per visą linijos plotį. Tai galima pamatyti ir grafike (žr. 2.24 pav.). Logiška ir tai, kadangi lazeris yra šviesos šaltinis, vaizde aplink lazerį neturėtų būti ryškesnių objektų. Remiantis tuo, lazerio linijos centro patikslinimas yra iš esmės tiesiog ryškiausio lokalaus taško radimas (žr. 2.27 pav.). Lokalios zonos dydis taip pat yra apibrėžiamas pagal turimo vaizdo skersmenį, tai yra 1%.



2.28 pav. Lazerio linijos galai



Abiejose linijos taškų zonose yra vykdomas raudonojo kanalo pikselių skenavimas. Ryškiausias pikselis toje zonoje yra laikomas lazerio centru. Žinoma, tai nėra pats tiksliausias lazerio linijos centro nustatymo būdas, nes jis turi tam tikrą neapibrėžtumą. Kameros persotinimo atveju, kai raudona lazerio linija tampa balta, centre esantys pikseliai įgyja maksimalias reikšmes – 255. Dėl to yra galimos tam tikros centro nustatymo paklaidos, nes lokatioje zonoje nėra vieno pačio ryškiausio pikselio. Tokiu atveju yra parenkamas pirmas pasitaikęs ryškiausias pikselis, kuris gali būti nutolęs nuo tikrojo centro per kelis pikselius.

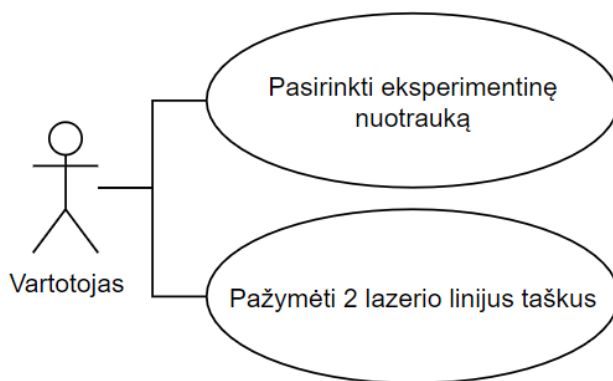
### 3. Algoritmo tyrimas

Šiame skyriuje yra aprašomas algoritmo tyrimo realizavimas. Yra rašoma apie tyrimui naudojamą sukurtą programinį įrankį. Po to yra apžvelgiama visa eksperimento eiga ir galiausiai aptariami gauti rezultatai.

#### 3.1. Tyrimo programinis įrankis

Tyrimo programinis įrankis, kaip ir pats algoritmas yra realizuotas naudojant „Matlab“ programinį paketą. Parašyta programa iš esmės turi 3 režimus: lazerio linijos aptikimas nuotraukoje, lazerio linijos aptikimas vaizdo įrašė ir lazerio linijos tyrimas, naudojant eksperimentines nuotraukas. Toliau yra rašoma būtent apie pastarąjį programos režimą, nes šis skyrius skirtas algoritmo tyrimui.

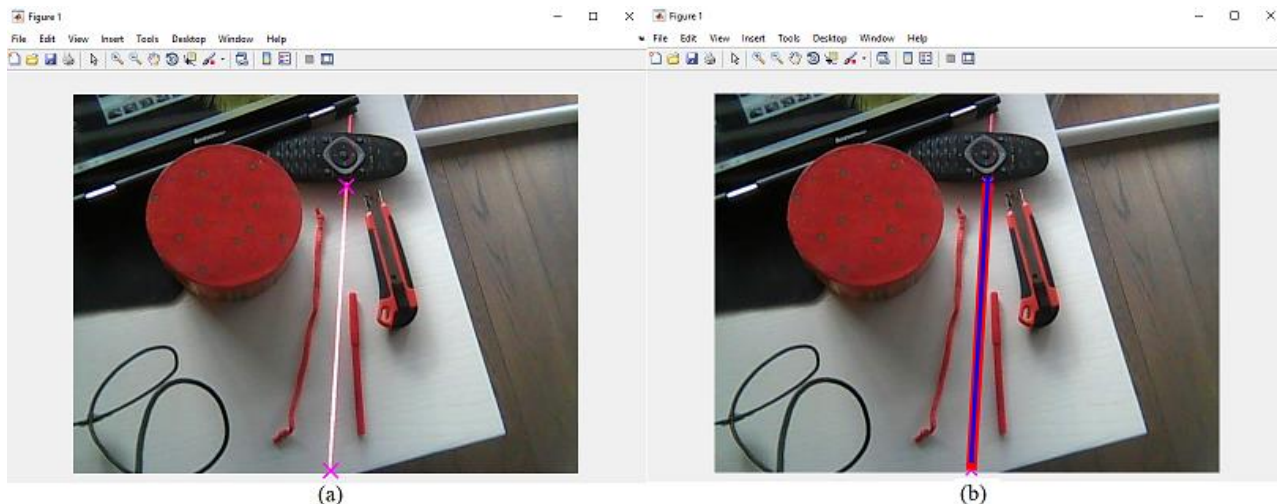
Pats tyrimo įrankis iš esmės yra labai paprastas. Vartotojas gali atlikti tik 2 veiksmus: pasirinkti eksperimentinę nuotrauką ir joje pažymėti 2 matomos lazerio linijos taškus (žr. 3.1 pav.). Prieš paleidžiant programą, vartotojas privalo programiniame kode nurodyti eksperimentinės nuotraukos buvimo vietą kompiuteryje.



3.1 pav. Panaudos atvejų diagrama

Nurodžius eksperimentinės nuotraukos buvimo vietą ir paleidus programą, toliau vartotojui yra atvaizduojama pati nuotrauka. Norint toliau tęsti bandymą, jis privalo kairiuoju pelės klavišu paspausti ant 2 nuotraukos vietų, taip nurodydamas 2 lazerio linijos taškus. Sulig kiekvienu paspaudimu taškas yra pažymimas ryškiai violetinės spalvos žymekliais. (žr. 3.2 pav. (a)). Pažymėjus antrąjį linijos tašką, programa nuotraukoje atvaizduoja vartotojo pažymėtą lazerio liniją stora, ryškiai raudona linija, o po to automatiškai yra paleidžiamas lazerio linijos aptikimo algoritmas. Algoritmui aptikus lazerio liniją, ji yra atvaizduojama toje pačioje nuotraukoje plonesne tamsiai mėlynos spalvos linija (žr. 3.2 pav. (b)).

Taip pateikta vizualinė informacija leidžia vartotojui padaryti pirmines išvadas apie lazerio linijos aptikimo algoritmo veikimą. Iš esmės, jeigu mėlyna linija (algoritmo aptikta linija) yra atvaizduota ant storesnės raudonos linijos (vartotojo pažymėtos linijos), tada galima teigti, jog algoritmas tikrai aptiko lazerio liniją. Žinoma, tai yra tik pirminis vizualinis įvertinimas, šis programinis įrankis pateikia ir detalesnę informaciją apie aptikimo kokybę.



3.2 pav. Programinio įrankio vaizdas (a) pažymėti 2 lazerio linijos taškai (b) pažymėta vartotojo nurodyta ir algoritmo aptikta lazerio linija

Siekiant įvertinti lazerio linijos aptikimo kokybę, programinis įrankis palygina vartotojo pažymėtą liniją su algoritmo aptikta linija. Palyginimas vyksta tokiu principu, jog vartotojo pažymėta linija yra laikoma atskaitos, arba, kitaip tariant, tikrąja linija, nes žmogus bet kokioje nuotraukoje intuityviai geba nustatyti, kurioje vietoje yra lazerio linija. Algoritmo aptikta linija yra galutinis gautas rezultatas.

Vienas iš aptikimo kokybės įvertinimo kriterijų yra linijos posūkio kampo paklaida. Paprasčiau tariant, yra skaičiuojama posūkio kampo absoliuti skirtumo reikšmė tarp referencinės ir aptiktos linijos. Kadangi yra žinomi abiejų linijų galutiniai taškai, tai jų kampus galima apskaičiuoti pagal formulę:

$$\alpha = \arctan \frac{y_2 - y_1}{x_2 - x_1}, \quad (3.1)$$

čia  $\alpha$  yra linijos posūkio kampas, o  $y_1, y_2, x_1, x_2$  – atitinkamai pirmojo ir antrojo taškų koordinatės. Tyrimo metu, lygiai taip, kaip jau yra apibrėžta ir algoritmo veikimo metu, linijos kampas yra skaičiuojamas nuo horizontalios pozicijos pagal laikrodžio rodyklę (žr. 2.20 pav.).

Antrasis vertinimo kriterijus yra mažiausias atstumas tarp vartotojo pažymėto linijos taško ir algoritmo aptiktos linijos taškų. Kadangi algoritmas apskaičiuoja tik galinius 2 linijos taškus, norint surasti artimiausią atstumą iki vartotojo pažymėto taško, reikia toliau suskaičiuoti visus tarpinius aptiktos linijos taškus. Tai yra daroma pasitelkiant paprasčiausią tiesės lygtį:

$$y = mx + b, \quad (3.2)$$

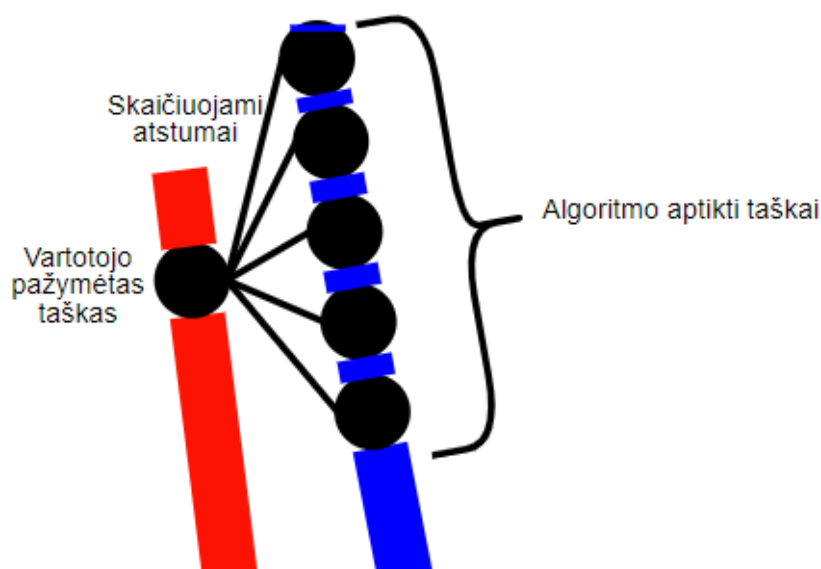
čia  $y$  ir  $x$  yra taško koordinatės,  $m$  – teisės krypties koeficientas, o  $b$  – taško, kuris kerta  $y$  ašį,  $y$  koordinatė. Toliau krypties koeficientas  $m$  yra randamas pagal aptiktos linijos 2 taškus:

$$m = \frac{y_2 - y_1}{x_2 - x_1}, \quad (3.3)$$

čia  $y_1, y_2, x_1, x_2$  reiškia tą patį kaip ir (3.1) formulėje. Tai yra algoritmo aptiktos linijos galinių taškų koordinatės. Apskaičiavus  $m$ , toliau kintamasis  $b$  yra surandamas tiesiog jį išreiškiant iš (3.2) formulės:

$$b = y - mx, \quad (3.4)$$

čia visi lygties parametrai reiškia tą patį, kaip ir (3.2) formulėje. Apskaičiavus  $m$  ir  $b$  toliau yra skaičiuojamos aptiktos linijos taškų  $y$  koordinatės pagal (3.2) lygtį. Koordinatės  $x$  reikšmės kinta nuo 1 iki turimo vaizdo pločio pikseliais. Visi apskaičiuoti linijos taškai yra saugomi vidiniame programos masyve. Toliau yra skaičiuojami Euklido atstumai tarp vartotojo pažymėto pirmojo taško ir visų algoritmo aptiktos linijos taškų (žr. 3.3 pav.). Po to iš visų tokiu būdu apskaičiuotų atstumų yra surandamas mažiausias, kuris reiškia lazerio linijos aptikimo taškų paklaidą.



3.3 pav. Atstumo tarp taškų radimas

Šis programinis įrankis, kartu su lazerio linijos aptikimo paklaidomis taip pat įvertina ir algoritmo vykdymo greitaveiką. Kaip jau minėta anksčiau, vartotojui pažymėjus antrąją linijos tašką, programa paleidžia sukurtą aptikimo algoritmą. Vykdomo laikas yra pradedamas skaičiuoti nuo to momento, kai yra iškviečiama pagrindinė algoritmo funkcija, o baigiamas, kai algoritmas apskaičiuoja du aptiktus lazerio linijos taškus. Į visą vykdymo trukmę nėra įtraukiamas atvaizdavimui skirtas laikas.

Baigus tyrimą su eksperimentine nuotrauka, rezultatai yra išvedami į programinio paketo „Matlab“ konsolės langą (žr. 3.4 pav.). Paskutinėse trijose eilutėse yra pateikiama svarbiausia tyrimo informacija, tai yra posūkio kampo paklaida, taškų atstumo paklaida bei vykdymo laikas.

```

Starting detector initialization...
Frame given with size 640x480
Initializing detector constants with sizes:
High Pass filter size: 80
Kernel size: 9
Diagonal kernel size: 7
Hough Transform discontinuance length: 40
Hough Transform minimum line length: 160
Line center searching area size: 8
Rotated kernels count 42
Initialization done
-----
Detecting...
Filtering frequencies... Done
Detecting lines on kernel rotation... Done
Acquiring pixels of lines... Done
Possible lines found: 21
Searching for laser line... Done
Line detected at points: [347 110] [326 465] with angle 93.385383
-----
Angle error: 0.196713
Point error: 7.071068
Time elapsed: 1.157090

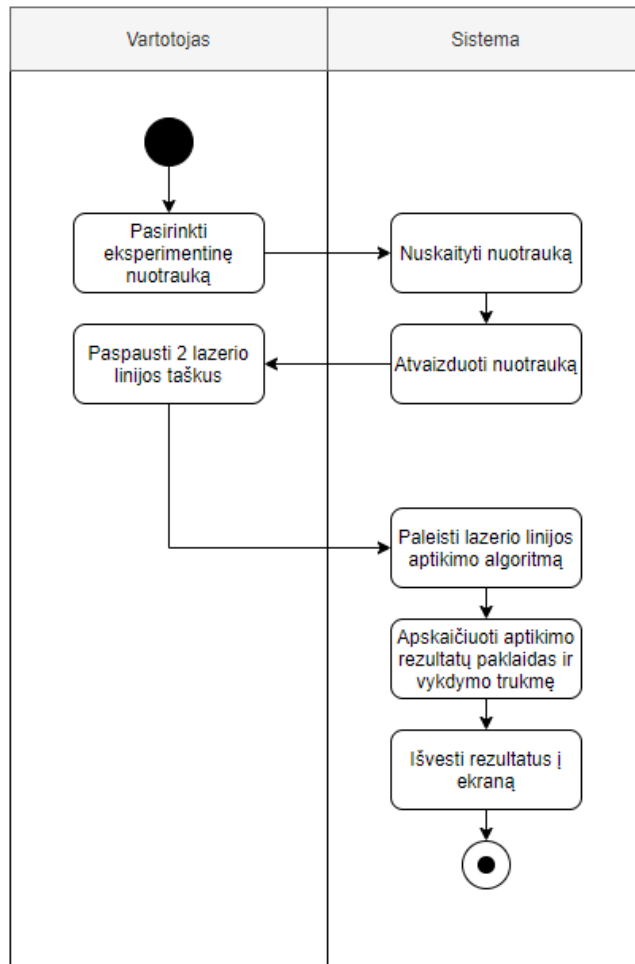
```

3.4 pav. Programinio įrankio konsolės išvestis

Šiame konsolės lange taip pat galima matyti ir kitus svarbius algoritmo parametrus. Vieni iš svarbiausių yra suinicializuotų konstantų dydžiai. Jie programiniame kode yra nurodomi kaip procentinė dalis nuo vaizdo skersmens. Dėl to yra svarbu matyti, kokius realius dydžius programa apskaičiavo. Yra 6 pagrindinės algoritmo konstantos, kurios gali būti koreguojamos programiniame kode atsižvelgiant į turimo vaizdo sąlygas:

- Aukšto dažnio filtro dydis (angl. *High Pass filter size*). Tai yra Gauso filtro, kuris naudojamas priešlaikinio apdorojimo metu, dydis.
- Branduolio dydis (angl. *Kernel size*). Tai yra branduolio, kuris naudojamas linijų aptikimui konvoliucijos metu, dydis.
- Statmeno branduolio dydis (angl. *Diagonal kernel size*). Tai yra branduolio, kuris naudojamas linijai išsaugoti, dydis.
- Hofo transformacijos linijos pertraukimo ilgis (angl. *Hough Transform discontinuance length*). Tai yra maksimalus linijos pertraukimo ilgis, iki kurio skirtingi linijos segmentai yra laikomi kaip viena linija.
- Hofo transformacijos minimalus linijos ilgis (angl. *Hough Transform minimum line length*). Tai yra linijos, kuri gali būti aptikta, minimalus ilgis.
- Linijos centro radimo paieškos lauko dydis (angl. *Line center searching area size*). Tai yra paieškos lauko dydis, kuris naudojamas linijos centro radimui.

Programos išvestyje taip pat galima matyti sugeneruotų pasuktų branduolių skaičių (angl. *Rotated kernels count*). Viso vykdymo metu taip pat yra spausdinama algoritmo veikimo esama stadija bei jau įvykdytos stadijos. Ši informacija yra naudinga vartotojui esant labai aukštos rezoliucijos vaizdams. Tokiais atvejais algoritmo vykdymo laikas gali apimti keliasdešimt sekundžių, todėl galima vizualiai matyti, kuri algoritmo dalis yra vykdoma ilgiausiai.



3.5 pav. Programinio įrankio veiklos diagrama

Apibendrintas sukurto programinio įrankio veikimas yra pavaizduotas veiklos diagrama (žr. 3.5 pav.). Joje galima matyti apibendrintą visą tyrimo eigą. Diagramoje taip pat atsispindi vartotojo ir algoritmo sąryšis. Gauti rezultatai iš esmės yra priklausomi ir nuo pačio vartotojo, nes rezultatams turi įtakos ir tai, kaip tiksliai vartotojas pažymi abu lazerio linijos taškus.

### 3.2. Tyrimo eiga

Tyrimui yra naudojama 30 eksperimentinių nuotraukų. Pusė jų yra užfiksuota naudojant Logitech C210 vaizdo kamerą su 640x480 rezoliucija, kita pusė – naudojant Logitech Pro 9000 su 320x180 rezoliucija.

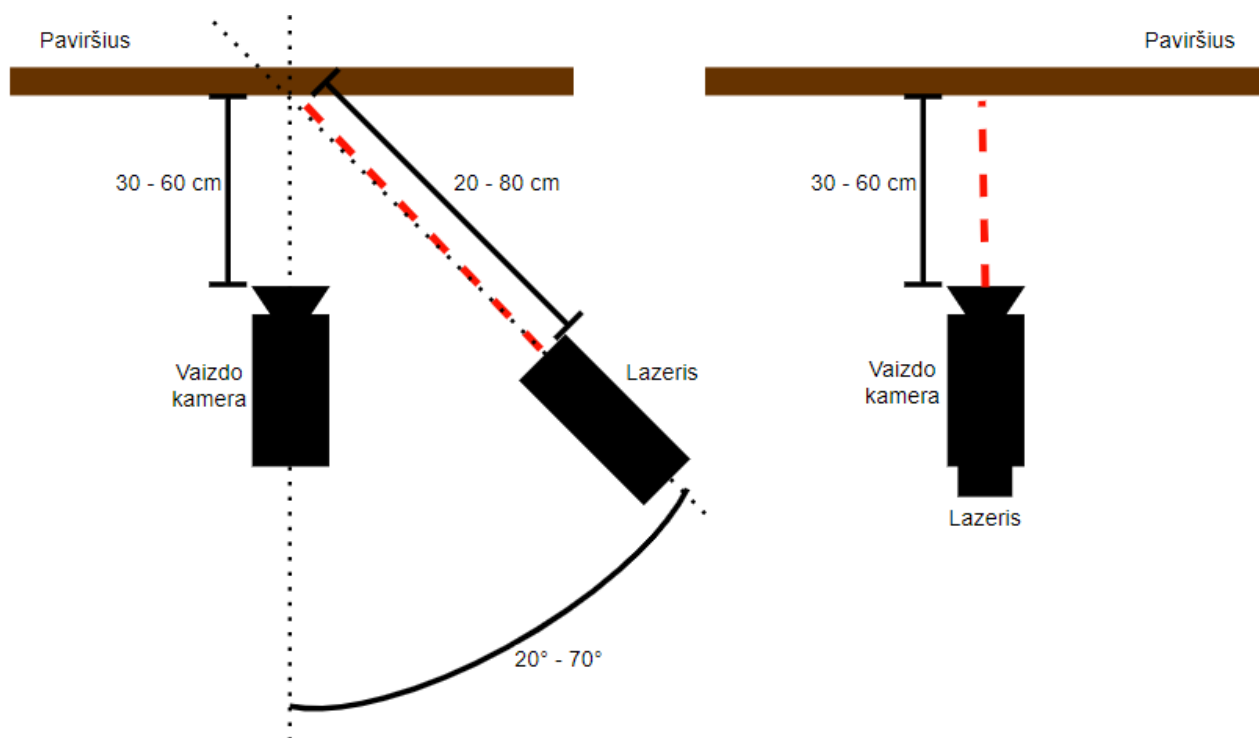


3.6 pav. Naudojamos internetinės vaizdo kameros (a) Logitech C210 [20] (b) Logitech Pro 9000 [21]

Vienas iš tyrimo tikslų taip pat yra išsiaiškinti algoritmo veikimo diapazoną. Dėl to darant eksperimentines nuotraukas, kiekvienoje yra stengiamasi sukurti vis skirtingas fono bei lazerio linijos projekcijos sąlygas. Eksperimentinėse nuotraukose atsispindi šios sąlygos arba jų deriniai:

- Lazerio linija yra stipriai atspindima, jos centras kameros vaizde tampa baltas;
- Lazerio linija blankiai matoma fone;
- Fone yra kitų raudonų objektų;
- Lazerio linija gali būti šiek tiek išlinkusi;
- Galimi trumpi lazerio linijos pertrūkimai.

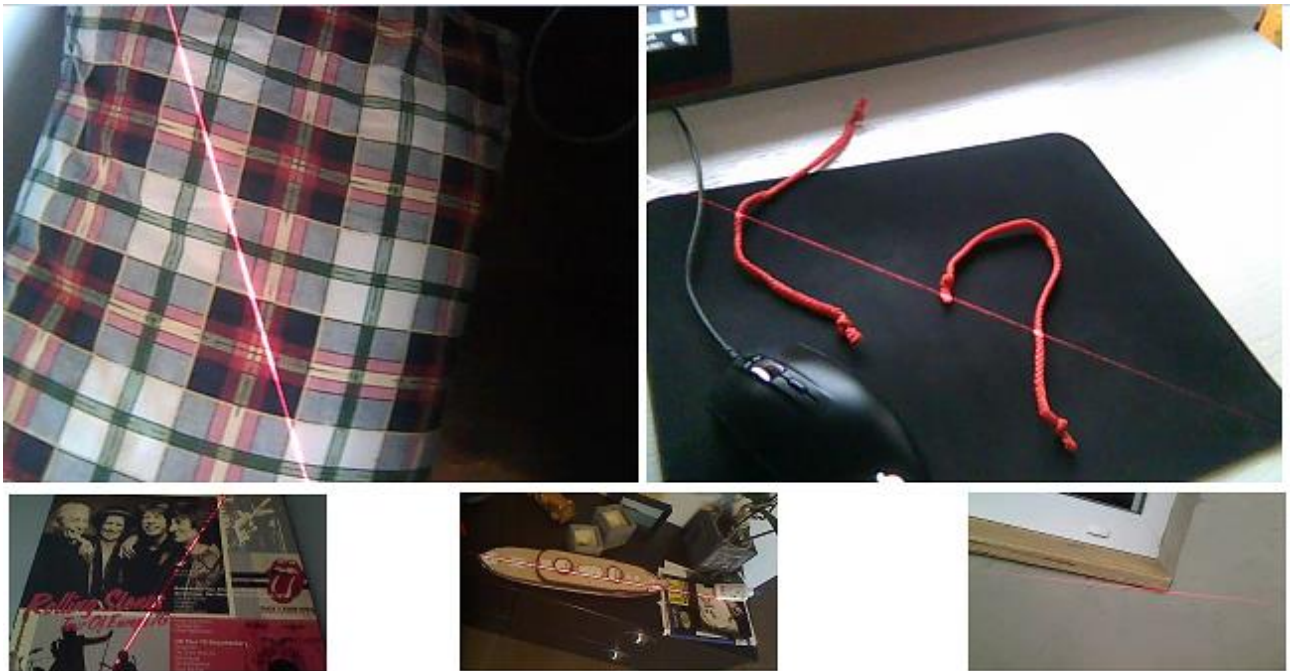
Eksperimentinės nuotraukos yra fiksuojamos dviem skirtingais būdais (žr. 3.7 pav.). Pirmasis yra, kai vaizdo kamera ir lazerio šviesą skleidžiantis prietaisas sudaro tam tikrą kampą, o antrasis, kai lazerio prietaisas ir kamera yra laikomi tame pačiame taške. Tokie 2 skirtingi išdėstymo būdai sukuria skirtingas sąlygas, kai paviršius, ant kurio projektuojama lazerio linija, nėra lygus. Pavyzdžiui, naudojant pirmąjį išdėstymo būdą, projektuojant lazerio liniją ant sferinio paviršiaus, ji išlinksta.



3.7 pav. Vaizdo kameros ir lazerio išdėstymo būdai

Kampas tarp vaizdo kameros ir lazerio linijos prietaiso nėra griežtai apibrėžtas, darant visas eksperimentines nuotraukas, jis svyruoja nuo apytiksliai nuo  $20^\circ$  iki  $70^\circ$ . Platus atstumas tarp kameros ir paviršiaus, ant kurio projektuojama lazerio linija, diapazonas taip pat sukuria skirtingas sąlygas kiekvienoje eksperimentinėje nuotraukoje. Nuo šio atstumo priklauso vaizde matomos lazerio linijos storis. Dėl to fiksuojant eksperimentinius vaizdus kamera yra laikoma apytiksliai 30 – 60 cm atstumu. Darant šias nuotraukas lazerio linijos prietaisas ir internetinė kamera yra laikomi tiesiog rankose.





3.8 pav. Eksperimentinių nuotraukų pavyzdžiai

Kaip jau žinoma, lazerio linijos aptikimo algoritmas turi 6 skirtingus parametrus, kurie gali būti derinami atsižvelgiant į turimo vaizdo sąlygas. Jie yra nurodomi kaip procentinė dalis nuo turimo vaizdo skersmens. Prieš pradėdant tyrimą yra nustatytos tokios šių parametrų reikšmės:

- Aukšto dažnio filtro dydis – 10%;
- Branduolio dydis – 1%;
- Statmeno branduolio dydis – 0,75%;
- Hofo transformacijos linijos pertraukimo ilgis – 5%;
- Hofo transformacijos minimalus linijos ilgis – 20%;
- Linijos centro radimo paieškos lauko dydis – 1%.

Viso tyrimo metu, nepaisant tiriamo vaizdo sąlygų, šie parametrai toliau nėra keičiami. Atliekant eksperimentą toliau yra saugomi kiekvienos eksperimentinės nuotraukos lazerio linijos aptikimo kampo paklaida, mažiausias atstumas tarp vartotojo pažymėto taško ir algoritmo aptiktos linijos taškų bei vaizdo apdorojimo laikas. Šie rezultatai yra pateikiami kitame poskyryje.

### 3.3. Tyrimo rezultatai

Lazerio linija vaizde yra laikoma aptikta, kai jos posūkio kampo paklaida yra mažesnė, nei  $3^\circ$ , o mažiausias atstumas tarp taškų nedidesnis nei 15. Jeigu gauti rezultatai tenkina numatytus rėžius, programiniame įrankyje vizualiai yra matoma, jog sukurto algoritmo pažymėta linija apytiksliai atitinka nuotraukoje matomą lazerio liniją.

Šių dviejų parametrų pilnai pakanka norint nustatyti, ar algoritmas aptiko lazerio liniją vaizde. Jeigu posūkio kampo paklaida yra mažesnė nei  $3^\circ$ , tačiau mažiausias atstumas tarp taškų yra didesnis nei 15, tada reiškia, jog aptikta lazerio linija yra visai kitoje vaizdo vietoje, nei vartotojo pažymėta linija. O jeigu taškų atstumas yra mažesnis nei 15, bet posūkio kampas didesnis nei  $3^\circ$ , tai reiškia, jog



pažymėta ir aptikta linijos susikerta, tačiau linijos kampas nesutampa. Taigi, jeigu šie du įverčiai atitinka numatytus režius, galima teigti, jog algoritmas aptiko lazerio liniją.

Eksperimentinės nuotraukos iš esmės yra suskirstytos į dvi grupes pagal rezoliuciją ir internetinę vaizdo kamerą, kuria buvo užfiksuoti vaizdai. Pirmoje lentelėje (žr. 3.1 lentelė) yra pateikta penkiolikos nuotraukų, kurios buvo darytos su Logitech Pro 9000 internetine kamera, rezultatai o antroje (žr. 3.2 lentelė) kitos 15 nuotraukų, darytos su Logitech C210 internetine kamera. Šiose lentelėse yra įtraukti tik tų nuotraukų rezultatai, kuriose lazerio linija buvo aptikta.

### 3.1 lentelė 320x180 rezoliucijos nuotraukų tyrimo rezultatai

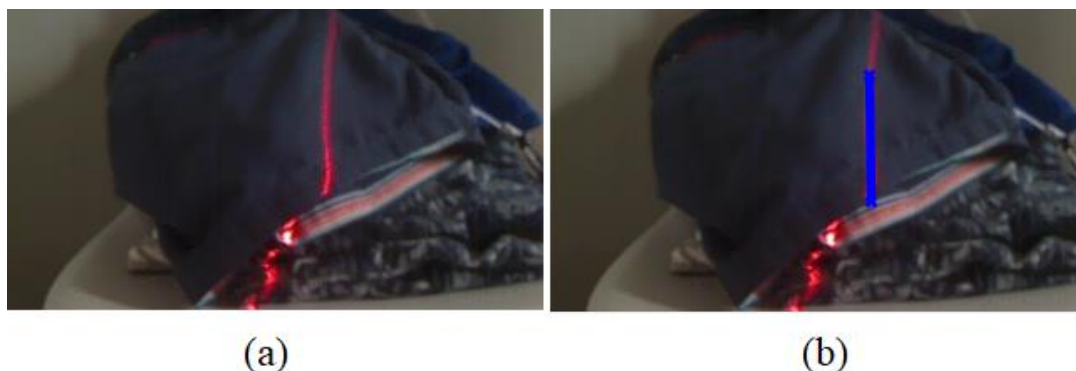
	Kampo paklaida (°)	Mažiausias atstumas tarp taškų	Vykdyto laikas (s)
Minimali reikšmė	0.042753	0.0261194	0.125343
Maksimali reikšmė	1.696417442	8.06225775	0.17979
Vidurkis	0.491141838	1.68716303	0.15398

### 3.2 lentelė 640x480 rezoliucijos nuotraukų tyrimo rezultatai

	Kampo paklaida	Mažiausias atstumas tarp taškų	Vykdyto laikas (s)
Minimali reikšmė	0.003570542	0.32142857	0.93297
Maksimali reikšmė	1.717366996	13.3059905	1.326736
Vidurkis	0.466379333	2.41047511	1.093825

Remiantis gautais tyrimo rezultatais, galima teigti, jog lazerio linija yra aptinkama pakankamai tiksliai. Abiejų nuotraukų rinkinių posūkio kampo paklaidos vidurkis neviršija nei pusės laipsnio ( $0.49^\circ$  ir  $0.47^\circ$ ). Minimalaus taško atstumo vidurkiai taip pat yra sąlyginai maži (1.69 ir 2.41).

Rezultatuose galima pastebėti, jog mažiausio atstumo tarp taškų maksimali reikšmė yra apytiksliai 5 kartus didesnė už vidurkį. Toks rezultatų netolydumas gali būti paaiškinamas tam tikru eksperimento neapibrėžtumu. Kai kuriuose eksperimentinėse nuotraukose lazerio linija yra praradusi tiesės formą (žr. 3.9 pav.). Tokiu atveju vartotojui yra sudėtinga pažymėti lazerio liniją kaip tiesę. Algoritmas tada aptinka tiesiausią lazerio linijos segmentą, kuris gali šiek tiek nesutapti su vartotojo pažymėtais taškais.



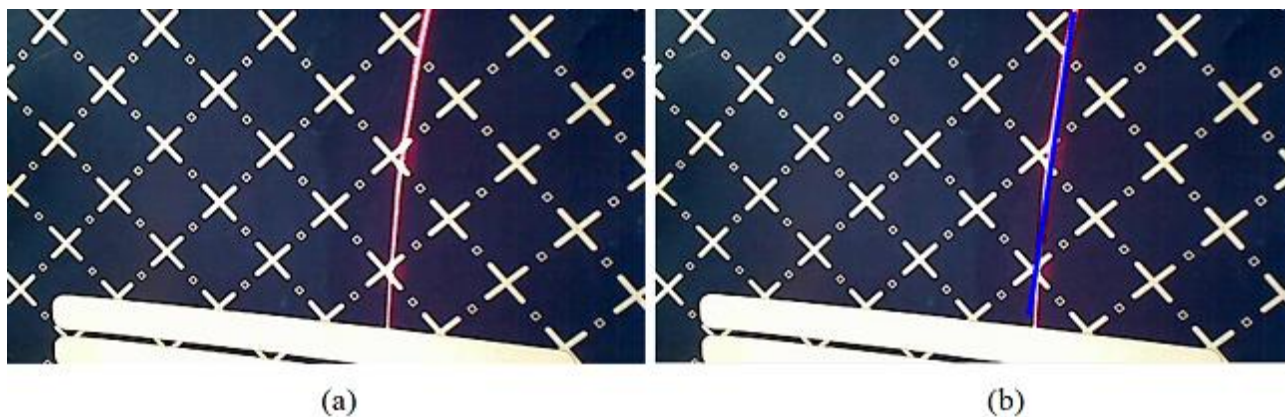
3.9 pav. Eksperimentinė nuotrauka (a) originalus vaizdas (b) aptikta lazerio linija

Remiantis vykdymo laiko tyrimo rezultatais darosi aišku, jog vaizdo apdorojimo laikas yra tiesiogiai priklausomas nuo turimo vaizdo dydžio. Tačiau vykdymo laiką taip pat įtakoja ir vaizdo sudėtingumas. Tai reiškia, jeigu vaizdas yra dinaminis, turintis daug briaunų, pikselių intensyvumas fone nuolat kinta, vaizdas bus apdorojamas kur kas ilgiau. Žinoma, tam tikrų algoritmo dalių vykdymo laikas, pavyzdžiui, dažnių spektro filtravimas, visiškai nepriklauso nuo to, kas yra matoma vaizde. Tačiau Hofo transformacijos greitaveika yra priklausoma nuo įvesties binarinio vaizdo. Kuo daugiau turima taškų binariniame vaizde, tuo Hofo transformacija ilgiau trunka jį apdoroti.

Šalia 30 anksčiau minėtų eksperimentinių nuotraukų, taip pat turima daugiau kaip 20 papildomų vaizdų, kurie buvo naudojami algoritmo kūrimo metu bei papildomiems testavimams. Kaip jau žinoma, lazerio linijos aptikimas yra sudėtingas uždavinys, jeigu naudojamas paprastas spalvų segmentavimas, nes tam tikrais atvejais raudona lazerio linija tampa balta. Dėl to pasiūlytas algoritmas spalvų segmentavimą vykdo tik ties linijos kraštais. Siekiant išanalizuoti 2.3 poskyryje pateiktą idėją, buvo padaryta papildoma eksperimentinė nuotrauka (žr. 3.10 pav.). Šiame vaizde yra matoma lazerio linija, kuri dėl nepakankamo kameros jautrumo yra balta, kuri taip pat yra išlinkusi. Fone yra matomi įvairūs balti ženkliniai, linijos. Algoritmas puikiai aptiko lazerio liniją ir čia, nepaisant jos išlinkimo. Gauti rezultatai yra tokie:

- Posūkio kampo paklaida –  $0.91^\circ$
- Mažiausias atstumas tarp taškų – 3.77
- Vykdyto trukmė – 0.72s

Tokius rezultatus yra ganėtinai sunku interpretuoti, nes linija yra išlinkusi. Tačiau, kaip matoma, gauti rezultatai tenkina tyrimo pradžioje nustatytus režius.



3.10 pav. Papildoma eksperimentinė nuotrauka (a) originalus vaizdas (b) aptikta lazerio linija

Kitos nuotraukos yra skirtos siekiant suklaidinti pasiūlytą algoritmą (žr. 3.11 pav. (a)). Šiuo atveju, naudojant tuos pačius algoritmo parametrus, algoritmas aptiko vaizde matomą raudoną virvutę. Tai būtų galima paaiškinti tuo, jog vaizde matoma virvutė vizualiai yra ryškesnė už pačią lazerio liniją, o apačioje ryškiai matomas lazerio segmentas tiesiog neatitinka minimalaus linijos ilgio parametro. Gauti rezultatai yra tokie:

- Posūkio kampo paklaida –  $3.31^\circ$
- Mažiausias atstumas tarp taškų – 152.06
- Vykdyto trukmė – 1.03s



3.11 pav. Nepavykęs eksperimentas (a) originali nuotrauka (b) klaidingai aptikta lazerio linija

Kaip jau minėta anksčiau, algoritmo parametrus galima koreguoti, atsižvelgiant į vaizdo sąlygas. Taigi, padidinus konvoliucijos branduolio dydį iki 5%, o minimalų linijos ilgį iki 40%, algoritmas sėkmingai aptiko lazerio liniją tame pačiame vaizde. Gauti rezultatai yra tokie:

- Posūkio kampo paklaida –  $0.51^\circ$ ;
- Mažiausias atstumas tarp taškų – 5.11;
- Vykdomo trukmė – 1.41s.

Šiuo atveju didesnis konvoliucijos branduolys kokybiškiau išfiltruoja triukšmą iš vaizdo, o padidintas minimalus linijos ilgis sumažina Hough transformacijos aptiktų linijų skaičių. Tačiau tokie parametrai turi ir neigiamą poveikį visam algoritmui. Naudojant didesnę branduolį, lazerio linija privalo būti idealiai tiesi, kai tuo tarpu mažesnis branduolys leidžia aptikti ir išlinkusias linijas. Didelis minimalus linijos ilgis taip pat riboja lazerio linijos projektavimo ant paviršiaus galimybes.

Taigi, galima teigti, jog buvo pasiektas tam tikras algoritmo adaptyvumo lygis, nes lazerio linija yra sėkmingai aptinkama skirtingų rezoliucijų vaizduose, naudojant tuos pačius algoritmo parametrus. Algoritmas taip pat yra universalus. Jo parametrai gali būti derinami atsižvelgiant į turimo vaizdo sąlygas.

## Išvados

1. Atlikus jau pasiūlytų lazerio linijos metodų apžvalgą, buvo pastebėta, jog daugelis jų yra paremti spalvų segmentavimo arba slenkstinių ribų metodais. Pateiktuose pavyzdžiuose yra matoma, jog lazerio linija yra projektuojama esant nesudėtingoms fono sąlygoms. Naudojant tokius metodus, slenkstinės ribos turi būti priderinamos atsižvelgiant į turimo vaizdo sąlygas. Dėl to galima daryti išvadą, jog išanalizuoti lazerio linijos aptikimo algoritmai nėra adaptyvūs, reikalaujantys tikslaus parametrų derinimo.
2. Buvo atlikta 3 aukšto dažnio filtrų (idealiojo, Gauso ir Butterworth'o) filtrų analizė. Remiantis analizės rezultatais buvo nuspręsta, jog Gauso filtras yra optimaliausias sprendžiant lazerio linijos aptikimo uždavinį. Jis po filtravimo vidutiniškai vaizde palieka mažiausią triukšmo dalį – 58144 pikseliai. Tuo tarpu idealusis ir Butterworth'o filurai – atitinkamai 77448 ir 64146 pikseliai. Gauso filtras taip pat daro mažiausią įtaką lazerio linijai priklausantiems pikseliams. Jis vidutiniškai išfiltruoja 10,709% linijai priklausančių pikselių, o idealusis ir Butterworth'o filurai – atitinkamai 34,773% ir 29,698%.
3. Algoritmas sėkmingai realizuotas naudojant *Matlab* programinį paketą. Pasiūlytas algoritmas turi 6 skirtingus parametrus, kurie gali būti derinami atsižvelgiant į apdorojamo vaizdo sąlygas. Šiuos parametrus galima suderinti taip, jog algoritmas veikai pakankamai adaptyviai. Lazerio linija sėkmingai aptinkama esant įvairioms sudėtingoms sąlygoms. Linija gali būti šiek tiek išlinkusi, pertrūkusi, silpnai arba ryškiai matoma. Esminėse algoritmo dalyse nėra naudojami spalvų segmentavimo metodai, dėl to kameros persotinimo atveju (kai raudona lazerio linija tampa balta) lazerio linija yra taip pat sėkmingai aptinkama.
4. Pasiūlytas lazerio linijos aptikimo algoritmas buvo ištirtas naudojant programinį įrankį, kuris taip pat buvo realizuotas *Matlab* programiniu paketu. Lazerio linija sėkmingai aptikta trisdešimtyje eksperimentinių nuotraukų. Vidutinė aptiktos linijos posūkio kampo paklaida yra  $0,478761^\circ$ , o vidutinis trumpiausias atstumas tarp pažymėto taško ir aptiktos linijos taškų yra 2,048819. Apdorojant skirtingų rezoliucijų vaizdus, vidutinis vaizdo apdorojimo laikas siekia 0,623902 s.

## Literatūros sąrašas

1. LOW, C., Zamzuri, H. and Amri Mazlan, S. Simple robust road lane detection algorithm. [interaktyvus] [žiūrėta 2021-05-01] Prieiga per: <https://ieeexplore.ieee.org/document/6869550>
2. GONZALEZ, R. C and WOODS, R. E., 2008, Digital image processing. 3. New Jersey : Parson. [interaktyvus] [žiūrėta 2021-05-01] Prieiga per: [http://sdeuoc.ac.in/sites/default/files/sde\\_videos/Digital%20Image%20Processing%203rd%20ed.%20-%20R.%20Gonzalez%2C%20R.%20Woods-ilovepdf-compressed.pdf](http://sdeuoc.ac.in/sites/default/files/sde_videos/Digital%20Image%20Processing%203rd%20ed.%20-%20R.%20Gonzalez%2C%20R.%20Woods-ilovepdf-compressed.pdf)
3. CANNY, J, 1968, A Computational Approach to Edge Detection [interaktyvus]. [žiūrėta 2021-05-01]. Prieiga per: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf>
4. DERVINIS, Donatas, Vaizdų apdorojimas. [interaktyvus]. 2012. [žiūrėta 2021-05-01]. Prieiga per: <https://www.ebooks.ktu.lt/eb/451/vaizdu-apdorojimas/>
5. PULFER, Erich-Matthew, Different Approaches to Blurring Digital Images and Their Effect on Facial Detection. Scholarworks.uark.edu [interaktyvus]. 2019. [žiūrėta 2021-05-01]. Prieiga per: <https://scholarworks.uark.edu/cgi/viewcontent.cgi?article=1067&context=csceugt>
6. ZHANG, Liguo ir kt., 2014, A novel laser vision sensor for weld line detection on wall-climbing robot. [interaktyvus]. [žiūrėta 2021-05-01]. Prieiga per: <https://www.sciencedirect.com/science/article/pii/S0030399214000061>
7. FOSSUM, Eric R. and HONDONGWA, Donald B. A Review of the Pinned Photodiode for CCD and CMOS Image Sensors [interaktyvus]. 2014. [žiūrėta 2021-05-01]. Prieiga per: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6742594>
8. GUO, Hongwei, 2011, A Simple Algorithm for Fitting a Gaussian Function. [interaktyvus]. 2011. [žiūrėta 2021-05-01]. Prieiga per: [https://www.researchgate.net/publication/252062037\\_A\\_Simple\\_Algorithm\\_for\\_Fitting\\_a\\_Gaussian\\_Function\\_DSP\\_Tips\\_and\\_Tricks](https://www.researchgate.net/publication/252062037_A_Simple_Algorithm_for_Fitting_a_Gaussian_Function_DSP_Tips_and_Tricks)
9. UTAMININGRUM, Fitri ir kt., Fast obstacle distance estimation using laser line imaging technique for smart wheelchair. [interaktyvus]. 2016. [žiūrėta 2021-05-01]. Prieiga per: [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](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)
10. BARRETO, Saulo Vinicius Ferreira, Sant'Anna, Remy Eskinazi and Feitosa, Marcilio Andre Felix, A method for image processing and distance measuring based on laser distance triangulation. [interaktyvus]. [žiūrėta 2021-05-01]. Prieiga per: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6815509&tag=1>
11. CHMELAR, Pavel, Bean, Ladislav and Kudriatseva, Natalia, The laser color detection for 3D range scanning using Gaussian mixture model. [interaktyvus]. [žiūrėta 2021-05-01]. Prieiga per: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7129023&tag=1>
12. MEŠKO, Matej and, Štefan Toth, Laser Spot Detection. [interaktyvus]. 2013. [žiūrėta 2021-05-01]. Prieiga per: [https://www.researchgate.net/publication/350124875\\_Laser\\_spot\\_detection](https://www.researchgate.net/publication/350124875_Laser_spot_detection)
13. Defining Connectivity [žiūrėta 2020m. sausio 14d.] Internetinė prieiga: [http://www.imageprocessingplace.com/downloads\\_V3/root\\_downloads/tutorials/contour\\_tracig\\_Abeer\\_George\\_Ghuneim/connect.html](http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracig_Abeer_George_Ghuneim/connect.html)

14. "Fast Fourier transform – FFT", Cooley-Tukey technique, Article. 10, "A simple, pedagogical radix-2 algorithm in C++". [žiūrēta 2020-06-02]. Internetinė prieiga: <http://www.librow.com/articles/article-10>
15. CHMELAR, Pavel and, Martin Dobrovolny, The Laser Line Detection for Autonomous Mapping Based on Color Segmentation. Publications.waset.org [interaktyvus]. [žiūrēta 2021-05-01]. Prieiga per: <https://publications.waset.org/9996622/the-laser-line-detection-for-autonomous-mapping-based-on-color-segmentation>
16. BIJMAN, A, A low cost 3D laser-line scanner for facial acquisition. [interaktyvus]. [žiūrēta 2021-05-01]. Prieiga per: [https://essay.utwente.nl/72064/1/Bijman\\_BA\\_EEMCS.pdf](https://essay.utwente.nl/72064/1/Bijman_BA_EEMCS.pdf)
17. FISHER, R, Image Transforms - Fourier Transform. [interaktyvus]. 2003. [žiūrēta 2021-05-01]. Prieiga per: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm>
18. Concepts - NI Vision 2011 Concepts Help - National Instruments, [interaktyvus]. [žiūrēta 2021-05-01]. Prieiga per: [https://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/edge\\_detection\\_concepts/](https://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/edge_detection_concepts/)
19. CHEN, C, Digital Image Processing using Fourier Transform in Python, [interaktyvus]. [žiūrēta 2021-05-01]. Prieiga per: <https://hicraigchen.medium.com/digital-image-processing-using-fourier-transform-in-python-bcb49424fd82>
20. Cnet, Logitech C210 Webcam Specs and Prices, [interaktyvus]. [žiūrēta 2021-05-22]. Prieiga per: <https://www.cnet.com/products/logitech-c210-webcam/>
21. Cnet, Logitech Webcam Pro 9000 Specs and Prices, [interaktyvus]. [žiūrēta 2021-05-22]. Prieiga per: <https://www.cnet.com/products/logitech-webcam-pro-9000/>